

32nd International Conference on DNA Computing and Molecular Programming

DNA 32, August 3–7, 2026, Fayetteville, Arkansas, US

Edited by

Dominic Scalise

Robert Schweller



Editors

Dominic Scalise 

Washington State University, Pullman, WA, USA
dominic.scalise@wsu.edu

Robert Schweller 

University of Texas Rio Grande Valley, Edinburg, TX, USA
robert.schweller@utrgv.edu

ACM Classification 2012

Theory of computation → Models of computation; Applied computing → Biological networks; Applied computing → Molecular structural biology; Information systems → Information storage systems

ISBN 978-3-95977-444-4

Published online and open access by

Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at <https://www.dagstuhl.de/dagpub/978-3-95977-444-4>.

Publication date

July, 2026

Bibliographic information published by the Deutsche Nationalbibliothek

The Deutsche Nationalbibliothek lists all publications of this volume in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at <https://portal.dnb.de>.

License

This work is licensed under a Creative Commons Attribution 4.0 International license (CC-BY 4.0):
<https://creativecommons.org/licenses/by/4.0/legalcode>.



In brief, this license authorizes each and everybody to share (to copy, distribute and transmit) the work under the following conditions, without impairing or restricting the authors' moral rights:

- Attribution: The work must be attributed to its authors.

The copyright is retained by the corresponding authors.

Digital Object Identifier: 10.4230/LIPIcs.DNA.32.0

ISBN 978-3-95977-444-4

ISSN 1868-8969

<https://www.dagstuhl.de/lipics>

LIPICs – Leibniz International Proceedings in Informatics

LIPICs is a series of high-quality conference proceedings across all fields in informatics. LIPICs volumes are published according to the principle of Open Access, i.e., they are available online and free of charge.

Editorial Board

- Christel Baier (TU Dresden, DE)
- Roberto Di Cosmo (Inria and Université Paris Cité, FR)
- Faith Ellen (University of Toronto, CA)
- Javier Esparza (TU München, DE)
- Holger Hermanns (Universität des Saarlandes, Saarbrücken, DE and Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Wadern, DE)
- Daniel Král' (Leipzig University, DE and Max Planck Institute for Mathematics in the Sciences, Leipzig, DE – Chair)
- Sławomir Lasota (University of Warsaw, PL)
- Meena Mahajan (Institute of Mathematical Sciences, Chennai, IN)
- Chih-Hao Luke Ong (Nanyang Technological University, SG)
- Eva Rotenberg (IT University of Copenhagen, DK)
- Pierre Senellart (ENS, Université PSL, Paris, France)
- Alexandra Silva (Cornell University, Ithaca, US)

ISSN 1868-8969

<https://www.dagstuhl.de/lipics>

Contents

Preface	
<i>Dominic Scalise and Robert Schweller</i>	0:vii
Organization	
.....	0:ix
List of Authors	
.....	0:xv

Papers

Sequential Non-Determinism in Tile Self-Assembly: A General Framework and an Application to Efficient Temperature-1 Self-Assembly of Squares	
<i>David Furcy and Scott M. Summers</i>	1:1–1:18
Powers and Limitations of Synchronous Self-Assembly: Non-cooperative Assemblies and Limited Synchronization	
<i>Florent Becker, Phillip Drake, Matthew J. Patitz, and Ryder Smith</i>	2:1–2:23
Shape Reachability in Oritatami Is Complete for P	
<i>Sota Kano and Shinnosuke Seki</i>	3:1–3:18
PENSIm: A Toolkit for PEN-DNA Systems	
<i>Gwendal Ducloz and Nicolas Schabanel</i>	4:1–4:24
Geometric Constraint Optimization for Localized Strand Displacement Reactions	
<i>Matthew R. Lakin</i>	5:1–5:23
Reverse-Robust Computation with Chemical Reaction Networks	
<i>Ravi Kini and David Doty</i>	6:1–6:16
Scaling up Thermodynamically Favoured Scaffolded DNA Computing by Sculpting the Energy Landscape	
<i>Joshua Petrack, Constantine G. Evans, Angel Cervera Roldan, Mahboobeh Enayati, and Damien Woods</i>	7:1–7:22
Amplification at Equilibrium: Structural and Thermodynamic Limitations, and Implementation	
<i>Hamidreza Akef, Chia-Yu Sung, Aneesh Vanguri, and David Soloveichik</i>	8:1–8:24
Thermodynamic Binding Networks Capture Equilibrium Monomer-Polymer Thermodynamics	
<i>And Kaan Ata Yilmaz, David Soloveichik, and Austin Luchsinger</i>	9:1–9:20
Modularity of Signal Propagation Networks in the Thermodynamic Binding Network Model	
<i>And Kaan Ata Yilmaz and David Soloveichik</i>	10:1–10:23
Universal Spatial Distribution Approximation in Equilibrium Lattice Models with Local Pairwise Interactions	
<i>Cameron Chalk and Erik Winfree</i>	11:1–11:24

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Scalable Enumeration of Pareto-Optimal Polymers for Computing Equilibrium Concentrations
 Archit Patil, Minki Hhan, and David Soloveichik 12:1–12:24

Contrastive Hebbian Learning for Multicomponent Liquids
 Yancheng Du, Cameron Chalk, Salvador Buse, Lulu Qian, and Erik Winfree 13:1–13:27

Bounded Analog Complexity
 Ho-Lin Chen and Xiang Huang 14:1–14:22

Preface

This LIPIcs volume contains the papers that have been accepted to Track A of the 32nd International Conference on DNA Computing and Molecular Programming (DNA 32), organized by Matt Patitz and his team at the University of Arkansas, Fayetteville, Arkansas, US, during August 3–7, 2026, under the auspices of the International Society for Nanoscale Science, Computation, and Engineering (ISNSCE).

The DNA conference series aims at providing researchers from the fields of mathematics, computer science, physics, chemistry, biology, and nanotechnology, among others, with a forum to collaboratively address the analysis, design, and synthesis of information-based molecular systems. Papers and presentations were sought in all areas that relate to biomolecular computing including, but not restricted to, algorithms and models of computation for biomolecular systems, computational processes *in vitro* and *in vivo*, molecular switches, gates, devices, and circuits, molecular folding and self-assembly of nanostructures, analysis and theoretical models of laboratory techniques, molecular motors and molecular robotics, information storage, studies of fault-tolerance and error correction, software tools for analysis, simulation, and design, synthetic biology and *in vitro* evolution, and applications in engineering, physics, chemistry, biology, and medicine.

Authors who wished to orally present their work were asked to select one of the two submission tracks: Track A (full paper) or Track B (one-page abstract accompanied by a supplementary document). Track B is primarily for those who plan to submit experimental or theoretical results to a journal rather than publish in the conference proceedings. There were 42 submissions for oral presentations (23 to Track A and 19 to Track B), from Belgium, Canada, China, France, Germany, Iraq, Ireland, Italy, Japan, South Korea, Sweden, Taiwan, the UK, and the USA. Each of the 42 qualified submissions was reviewed by at least three reviewers, with most reviewed by four or more, and thoroughly discussed by the Program Committee (PC). The committee decided to accept 14 papers for Track A (61%) and 13 submissions for Track B (68%). We also received 30 submissions for Track C (poster), of which 6 were selected as additional short oral presentations.

We warmly thank Yann Ponty, Chris Thachuk, Fei Zhang, Chaim Goodman-Strauss, and Shelley Wickham for their intellectually stimulating invited talks, and all the authors of the submissions for making DNA 32 successful. As the PC co-chairs, we would like to express our gratitude towards the PC members and the external reviewers for their hard work in reviewing the papers within a tight schedule and for providing insightful and constructive comments in order to maintain the high standard of the DNA conferences. We would like to thank the editorial staff of Dagstuhl Publishing Team, and in particular Michael Wagner for his guidance and help during the process of publishing this volume. We also greatly appreciate the guidance from Damien Woods, David Soloveichik, and Josie Schaeffer during the process of organizing the program. Last but not least, we are grateful to the Organizing Committee members Matt Patitz and Jenna Stacey.

Dominic Scalise and Robert Schweller

Organization

Steering Committee

Anne Condon	University of British Columbia, Canada
Natasha Jonoska	University of South Florida, USA
Matthew Lakin	University of New Mexico, USA
Satoshi Murata	Tohoku University, Japan
Thomas Ouldridge	Imperial College London, UK
John H. Reif	Duke University, USA
Grzegorz Rozenberg	University of Leiden, The Netherlands
Rebecca Schulman	Johns Hopkins University, USA
Friedrich Simmel	Technical University Munich, Germany
David Soloveichik	University of Texas at Austin, USA
Shelley Wickham	The University of Sydney, Australia
Erik Winfree	California Institute of Technology, USA
Damien Woods (Chair)	Maynooth University, Ireland



Program Committee

Mathew Aquilina	Harvard Wyss, Dana-Farber Cancer Institute
Wooli Bae	University of Surrey
Damien Baigl	ENS Paris
Florent Becker	Université d'Orléans
Cameron Chalk	California Institute of Technology
Ho-Lin Chen	National Taiwan University
Anne Condon	University of British Columbia
David Doty	University of California, Davis
Hannah Earley	University of Cambridge
Abeer Eshra	Maynooth University
Constantine Evans	Evans Foundation and Maynooth University
Giuditta Franco	Universita di Verona
Jinglin Fu	Rutgers University - Camden
Cody Geary	Aarhus University
Leopold Green	Purdue University
Ibuki Kawamata	Kyoto University
Yonggang Ke	Georgia Institute of Technology
Chenxiang Lin	Yale University
Dongsheng Liu	Hong Kong Polytechnic University
Austin Luchsinger	University of Texas - Rio Grande Valley
Tracy Mallette	University of Washington
Chengde Mao	Purdue University
Pekka Orponen	Aalto University
Tom Ouldrige	Imperial College London
Ayush Pandey	UC Irvine
Matthew Patitz	University of Arkansas
Yann Ponty	LIX, Ecole Polytechnique
Erik Poppleton	Heidelberg University
Felix Rizzuto	University of New South Wales
Trent Rogers	University of Arkansas
Lorenzo Rovigatti	Sapienza University of Rome
Sarah Sandler	Harvard Wyss
Dominic Scalise	Washington State University
Josie Schaeffer	Google, Inc
Robert Schweller	University of Texas, Rio Grande Valley
Shinnosuke Seki	University of Electro-Communications
David Soloveichik	University of Texas, Austin
Rebecca Taylor	Carnegie Mellon University
Chris Thachuk	University of Washington
Guillaume Theyssier	AIX Marseille
Shelley Wickham	University of Sydney
Erik Winfree	California Institute of Technology
Sungwook Woo	Pohang University of Science and Technology
Damien Woods	Maynooth University
Fei Zhang	Rutgers University

Additional Reviewers for Tracks A and B

Nathanael Aubert-Kato	Ochanomizu University
Jonathan Berengut	University of New South Wales
Olivier Bournez	École Polytechnique (LIX)
Baoquan Ding	National Center for Nanoscience and Technology (China)
Gwendal Ducloz	ENS Lyon
Daniel Hader	University of Arkansas
Dawn Nye	Grinnell College
Josh Petrack	Maynooth University
Hugh Potter	Vaire Computing
Yanting Wang	Institute of Theoretical Physics, Chinese Academy of Sciences
Diming Wei	Tsinghua University
Cai Wood	Maynooth University
Yang Yang	Shanghai Jiao Tong University
Tao Zhang	Yantai University
Chao Zhou	Suzhou Institute of Nano-Tech and Nano-Bionics (SINANO)

Organizing Committee for DNA 32

Matt Patitz	University of Arkansas (OC Co-Chair)
Trent Rogers	University of Arkansas (OC Co-Chair)
Jenna Stacey	University of Arkansas

Sponsors

International Society for Nanoscale Science (ISNSCE)

National Science Foundation (NSF)

AtlasBase

University of Arkansas College of Engineering

■ List of Authors

Hamidreza Akef  (8)

Department of Electrical and Computer Engineering, The University of Texas at Austin, TX, USA

Florent Becker  (2)

Laboratoire d'Informatique Fondamentale d'Orléans (UR4022), Université d'Orléans and INSA Centre-Val-de-Loire, France

Salvador Buse  (13)

California Institute of Technology, Pasadena, CA, USA

Angel Cervera Roldan  (7)

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Cameron Chalk  (11, 13)

California Institute of Technology, Pasadena, CA, USA

Ho-Lin Chen  (14)

Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan

David Doty  (6)

Computer Science, University of California, Davis, CA, USA

Phillip Drake  (2)

University of Arkansas, Fayetteville, AR, USA

Yancheng Du  (13)

California Institute of Technology, Pasadena, CA, USA

Gwendal Ducloz  (4)

École Normale Supérieure de Lyon, LIP (UMR 5668, Équipe MC2), France

Mahboobeh Enayati  (7)

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Constantine G. Evans  (7)

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

David Furcy  (1)

Computer Science Department, University of Wisconsin Oshkosh, WI, USA

Minki Hhan  (12)

Korea Advanced Institute of Science and Technology, Daejeon, South Korea

Xiang Huang  (14)

Department of Computer Science, University of Illinois Springfield, IL, USA

Sota Kano  (3)

The University of Electro-Communications, Tokyo, Japan

Ravi Kini  (6)

Computer Science, University of California, Davis, CA, USA

Matthew R. Lakin  (5)

Department of Computer Science, University of New Mexico, Albuquerque, NM, USA; Department of Chemical & Biological Engineering, University of New Mexico, Albuquerque, NM, USA

Austin Luchsinger  (9)

Computer Science, University of Texas Rio Grande Valley, Edinburg, TX, USA

Archit Patil  (12)

University of Texas at Austin, TX, USA

Matthew J. Patitz  (2)

University of Arkansas, Fayetteville, AR, USA

Joshua Petrack  (7)

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Lulu Qian  (13)

California Institute of Technology, Pasadena, CA, USA

Nicolas Schabanel  (4)

CNRS, École Normale Supérieure de Lyon, LIP (UMR 5668, Équipe MC2) and IXXI, France

Shinnosuke Seki  (3)

The University of Electro-Communications, Tokyo, Japan

Ryder Smith  (2)

Hass Hall Academy, Bentonville, AR, USA

David Soloveichik  (8, 9, 10, 12)

Department of Electrical and Computer Engineering, The University of Texas at Austin, TX, USA

Scott M. Summers  (1)

Computer Science Department, University of Wisconsin Oshkosh, WI, USA

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Chia-Yu Sung  (8)
Department of Molecular Biosciences,
The University of Texas at Austin, TX, USA

Aneesh Vanguri  (8)
Department of Molecular Biosciences,
The University of Texas at Austin, TX, USA

Erik Winfree  (11, 13)
California Institute of Technology,
Pasadena, CA, USA

Damien Woods  (7)
Hamilton Institute and Department of Computer
Science, Maynooth University, Ireland

And Kaan Ata Yilmaz  (9, 10)
Electrical and Computer Engineering,
University of Texas at Austin, TX, USA

Sequential Non-Determinism in Tile Self-Assembly: A General Framework and an Application to Efficient Temperature-1 Self-Assembly of Squares

David Furcy ✉ 

Computer Science Department, University of Wisconsin Oshkosh, WI, USA

Scott M. Summers¹ ✉ 

Computer Science Department, University of Wisconsin Oshkosh, WI, USA

Abstract

In this paper, we work in a 2D version of the probabilistic variant of Winfree’s abstract Tile Assembly Model defined by Chandran, Gopalkrishnan and Reif (SICOMP 2012) in which attaching tiles are sampled uniformly with replacement. First, we develop a framework called “sequential non-determinism” for analyzing the probabilistic correctness of a non-deterministic, temperature-1 tile assembly system (TAS) in which most (but not all) tile attachments are deterministic and the non-deterministic attachments always occur in a specific order. Our main sequential non-determinism result equates the probabilistic correctness of such a TAS to a finite product of probabilities, each of which (1) corresponds to the probability of the correct type of tile attaching at a point where it is possible for two different types to attach, and (2) ignores all other tile attachments that do not affect the non-deterministic attachment. We then show that sequential non-determinism allows for efficient and geometrically expressive self-assembly. To that end, we constructively prove that for any positive integer N and any real $\delta \in (0, 1)$, there exists a TAS that self-assembles into an $N \times N$ square with probability at least $1 - \delta$ using only $O(\log N + \log \frac{1}{\delta})$ types of tiles. Our bound improves upon the previous state-of-the-art bound for this problem by Cook, Fu and Schweller (SODA 2011).

2012 ACM Subject Classification Theory of computation

Keywords and phrases Self-assembly, tile assembly model, temperature 1, high probability self-assembly

Digital Object Identifier 10.4230/LIPIcs.DNA.32.1

Funding *Scott M. Summers:* This author’s research was supported in part by University of Wisconsin Oshkosh Research Sabbatical (S581) during Fall 2023.

Acknowledgements We thank Matthew Patitz for providing helpful insights regarding the implementation of several aspects of our construction. We also thank Dave Doty for providing initial guidance on probabilistic self-assembly. Finally, we are grateful to the anonymous reviewers whose detailed comments helped improve the clarity and technical accuracy of the paper.

1 Introduction

The notion of “self-assembly” is generally said to involve seemingly simple, fundamental components, evolving through instances of local interaction, to a terminal assembly whose complexity is greater than the sum of its parts. Self-assembly is ubiquitous in nature, e.g., atoms self-assemble into molecules, molecules self-assemble into macromolecules, etc.

While self-assembly is ubiquitous throughout the natural world, researchers have recently begun investigating the extent to which the power of self-assembly can be harnessed for the systematic creation of atomically-precise computational, biomedical and mechanical

¹ Corresponding author



© David Furcy and Scott M. Summers;

licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 1; pp. 1:1–1:18

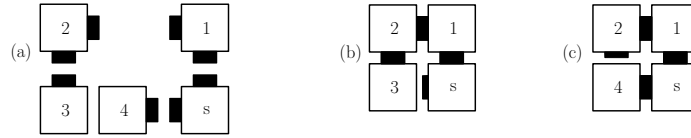
Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

devices at the nano-scale. For example, in the early 1980s, Nadrian Seeman introduced a revolutionary experimental technique for controlling nano-scale self-assembly known as “DNA tile self-assembly” [28] thus initiating the scientific study of algorithmic self-assembly. Erik Winfree’s abstract Tile Assembly Model (aTAM) [30] is a discrete mathematical model of DNA tile self-assembly, which we describe briefly in the next paragraph.

In the aTAM, a DNA tile is represented as a non-rotatable unit square *tile type*, each side of which has a corresponding *glue* comprising a string *label* and a positive integer *strength*. A *tile* is an instance of a tile type that is *placed* at some integer point in the 2D Cartesian coordinate space. The aTAM restricts the number of tile types in a tile set to be finite but infinitely many tiles of the same type may be placed. If two tiles are placed next to each other and their opposing glues match, then the tiles *bind* with the strength of the glue, thus creating a tile *assembly*, i.e., a mapping of integer points to tile types, of size 2. Before the process of self-assembly begins, a positive integer *temperature* value is chosen, which is usually 1 or 2 and a fixed *seed* tile is placed at a designated integer point defining the initial seed-containing assembly. The tile set, along with the associated temperature and placement of the seed tile make up a *tile assembly system* (TAS). Self-assembly in the aTAM is modeled by an *assembly sequence* of the TAS, which is a series of *tile attachment steps* such that in each step, the assembly sequence *attaches* a tile if the tile can be placed at an unoccupied integer point and bind to one or more tiles of the seed-containing assembly with total strength at least the temperature. If more than one tile may attach in a single step, then one tile is chosen non-deterministically to attach. Each tile attachment step in an assembly sequence *results* in a new assembly one tile larger than the assembly in the previous step, and we say the resulting assembly is *produced* by the TAS. The steps of an assembly sequence continue until a *terminal assembly* is reached to which no further tiles can bind. Figure 1 depicts an example of a *non-deterministic*, temperature-1 TAS $\mathcal{T}_1$ in which two different terminal assemblies are produced.



■ **Figure 1** Example of a TAS $\mathcal{T}_1$ with the tile set shown in (a), in which the temperature is 1 and s is the seed tile. Sub-figures (b) and (c) show assembly sequences of $\mathcal{T}_1$ that result in two different terminal assemblies, with α_{1c} denoting the terminal assembly in sub-figure (c).

Observe that temperature-1 self-assembly, which involves a TAS whose temperature is set to 1, does not possess a synchronization mechanism that can be used to enforce cooperative binding, i.e., a tile attaching at a point only after at least two of its neighboring tiles attach. However, temperature-2 self-assembly provides a mechanism to enforce cooperative binding, which can be leveraged to exhibit 1) optimal self-assembly of squares [1, 27] and algorithmically-specified finite shapes [29], 2) efficient self-assembly of rectangles [4], and 3) even an intrinsically universal tile set, i.e., a tile set that can, for an arbitrary input TAS, be configured to carry out a natural simulation of the input TAS but at a higher scale factor [8]. With the lack of a cooperative binding mechanism in temperature-1 self-assembly, it would seem natural to speculate that the computational and geometric expressiveness of temperature-1 self-assembly is fundamentally limited.

Such speculation was formalized back in 2008 by Doty, Patitz and Summers [10] through their pumpability conjecture in which they claimed that the only kinds of patterns that result from temperature-1 self-assembly in which a unique terminal assembly is produced effectively

correspond to regular languages. Even before the pumpability conjecture, temperature-1 self-assembly had been shown by Rothemund and Winfree [27] to hinder the efficient self-assembly of shapes when tile assemblies are required to be fully connected, i.e., if two tiles are neighbors, then they bind to each other. Manuch, Stacho and Stoll [21] generalized this result to temperature-1 self-assembly that always results in tile assemblies that merely contain no glue mismatches. Subsequent results showing that temperature-1 self-assembly is neither intrinsically universal [22, 25] nor capable of bounded Turing computation [25] continued to elucidate the supposed weakness of not being able to prevent non-cooperative binding. A major breakthrough by Meunier, Regnault and Woods [24] established a general pumping lemma for temperature-1 self-assembly. Although it did not directly affirm the pumpability conjecture, Meunier and Regnault [23] used it in their proof of the pumpability conjecture.

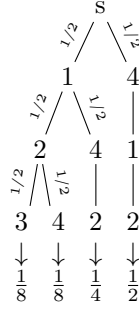
Yet the lack of enforceable cooperative binding is not solely to blame for the weakness of temperature-1 self-assembly. In fact, generalizations of the aTAM such as [6, 9, 11, 12, 18] enjoy a striking resemblance of computational and geometric expressiveness to that of temperature-2 self-assembly. This is also the case when the aTAM is only mildly generalized, e.g., by allowing: 1) the use of a unique negative glue [26], 2) duple tiles [19], 3) the attachment of cubic tiles in at most one additional plane [2, 5, 13–17], or 4) non-determinism [5] in which multiple terminal assemblies may be produced.

The results in this paper are motivated by the problem of computing the probability that a particular terminal assembly is produced by non-deterministic, temperature-1 self-assembly. To study this problem for a given non-deterministic, temperature-1 TAS $\mathcal{T}$, we use the Markov chain $\mathcal{M}_{\mathcal{T}}$, in which the states are the assembly sequences of $\mathcal{T}$ and there is a transition in $\mathcal{M}_{\mathcal{T}}$ from one assembly sequence to another with a positive probability equal to one over the number of distinct assemblies that can result from attaching a single tile to the result of the former. This is the *tile concentration programming* aTAM subject to the restriction that all tile types have the same concentration (see [7, 20] for a more general and thorough development of the tile concentration programming model). Note that this is also the same as the probabilistic aTAM (PTAM), which was defined by Chandran, Gopalkrishnan, and Reif [3], but extended to 2D, which we will refer to as the 2DPTAM going forward. It is easy to see that $\mathcal{M}_{\mathcal{T}}$ is a rooted tree. Then, the *probability* that a given TAS $\mathcal{T}$ produces a terminal assembly is the probability of $\mathcal{M}_{\mathcal{T}}$ reaching any state that corresponds to an assembly sequence of $\mathcal{T}$ that results in the terminal assembly. For example, the probability that the TAS shown in Figure 1 produces α_{1c} can be easily computed as $\frac{7}{8}$ and the corresponding $\mathcal{M}_{\mathcal{T}_1}$ is shown in Figure 2. In that and all subsequent figures, all edge label values equal to 1 are omitted (but implied). While $\mathcal{M}_{\mathcal{T}}$ can be equivalently formulated in terms of a directed acyclic graph (e.g., [5]), we use a tree to aid in the visualization of some of our results.

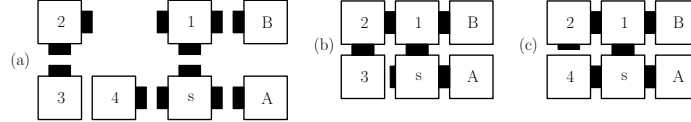
Computing the probability that $\mathcal{T}_1$ from Figure 1 produces one of its terminal assemblies is easy. However, consider the TAS shown in Figure 3, which we will denote as $\mathcal{T}_3$, and is the same as $\mathcal{T}_1$ from Figure 1 but with the addition of the A and B tile types, along with corresponding east-facing glues to the 1 and s tiles.

The presence of the A and B tiles in $\mathcal{T}_3$ does not affect, but rather complicates the computation of, the probability that $\mathcal{T}_3$ produces one of its terminal assemblies. To see this, Figure 4 shows $\mathcal{M}_{\mathcal{T}_3}$, from which the probability that $\mathcal{T}_3$ produces the terminal assembly α_{3c} can be derived.

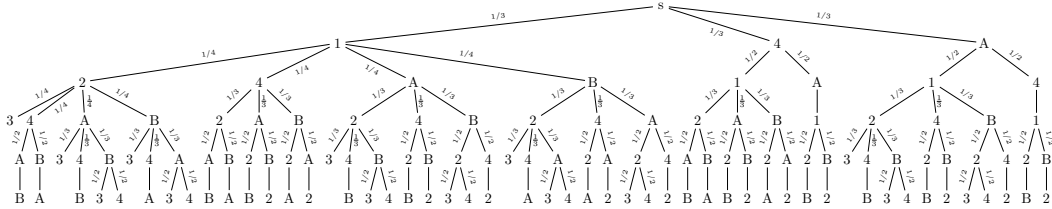
Based on $\mathcal{M}_{\mathcal{T}_3}$ shown in Figure 4, it is easy (although tedious) to see that the probability that $\mathcal{T}_3$ produces α_{3c} is: $\frac{1}{3} \frac{1}{4} \frac{1}{4} \frac{1}{2} + \frac{1}{3} \frac{1}{4} \frac{1}{4} \frac{1}{2} + \cdots + \frac{1}{3} \frac{1}{2} \frac{1}{2} = \frac{7}{8}$. Perhaps unsurprisingly, the above probability is equal to the probability of $\mathcal{T}_1$ producing α_{1c} . After all, the A and B tiles should



■ **Figure 2** $\mathcal{M}_{\mathcal{T}_1}$ corresponding to the TAS in Figure 1. Each path from the root to a leaf node corresponds to an assembly sequence of $\mathcal{T}_1$. For the sake of simplification, the label of each node is merely the type of the last tile to attach in the assembly sequence to which that node corresponds, rather than the full assembly sequence, which can easily be reconstructed by following (and then reversing) the path back up to the root. Each leaf node corresponds to an assembly sequence that results in a terminal assembly. The number under each leaf node is the probability of the corresponding sequence being produced by $\mathcal{T}_1$.



■ **Figure 3** Example of another TAS $\mathcal{T}_3$ with the tile set shown in (a), in which the temperature is 1 and s is the seed tile. Sub-figures (b) and (c) show assembly sequences of $\mathcal{T}_3$ that result in two different terminal assemblies, with α_{3c} denoting the terminal assembly in sub-figure (c).



■ **Figure 4** Markov chain $\mathcal{M}_{\mathcal{T}_3}$, where $\mathcal{T}_3$ is the TAS from Figure 3. Note that this tree is not quite complete because it only contains the assembly sequences that result in α_{3c} . All of the sub-trees whose root is labeled with a 3 and that do not grow down to the lowest level of the tree have been pruned below their root node (e.g., the leftmost node in the figure).

not affect the probability of $\mathcal{T}_3$ producing α_{3c} . So $\mathcal{T}_3$ should be equivalent to $\mathcal{T}_1$ from Figure 1 in a probabilistic sense. Nevertheless, the point of this example is to show how analyzing the probabilistic correctness of even a seemingly simple non-deterministic, temperature-1 TAS in the 2DPTAM can become unwieldy very quickly. This is because the probability that the TAS produces some terminal assembly depends on all the tile attachment steps of all the assembly sequences of the TAS that result in the assembly, including tile attachment steps in which tiles attach in such a way that should not affect the probability.

In this paper, we formalize the intuition that tiles that attach in such a way that should not affect the probability of a certain kind of non-deterministic, temperature-1 TAS in fact do not affect the probability. To that end, we develop a first-of-its-kind general framework called “sequential non-determinism” for analyzing the probabilistic correctness of such a

TAS as $\mathcal{T}_3$. In sequential non-determinism, we assume that for a given non-deterministic, temperature-1 TAS and in every step of every assembly sequence thereof, a tile initially attaches via exactly one of its glues and is the only type of tile that can attach at that location and via that glue. We also assume there exist (potentially infinitely many) *points of competition* (POC), at which one of only two types of tiles may (and may only) be placed via two corresponding competing paths of tiles that originate from a common starting point and are disjoint until potentially meeting at the POC, e.g., the location at which tiles of type 3 or 4 in Figure 3 may attach. Without going into all the technical details, we require that the POCs be ordered, sufficiently spaced out, and that, in every assembly sequence of the TAS, tiles attach at the POCs in the specified order. We assume that each POC is either *essential* or *inessential*, that the number of essential ones is finite, and that each essential POC is associated with a type of tile that is the correct type of tile that should be placed at that POC. Under these assumptions, the TAS produces a unique *correct* terminal assembly (as well as arbitrarily many incorrect ones). Our main sequential non-determinism theorem says that the probability in the 2DPTAM that a given sequentially non-deterministic, temperature-1 TAS produces the unique correct (potentially infinite) terminal assembly is a finite product of probabilities, each of which corresponds to an essential POC having the correct type of tile placed at it. Moreover, our framework allows for the computation of this probability to be carried out while effectively ignoring the tile attachments that should not affect whether the correct type of tile attaches at an essential POC. For example, our main theorem allows us to analyze $\mathcal{T}_3$ as if it were $\mathcal{T}_1$.

We then show that the requirements of sequential non-determinism are not overly restrictive, since sequential non-determinism does not hinder sufficiently geometrically and computationally expressive self-assembly. To that end, we constructively prove that for any real $\delta \in (0, 1)$ and any $N \in \mathbb{Z}^+$, there exists a TAS that produces the correct terminal assembly with probability at least $1 - \delta$, whose domain is the $N \times N$ square $\{0, 1, \dots, N - 1\}^2$ and whose tile set has a size that is $O(\log N + \log \frac{1}{\delta})$. Note that, for a fixed δ , our bound is simply $O(\log N)$, which is an improvement over the previous state-of-the-art bound of $O(\log^2 N + \log N \log \frac{1}{\delta})$ for this problem by Cook, Fu and Schweller [5].

In addition to the previous bound by Cook, Fu and Schweller, there are some other notable previous results related to probabilistic self-assembly in the aTAM. Working in the general tile concentration programming aTAM, where the concentration of each tile type can be set accordingly to affect the behavior of the TAS, Kao and Schweller [20] showed that for some fixed TAS and any positive integer N , it is possible to set the tile concentrations of the tile set of the TAS so that it produces, with high probability, a terminal assembly whose domain corresponds to an “approximation” of an $N \times N$ square. Technically, they showed that there exists a TAS such that for any $N \in \mathbb{Z}^+$ and for any real $\delta, \varepsilon > 0$, the tile concentrations can be set so that the TAS produces with probability $1 - \delta$ a terminal assembly whose domain is $\{0, 1, \dots, N' - 1\}^2$ such that $(1 - \varepsilon)N \leq N' \leq (1 + \varepsilon)N$. Among several other results, Doty [7] proved an “exact” analog of the aforementioned “approximate” result by demonstrating the existence of a TAS with constant size tile set such that, for any $N \in \mathbb{Z}^+$ and for any real $\delta > 0$, the concentrations of the tile types of the TAS can be set so that it produces with probability $1 - \delta$ a terminal assembly whose domain is $\{0, 1, \dots, N - 1\}^2$. Chandran, Gopalkrishnan, and Reif [3] investigated the size of the smallest tile set such that there is a corresponding singly-seeded, temperature-1, non-deterministic TAS that produces a terminal assembly whose domain is expected to be a *linear assembly*, i.e., an assembly whose domain, for some $N \in \mathbb{Z}^+$, is $\{0, 1, \dots, N - 1\} \times \{0\}$. While the previously-mentioned results by Doty, and Kao and Schweller utilize arbitrarily-defined tile concentrations, in

several of their results, Chandran, Gopalkrishnan and Reif worked in the PTAM (in 1D) and assumed the same uniform tile concentration programming model that we do in this paper. Among numerous other results, they proved that for any $N \in \mathbb{Z}^+$, there exists a TAS, whose tile set has size $O(\log N)$, that produces a linear terminal assembly with expected but perhaps widely varying length N . Chandran, Gopalkrishnan and Reif followed this result up by showing that there is a TAS whose tile set has size $O(\log^3 N)$ that does the same thing but can more precisely control the variance of the length of the produced terminal assembly. In this paper, we extend the PTAM to the 2DPTAM and use it to self-assemble squares rather than lines.

2 Preliminaries

In this section, we give definitions for the abstract Tile Assembly Model (aTAM) as well as probabilistic self-assembly in the aTAM, yielding a model that we call the 2DPTAM.

2.1 Formal description of the abstract Tile Assembly Model

Fix an alphabet Σ . Σ^* is the set of finite strings over Σ . Let $\mathbb{Z}$, $\mathbb{Z}^+$, and $\mathbb{N}$ denote the set of integers, positive integers, and nonnegative integers, respectively. Given $V \subseteq \mathbb{Z}^2$, the *full grid graph* of V is the undirected graph $G_V^f = (V, E)$, and for all $\vec{x}, \vec{y} \in V$, $\{\vec{x}, \vec{y}\} \in E \iff \|\vec{x} - \vec{y}\| = 1$.

A *tile type* is a tuple $t \in (\Sigma^* \times \mathbb{N})^4$, that is, a unit square with four sides listed in some standardized order, and each side having a *glue* $g \in \Sigma^* \times \mathbb{N}$ consisting of a finite string *label* and nonnegative integer *strength*. We assume a finite set of tile types, usually denoted T , but an infinite number of copies of each tile type. Intuitively, a *tile* is a copy of a tile type that is placed at a point but technically it is a pair $(\vec{x}, t) \in \mathbb{Z}^2 \times T$.

A *configuration* is a possibly empty arrangement of tile types on the integer lattice $\mathbb{Z}^2$, i.e., a partial function $\alpha : \mathbb{Z}^2 \dashrightarrow T$, or, equivalently, a subset of $\mathbb{Z}^2 \times T$. We say that the tile $(\vec{x}, t)$ is *placed* by α if $\alpha(\vec{x}) = t$. Two adjacent tiles in a configuration *bind* if the glues on their opposing sides are equal (in both label and strength) and have positive strength. Each configuration α induces a *binding graph* G_α^b , a grid graph whose vertices are positions occupied by tile types, according to α , with an edge between two vertices if the tile types at those vertices bind.

An *assembly* is a connected, non-empty configuration, i.e., a partial function $\alpha : \mathbb{Z}^2 \dashrightarrow T$ such that $G_{\text{dom } \alpha}^f$ is connected and $\text{dom } \alpha \neq \emptyset$. The *shape* $S_\alpha \subseteq \mathbb{Z}^2$ of α is $\text{dom } \alpha$ and its size is $|\text{dom } \alpha|$.

Given $\tau \in \mathbb{Z}^+$, α is τ -*stable* if every cut of G_α^b has weight at least τ , where the weight of an edge is the strength of the glue it represents. When τ is clear from the context, we say α is *stable*. Given two assemblies α and β , we say that α and β *agree* if, for all $\vec{x} \in \text{dom } \alpha \cap \text{dom } \beta$, $\alpha(\vec{x}) = \beta(\vec{x})$. Furthermore, we say that α is a *subassembly* of β , and we write $\alpha \sqsubseteq \beta$, if $S_\alpha \subseteq S_\beta$ and α and β agree.

A *tile assembly system* (TAS) is a triple $\mathcal{T} = (T, \sigma, \tau)$, where T is a tile set, $\sigma : \mathbb{Z}^2 \dashrightarrow T$ is the finite, τ -stable, *seed assembly*, and $\tau \in \mathbb{Z}^+$ is the *temperature*. We say that $\mathcal{T}$ is *singly-seeded* if $|\text{dom } \sigma| = 1$. From this point on, unless explicitly stated otherwise, we assume that every TAS is singly-seeded, with $\text{dom } \sigma = \{\vec{s}\}$, and has temperature 1.

Given two τ -stable assemblies α, β , we write $\alpha \rightarrow_{\mathcal{T}} \beta$ if $\alpha \sqsubseteq \beta$ and $|S_\beta \setminus S_\alpha| = 1$. In this case we say α *$\mathcal{T}$ -produces β in one step*. If $\alpha \rightarrow_{\mathcal{T}} \beta$, $S_\beta \setminus S_\alpha = \{\vec{p}\}$, and $t = \beta(\vec{p})$, we write $\beta = \alpha + (\vec{p}, t)$ to denote the *tile attachment step* in which a tile of type t *attaches* to α at $\vec{p}$.

to produce β . We use the notation $\beta \setminus \alpha$ to denote $(\vec{p}, t)$, i.e., the tile that attaches to α to produce β in one step. If $\beta \setminus \alpha$ attaches to α with total strength 1 and the tile in α to which $\beta \setminus \alpha$ binds is located at point $\vec{p}'$, we use $\vec{u}_{\beta \setminus \alpha}$ to denote the unit vector $\vec{p}' - \vec{p}$.

The $\mathcal{T}$ -frontier of α is the set $\partial^{\mathcal{T}}\alpha = \bigcup_{\alpha \rightarrow_1^{\mathcal{T}} \beta} S_\beta \setminus S_\alpha$, the set of empty locations at which a tile could stably attach to α . When $\mathcal{T}$ is clear from the context, we omit the superscript $\mathcal{T}$ on the $\mathcal{T}$ -frontier of α . The t -frontier $\partial_t\alpha \subseteq \partial\alpha$ of α is the set $\{ \vec{p} \in \partial\alpha \mid \alpha \rightarrow_1^{\mathcal{T}} \beta \text{ and } \beta(\vec{p}) = t \}$.

Let $\mathcal{A}^T$ denote the set of all assemblies of tiles from T , and let $\mathcal{A}_{<\infty}^T$ denote the set of finite assemblies of tiles from T . A sequence of $k \in \mathbb{Z}^+ \cup \{\infty\}$ assemblies $\alpha_1, \alpha_2, \dots$ over $\mathcal{A}^T$ denoted as $\vec{\alpha} = (\alpha_i \mid 0 \leq i-1 < k)$ is a $\mathcal{T}$ -assembly sequence if, for all $1 \leq i-1 < k$, $\alpha_{i-1} \rightarrow_1^{\mathcal{T}} \alpha_i$. We use this notation for defining the range of values for i because it allows us to handle both cases for k (i.e., finite or infinite) simultaneously. For integers $0 \leq i-1 \leq j-1 < k$, $\vec{\alpha}[i \dots j]$ denotes the subsequence $(\alpha_i, \dots, \alpha_j)$. When $\vec{\alpha}$ is finite, we say that $\vec{\alpha}$ *terminates* at $\vec{p} \in \mathbb{Z}^2$, where $\{\vec{p}\} = \text{dom } \alpha_1$ if $k = 1$, and $\{\vec{p}\} = \text{dom } \alpha_k \setminus \text{dom } \alpha_{k-1}$ otherwise. The *result* of an assembly sequence $\vec{\alpha}$, written as $\text{res}(\vec{\alpha})$ is the unique assembly such that $\text{dom } \text{res}(\vec{\alpha}) = \bigcup_{i=1}^k \text{dom } \alpha_i$ and for all $0 \leq i-1 < k$, $\alpha_i \sqsubseteq \text{res}(\vec{\alpha})$. In other words, $\text{res}(\vec{\alpha})$ is the unique limiting assembly, which for a finite sequence, is the final assembly in the sequence. If $\text{res}(\vec{\alpha}) = \alpha$ and $\vec{x} \in \text{dom } \alpha$, then $\text{index}_{\vec{\alpha}}(\vec{x}) = \min \{ i \mid \vec{x} \in \text{dom } \alpha_i \}$.

If $\alpha = \text{res}(\vec{\alpha})$ is finite and, for some $t \in T$, $\vec{x} \in \partial_t\alpha$, then the notation $\vec{\alpha} + (\vec{x}, t)$ denotes *appending the tile $(\vec{x}, t)$ to $\vec{\alpha}$* , and is the assembly sequence $(\alpha_1, \dots, \alpha_k, \alpha_{k+1})$, where $\alpha_{k+1} = \alpha_k + (\vec{x}, t)$.

We write $\alpha \rightarrow^{\mathcal{T}} \beta$, and we say α $\mathcal{T}$ -produces β (in 0 or more steps) if there is a $\mathcal{T}$ -assembly sequence $(\alpha_1, \alpha_2, \dots)$ of length $k = |S_\beta \setminus S_\alpha| + 1$ such that (1) $\alpha = \alpha_1$, (2) $S_\beta = \bigcup_{0 \leq i-1 < k} S_{\alpha_i}$, and (3) for all $0 \leq i-1 < k$, $\alpha_i \sqsubseteq \beta$. If k is finite then it is routine to verify that $\beta = \alpha_k$. We say α is $\mathcal{T}$ -producible if $\sigma \rightarrow^{\mathcal{T}} \alpha$, and we write $\mathcal{A}[\mathcal{T}]$ to denote the set of $\mathcal{T}$ -producible assemblies. We say that $\vec{\alpha}$ is $\mathcal{T}$ -producing if $\vec{\alpha} = (\sigma, \dots)$ and $\text{res}(\vec{\alpha}) \in \mathcal{A}[\mathcal{T}]$. An assembly α is $\mathcal{T}$ -terminal if α is τ -stable and $\partial^{\mathcal{T}}\alpha = \emptyset$. We write $\mathcal{A}_{\square}[\mathcal{T}] \subseteq \mathcal{A}[\mathcal{T}]$ to denote the set of $\mathcal{T}$ -producible, $\mathcal{T}$ -terminal assemblies.

2.2 The two-dimensional probabilistic aTAM or 2DPTAM

We model probabilistic self-assembly in the aTAM with non-empty, rooted, directed, weighted tree structures in which all edges point away from the root and, for each node v , S_v denotes the sum of all the numerical weights over all the outgoing edges on v . If v is a leaf node, then $S_v = 0$. We say that a non-empty, rooted, weighted tree $\mathcal{Q}$ is a *sub-probability tree* (SPT) if:

1. all edges of $\mathcal{Q}$ are labeled with some *probability* $p \in (0, 1]$, and
2. for each node v in $\mathcal{Q}$, $S_v \leq 1$.

If $\mathcal{Q}$ is an SPT and v is a node of $\mathcal{Q}$, then we say v is *normalized* in $\mathcal{Q}$ if $S_v = 1$. If all internal nodes of an SPT $\mathcal{Q}$ are normalized, then $\mathcal{Q}$ is a discrete time Markov chain, or simply Markov chain. For an SPT $\mathcal{Q}$, if v is a node of $\mathcal{Q}$, then we define the *probability* of v , denoted as $\text{Pr}_{\mathcal{Q}}[v]$, as follows:

1. If v is the root node of $\mathcal{Q}$, then $\text{Pr}_{\mathcal{Q}}[v] = 1$.
2. Otherwise, if u is the parent of v and p is the probability of the edge from u to v , then $\text{Pr}_{\mathcal{Q}}[v] = \text{Pr}_{\mathcal{Q}}[u] \cdot p$.

Let $m \in \mathbb{Z}^+ \cup \{\infty\}$, $\pi = (q_i \mid 0 \leq i-1 < m)$ be a sequence of nodes in $\mathcal{Q}$ where q_1 is the root of $\mathcal{Q}$, for all $1 \leq i-1 < m$, there is an edge from q_{i-1} to q_i in $\mathcal{Q}$, and p_i is the probability on this edge. We say that π is a *maximal path* in $\mathcal{Q}$ if $m = \infty$, or $m \in \mathbb{Z}^+$ and q_m is a leaf node of $\mathcal{Q}$. If $m = 1$, then $\text{Pr}_{\mathcal{Q}}[\pi] = 1$. Otherwise, if $m > 1$, then we define

$\Pr_{\mathcal{Q}}[\pi] = \prod_{\{i \in \mathbb{Z}^+ \mid 1 \leq i-1 < m\}} p_i$. Note that if $m = \infty$, then π is a ray in $\mathcal{Q}$, starting from the root. We define the *probability of SPT* $\mathcal{Q}$ to be $\Pr[\mathcal{Q}] = \sum_{\pi \text{ maximal path in } \mathcal{Q}} \Pr_{\mathcal{Q}}[\pi]$.

If $\mathcal{Q}$ is a rooted tree and u is a node in $\mathcal{Q}$, then $\mathcal{Q}^u$ is the subtree of $\mathcal{Q}$ that contains u , along with all descendants of u in $\mathcal{Q}$. Note that, if $\mathcal{Q}$ is an SPT, then $\mathcal{Q}^u$ is also an SPT. We say that any subtree $\mathcal{Q}'$ of $\mathcal{Q}$ is *full relative to* $\mathcal{Q}$ if, for every internal node v of $\mathcal{Q}'$, all children of v in $\mathcal{Q}$ are children of v in $\mathcal{Q}'$. We say $\mathcal{Q}$ is *finitely branching* (or *locally finite*) if, for every internal node v of $\mathcal{Q}$, there exists $c \in \mathbb{Z}^+$ such that v has c children in $\mathcal{Q}$.

Assume $\pi = (x_i \mid 0 \leq i-1 < m)$ for $m \in \mathbb{Z}^+ \cup \{\infty\}$ denotes a simple path whose elements may be, for example, either points in $\mathbb{Z}^2$ or nodes in an SPT. Then $\text{dom } \pi$ denotes the set of elements $\{x_i \mid i \geq 1\}$, $|\pi| = m$ denotes the length of π , $\pi[i]$ denotes x_i for $i \geq 1$, and, for integers $0 \leq i-1 \leq j-1 < m$, $\pi[i \dots j]$ denotes the subsequence $(x_i, \dots, x_j)$. Any subsequence $\pi[1 \dots j]$ for integers $j \geq 1$ is called a *prefix* of π , whereas, assuming π is a finite path, any subsequence $\pi[i \dots m]$ for integers $1 \leq i \leq m$ is called a *suffix* of π .

We study probabilistic self-assembly in the aTAM using a simplified version of the tile concentration programming model, which we call the 2DPTAM, where tiles are assumed to have uniform concentration (see [7] for a formal development of the general model). We model probabilistic self-assembly of a TAS $\mathcal{T} = (T, \sigma, 1)$ with the SPT $\mathcal{M}_{\mathcal{T}}$, where:

1. the root node of $\mathcal{M}_{\mathcal{T}}$ is the $\mathcal{T}$ -assembly sequence (σ) ,
2. there is a one-to-one correspondence between nodes in $\mathcal{M}_{\mathcal{T}}$ and finite $\mathcal{T}$ -producing assembly sequences,
3. there is an edge in $\mathcal{M}_{\mathcal{T}}$ from $\vec{\alpha}$ to $\vec{\beta}$, assuming $\text{res}(\vec{\alpha}) = \alpha$ and $\text{res}(\vec{\beta}) = \beta$, if $\alpha \rightarrow_1^{\mathcal{T}} \beta$, and
4. the probability of the edge in $\mathcal{M}_{\mathcal{T}}$ from $\vec{\alpha}$ to $\vec{\beta}$, assuming $\text{res}(\vec{\alpha}) = \alpha$ and $\text{res}(\vec{\beta}) = \beta$, is $\frac{1}{M_{\alpha}}$, where $M_{\alpha} = |\{\gamma \mid \alpha \rightarrow_1^{\mathcal{T}} \gamma\}| > 0$, since $\alpha \rightarrow_1^{\mathcal{T}} \beta$.

Note that $\mathcal{M}_{\mathcal{T}}$ is a finitely branching SPT.

We define the probability of a $\mathcal{T}$ -producible assembly in terms of the $\mathcal{T}$ -producing assembly sequences from which it may result. Since there is a one-to-one correspondence between nodes of $\mathcal{M}_{\mathcal{T}}$ and finite $\mathcal{T}$ -producing assembly sequences, for $\alpha \in \mathcal{A}[\mathcal{T}]$ we define the *probability of* α as: $\Pr_{\mathcal{T}}[\alpha] = \sum_{\substack{\vec{\alpha} \text{ is a } \mathcal{T}\text{-producing assembly sequence} \\ \text{such that } \text{res}(\vec{\alpha}) = \alpha}} \Pr_{\mathcal{M}_{\mathcal{T}}}[\vec{\alpha}]$.

We say that $\mathcal{T}$ *strictly self-assembles a shape* $X \subseteq \mathbb{Z}^2$ with probability at least $p \in [0, 1]$ if $\sum_{\substack{\alpha \in \mathcal{A}[\mathcal{T}] \text{ such} \\ \text{that } \text{dom } \alpha = X}} \Pr_{\mathcal{T}}[\alpha] \geq p$. We say that X *strictly self-assembles* with probability at least p if there exists some TAS that strictly self-assembles it with probability at least p .

3 Defining sequential non-determinism

In this section, we give a formulation of sequential non-determinism.

Going forward, assume $\mathcal{T} = (T, \sigma, 1)$ is a singly-seeded TAS with $\text{dom } \sigma = \{\vec{s}\}$.

Note that we use boxes like the one above to state assumptions that hold by default, unless otherwise stated. This practice allows us to significantly shorten the statement of several lemmas that share an increasingly long sequence of assumptions.

► **Definition 1.** Let $\vec{x}, \vec{y} \in \mathbb{Z}^2$ with $\vec{x} \neq \vec{y}$, and π and π' be finite simple paths in $G_{\mathbb{Z}^2}^f$ from $\vec{x}$ to $\vec{y}$. We say that π and π' are competing for $\vec{y}$, from $\vec{x}$ in $\mathcal{T}$, if all of the following conditions hold.

1. $\pi \neq \pi'$
2. $\text{dom } \pi \cap \text{dom } \pi' = \{\vec{x}, \vec{y}\}$
3. There exist a path $p \in \{\pi, \pi'\}$ and an assembly $\alpha \in \mathcal{A}[\mathcal{T}]$ such that p is a simple path in G_α^b .
4. For all $\alpha \in \mathcal{A}[\mathcal{T}]$, each path $p \in \{\pi, \pi'\}$ and all integers $1 \leq l < |p|$, if p' is any finite simple path from $\vec{s}$ to $p[l]$ in G_α^b , then:
 - a. if p' goes through $\vec{x} = p[1]$, then $p[1 \dots l]$ is a suffix of p' ,
 - b. otherwise p' is a prefix of every finite simple path from $\vec{s}$ to $p[l]$ in G_α^b .
5. For all $\alpha \in \mathcal{A}[\mathcal{T}]$, if p' is any simple path from $\vec{s}$ to $\vec{y}$ in G_α^b , then there exists a path $p \in \{\pi, \pi'\}$ such that p is a suffix of p' .
6. For all $\alpha \in \mathcal{A}[\mathcal{T}]$ such that $\vec{x} \in \text{dom } \alpha$, the following conditions hold.
 - a. For each path $p \in \{\pi, \pi'\}$, there exists a $\mathcal{T}$ -assembly sequence $\vec{\beta} = (\beta_i \mid 1 \leq i \leq k)$ such that $\beta_1 = \{(\vec{x}, \alpha(\vec{x}))\}$ and $\text{dom } \beta_k = \text{dom } p$.
 - b. For all $\mathcal{T}$ -assembly sequences $\vec{\beta} = (\beta_i \mid 1 \leq i \leq k)$, if $\beta_1 = \{(\vec{x}, \alpha(\vec{x}))\}$ and p is a simple path from $\vec{x}$ to $\vec{y}$ in $G_{\beta_k}^b$, then $p \in \{\pi, \pi'\}$.

We call p a competing path if there exist $\vec{x}, \vec{y} \in \mathbb{Z}^2$ and paths π and π' that are competing for $\vec{y}$ from $\vec{x}$ such that $p \in \{\pi, \pi'\}$.

Conditions 1 and 2 together mean that the two competing paths only share their starting and ending points and that their domains are not identical. Condition 3 means that there exists a $\mathcal{T}$ -producing assembly sequence that results in an assembly that places tiles on all the points in π or π' . Condition 4a means that, if the binding graph G of a $\mathcal{T}$ -producible assembly contains a simple path p from $\vec{s}$ to any non-ending point on one of the competing paths such that p goes through $\vec{x}$, then p must first go through $\vec{x}$ and follow π or π' . Condition 4b means that, if the binding graph G of a $\mathcal{T}$ -producible assembly contains a simple path p from $\vec{s}$ to any non-ending point on one of the competing paths such that p does not go through $\vec{x}$, then p must be a prefix of all paths from $\vec{s}$ to $\vec{x}$ in G . Condition 5 means that all $\mathcal{T}$ -assembly sequences that follow a path from $\vec{s}$ to $\vec{y}$ must go through $\vec{x}$ and then follow either π or π' . Condition 6a means that it is possible for either competing path to self-assemble in isolation in a $\mathcal{T}$ -assembly sequence starting from any tile that is placed at $\vec{x}$ by some $\mathcal{T}$ -producible assembly. Condition 6b means that, if there is a $\mathcal{T}$ -assembly sequence that starts from any tile that is placed at $\vec{x}$ by some $\mathcal{T}$ -producible assembly and follows a simple path to $\vec{y}$, then this simple path must be one of the two competing paths π or π' .

► **Definition 2.** Let $\vec{y} \in \mathbb{Z}^2$. We call the point $\vec{y}$ a point of competition, or POC for short, in $\mathcal{T}$, if there exist $\vec{x} \in \mathbb{Z}^2$ and simple paths π and π' in $G_{\mathbb{Z}^2}^f$ that are competing for $\vec{y}$, from $\vec{x}$ in $\mathcal{T}$. We say that π and π' are the competing paths that correspond to $\vec{y}$, and $\vec{x}$ is the corresponding starting point.

Going forward, let $r \in \mathbb{Z}^+$ and assume $\mathcal{T}$ has a non-empty, possibly infinite set P of POCs with $|P| \geq r$.

► **Definition 3.** Let $Y = \{\vec{y}_1, \dots, \vec{y}_r\}$ be an r -element subset of P . A winner function (for $\mathcal{T}$) is a function of the form $w : Y \rightarrow T$. Having defined w , we call the set Y the set of essential POCs in $\mathcal{T}$.

Going forward, assume $Y = \{\vec{y}_1, \dots, \vec{y}_r\}$ is the r -element set of essential POCs in $\mathcal{T}$ and $S_Y = \{\vec{x}_1, \dots, \vec{x}_r\}$ is the corresponding set of starting points.

For example, a TAS could be designed to strictly self-assemble an $N \times N$ square with high probability such that its set of essential POCs is a particular subset of $\{0, \dots, N-1\}^2$. However, this TAS may induce infinite, erroneous $\mathcal{T}$ -assembly sequences that attach tiles at infinitely many POCs. The motivation behind Definition 3 is that $\mathcal{T}$ may produce multiple (possibly infinite) terminal assemblies and we use w to determine the correctness of such an assembly in the following sense.

► **Definition 4.** Let $\alpha \in \mathcal{A}[\mathcal{T}]$ and $w : Y \rightarrow T$ be a winner function for $\mathcal{T}$. If:

1. for all $\vec{p} \in \text{dom } \alpha \cap Y$, $\alpha(\vec{p}) = w(\vec{p})$ and
2. $\text{dom } \alpha \cap (P \setminus Y) = \emptyset$,

then we say that α is a w -correct $\mathcal{T}$ -assembly. If $\vec{\alpha}$ is any $\mathcal{T}$ -assembly sequence such that $\text{res}(\vec{\alpha}) = \alpha$, then we say that $\vec{\alpha}$ is a w -correct $\mathcal{T}$ -assembly sequence.

Condition 1 of this definition means that α may only place correct tiles at the essential POCs included in its domain. Condition 2 means that a tile cannot attach to an inessential POC unless it is attaching to a $\mathcal{T}$ -producible assembly that is not w -correct.

► **Definition 5.** Let $\vec{p} \in \mathbb{Z}^2$, $\alpha, \alpha', \beta, \beta' \in \mathcal{A}[\mathcal{T}]$ and $t_\alpha, t_\beta \in T$ such that $\alpha' = \alpha + (\vec{p}, t_\alpha)$ and $\beta' = \beta + (\vec{p}, t_\beta)$ are valid tile attachment steps. We say that $\vec{p}$ is a directionally deterministic point in $\mathcal{T}$ if either $\vec{p} = \vec{s}$, or the following two conditions hold:

1. If $\vec{p} \in P$, then $t_\alpha \neq t_\beta$ implies at least one of the following conditions hold:
 - a. $\vec{u}_{\alpha' \setminus \alpha} \neq \vec{u}_{\beta' \setminus \beta}$ or
 - b. $\alpha(\vec{p} + \vec{u}_{\alpha' \setminus \alpha}) \neq \beta(\vec{p} + \vec{u}_{\beta' \setminus \beta})$.
2. If $\vec{p} \in \mathbb{Z}^2 \setminus P$, then $t_\alpha \neq t_\beta$ implies that α and β do not agree.

Condition 2 of this definition means that, when two $\mathcal{T}$ -producible assemblies agree, no more than one tile may attach at any non-POC that belongs to the intersection of their frontiers. Condition 1 says that, if any two distinct tiles ever attach to any $\mathcal{T}$ -producible assemblies at any POC $\vec{p}$, then these tiles must attach via glues on different sides or bind to different tile types. The TAS shown in Figure 1 produces two terminal assemblies, respectively placing the 3 tile at the point $\vec{p}$ in the bottom-left corner, as depicted in Figure 1(b), or placing the 4 tile at the same point, as depicted in Figure 1(c). Note that, in this example, the 3 tile binds only via its north glue, whereas the 4 tile binds only via its east glue. Therefore, the point $\vec{p}$ is directionally deterministic in this TAS.

► **Definition 6.** We say that $\mathcal{T}$ is a directionally deterministic TAS if all points in the domains of all $\mathcal{T}$ -producible assemblies are directionally deterministic.

The next lemma says that if $\mathcal{T}$ is directionally deterministic and α is a w -correct $\mathcal{T}$ -terminal assembly, then any w -correct $\mathcal{T}$ -producing assembly sequence must result in a subassembly of α .

► **Lemma 7.** Let $w : Y \rightarrow T$ be a winner function for $\mathcal{T}$ and $\alpha \in \mathcal{A}_\square[\mathcal{T}]$ be w -correct. If $\mathcal{T}$ is directionally deterministic and β is any w -correct, $\mathcal{T}$ -producible assembly, then $\beta \sqsubseteq \alpha$.

► **Corollary 8.** If $\mathcal{T}$ is directionally deterministic and there exists a w -correct assembly $\alpha \in \mathcal{A}_\square[\mathcal{T}]$, then α is unique.

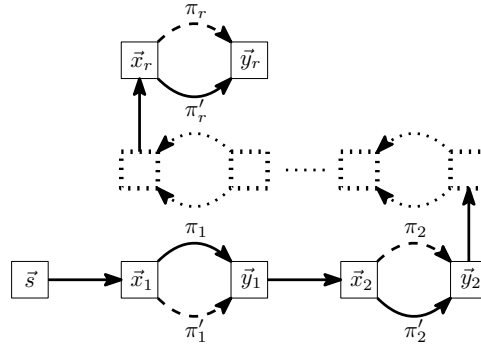
Going forward, assume $w : Y \rightarrow T$ is a winner function for $\mathcal{T}$.

The following definition is the cornerstone of the framework we introduce in this paper. For the sake of completeness, we explicitly state all of the assumptions that were instantiated in previous “Going forward” boxes.

► **Definition 9.** Assume $\mathcal{T} = (T, \sigma, 1)$ is a singly-seeded TAS with $\text{dom } \sigma = \{\vec{s}\}$, $r \in \mathbb{Z}^+$, $\mathcal{T}$ has a non-empty, possibly infinite set P of POCs with $|P| \geq r$, S_P denotes the set $\{\vec{x} \mid \vec{x} \text{ is the starting point for some } \vec{y} \in P\}$, $Y = \{\vec{y}_1, \dots, \vec{y}_r\}$ is the r -element set of essential POCs in $\mathcal{T}$ with corresponding set of starting points $S_Y = \{\vec{x}_1, \dots, \vec{x}_r\}$, and $w : Y \rightarrow T$ is a winner function for $\mathcal{T}$ such that α is the unique w -correct $\mathcal{T}$ -terminal assembly with $Y \subseteq \text{dom } \alpha$. We say that $\mathcal{T}$ is w -sequentially non-deterministic if all of the following conditions hold:

1. $\mathcal{T}$ is directionally deterministic,
2. for all $\mathcal{T}$ -producing assembly sequences $\vec{\alpha}$, if $Y \subseteq \text{dom } \text{res}(\vec{\alpha})$, then for all integers $1 \leq i < r$, $\text{index}_{\vec{\alpha}}(\vec{y}_i) < \text{index}_{\vec{\alpha}}(\vec{y}_{i+1})$, and
3. $S_P \cap P = \emptyset$.

Figure 5 depicts visually and explains informally the assumptions and implications of Definition 9.



■ **Figure 5** In this example, $\mathcal{T} = (T, \sigma, 1)$ is a w -sequentially non-deterministic TAS with $\text{dom } \sigma = \{\vec{s}\}$, $r > 2$, and α is the unique w -correct, $\mathcal{T}$ -terminal assembly whose domain contains all of the essential POCs. Note that each arrow represents a sequence of tile attachment steps that follow a simple path. Definition 9 either requires or implies that:

- a) [Condition 1] every point at which a tile attaches in the assembly sequence is directionally deterministic,
- b) [Assumption in Definition 9] the $\vec{y}_i$ points are all the essential POCs in $\mathcal{T}$,
- c) [Condition 2] in all $\mathcal{T}$ -assembly sequences whose results place tiles at every $\vec{y}_i$, the tiles always attach at the $\vec{y}_i$ points in order of increasing i values,
- d) [Implied] in all $\mathcal{T}$ -producing assembly sequences whose results place tiles at every $\vec{y}_i$, tiles also attach at all of the $\vec{x}_i$ points, and tile attachment steps always occur in the order $\vec{x}_1, \vec{y}_1, \vec{x}_2, \vec{y}_2, \dots, \vec{x}_r, \vec{y}_r$, and
- e) [Condition 3] none of the $\vec{x}_i$ points is a point of competition.

The solid arrows in the figure depict a $\mathcal{T}$ -producible assembly that is the result of a $\mathcal{T}$ -producing assembly sequence that attaches tiles at points along a simple path in $\mathbb{Z}^2$ and in the following, required order: from $\vec{s}$ (the seed's point) to $\vec{x}_1$, then to $\vec{y}_1$ via π_1 , then to $\vec{x}_2$, then to $\vec{y}_2$ via π_2 , ..., to $\vec{x}_r$, and finally to $\vec{y}_r$ via π_r .

Going forward, assume $\mathcal{T}$ is w -sequentially non-deterministic, with corresponding unique w -correct $\mathcal{T}$ -terminal assembly α .

In the following, we define the TAS that corresponds to a pair of paths competing for a POC.

► **Definition 10.** For all integers $1 \leq i \leq r$, let $\mathcal{C}_i = (T, \sigma_i, 1)$ be the TAS with $\sigma_i = \{(\vec{x}_i, \alpha(\vec{x}_i))\}$. We say that $\mathcal{C}_i$ is competition i in $\mathcal{T}$ associated with POC $\vec{y}_i$.

Note that Definition 10 defines each competition $\mathcal{C}$ as the TAS $\mathcal{T}$ with a different seed tile. This allows us to reason about the behavior of $\mathcal{C}$ in the absence of any other tiles, such as, e.g., a simple path of tiles from the seed of $\mathcal{T}$ to the starting point of $\mathcal{C}$ that may or may not block one of the competing paths of $\mathcal{C}$. Importantly, Definition 10 does not restrict the set of $\mathcal{C}$ -producible assemblies to only those that place no tiles outside the domains of the associated competing paths. $\mathcal{T}$. However, in the following definition, we will characterize a subset of “winning” $\mathcal{C}$ -producible assemblies that place tiles on every point in the winning path and possibly a prefix of the losing path. We will later define (see Definition 14) the probability of $\mathcal{C}$ only considering $\mathcal{C}$ -producing assembly sequences that not only result in a “winning” $\mathcal{C}$ -producible assembly but are also comprised of tile attachment steps that faithfully follow the competing paths.

► **Definition 11.** Let $\vec{y} \in Y$ with corresponding competing paths π and π' , $\mathcal{C}$ be the competition in $\mathcal{T}$ associated with $\vec{y}$, and $\gamma \in \mathcal{A}[\mathcal{C}]$ such that:

1. π is a path in G_γ^b ,
2. $\gamma \sqsubseteq \alpha$, and
3. every simple path in G_γ^b that starts at $\vec{x}$ is a prefix of either π or π' .

We say that π is the winning path for $\mathcal{C}$, π' is the losing path for $\mathcal{C}$, and that γ is a winning assembly for $\mathcal{C}$. Let $W_\mathcal{C}$ denote the set of all winning assemblies for $\mathcal{C}$. Any $\mathcal{C}$ -producing assembly sequence $\vec{\gamma}$ that terminates at $\vec{y}$ and whose result is in $W_\mathcal{C}$ is a winning assembly sequence for $\mathcal{C}$.

Going forward, assume $\vec{y} \in Y$ with corresponding starting point $\vec{x} \in S_Y$, $\vec{\beta}_\vec{x}$ is a $\mathcal{T}$ -producing sequence that terminates at $\vec{x}$ and whose result $\beta_\vec{x}$ is a subassembly of α , and $\mathcal{C}$ is the competition in $\mathcal{T}$ associated with $\vec{y}$ with corresponding winning and losing paths π_{win} and π_{lose} , respectively.

The following definition describes the notion that a competition need not be “fair” relative to a starting assembly.

► **Definition 12.** We say that $\mathcal{C}$ is rigged by $\vec{\beta}_\vec{x}$ if there exists an integer $1 < l < |\pi_{\text{lose}}|$ such that:

1. $\pi_{\text{lose}}[l] \in \text{dom } \beta_\vec{x}$ and
2. every simple path from $\vec{s}$ to $\vec{x}$ in $G_{\beta_\vec{x}}^b$ goes through $\pi_{\text{lose}}[l]$.

► **Definition 13.** We say that $\mathcal{C}$ is rigged in $\mathcal{T}$ if it is rigged by every $\mathcal{T}$ -producing sequence that terminates at $\vec{x}$ and whose result is a subassembly of α .

We now define the probability of a competition as the sum of the probabilities of all of its winning assembly sequences. The probability of a winning assembly sequence of the competition is defined relative to a modified version of its corresponding SPT where certain nodes are removed to simplify the computation of the probability of the winning assembly sequence.

► **Definition 14.** Let $\vec{y} \in Y$ with corresponding starting point $\vec{x}$ and competing paths π_{win} and π_{lose} . Let $\mathcal{C}$ be the competition in $\mathcal{T}$ associated with $\vec{y}$ and $\mathcal{M}'_{\mathcal{C}}$ be the SPT obtained from $\mathcal{M}_{\mathcal{C}}$ by removing all nodes $\vec{\beta}$ of $\mathcal{M}_{\mathcal{C}}$ such that the binding graph of $\text{res}(\vec{\beta})$ contains a simple path that starts at $\vec{x}$ and is not a prefix of either π_{win} or π_{lose} . If $\vec{\gamma}$ is any winning assembly sequence for $\mathcal{C}$, we define the probability of $\vec{\gamma}$ as $Pr_{\mathcal{C}}[\vec{\gamma}] = Pr_{\mathcal{M}'_{\mathcal{C}}}[\vec{\gamma}]$, and the probability of competition $\mathcal{C}$ as $Pr[\mathcal{C}] = \sum_{\vec{\gamma} \text{ winning assembly sequence for } \mathcal{C}} Pr_{\mathcal{C}}[\vec{\gamma}]$.

Intuitively, $Pr[\mathcal{C}]$ is defined so that it can be computed easily, i.e., without having to account for potentially numerous points at which tiles could attach outside of the competing paths. The next lemma first counts the number of winning assembly sequences for a competition that is not rigged that all result in a certain winning assembly and then derives the probability of each such winning assembly sequence.

► **Lemma 15.** Let $\vec{y} \in Y$ with corresponding competing paths π_{win} and π_{lose} of length $l \geq 2$ and $l' \geq 2$, respectively. Let $\mathcal{C}$ be the competition associated with $\vec{y}$ in $\mathcal{T}$. For any integer $0 \leq i \leq l' - 2$, let π_i denote the (possibly empty) simple path $\pi_{lose}[2 \dots (i+1)]$ and γ denote the winning assembly in $W_{\mathcal{C}}$ whose domain equals $\text{dom } \pi_{win} \cup \text{dom } \pi_i$. If $\mathcal{C}$ is not rigged in $\mathcal{T}$, then:

1. The number of winning assembly sequences for $\mathcal{C}$ with result γ is $\binom{l+i-2}{l-2}$.
2. If $\vec{\gamma}$ is a winning assembly sequence for $\mathcal{C}$, with result γ , then $Pr_{\mathcal{C}}[\vec{\gamma}] = \frac{1}{2^{|\vec{\gamma}|-1}} = \frac{1}{2^{l+i-1}}$.

► **Corollary 16.** $Pr[\mathcal{C}] = \sum_{i=0}^{l'-2} \binom{l+i-2}{l-2} \cdot \left(\frac{1}{2}\right)^{l+i-1}$.

The following is the first main result of this paper, which states that the probability that the w -sequentially non-deterministic TAS $\mathcal{T}$ strictly self-assembles the shape of the unique α is at least the product of the probabilities of all of its competitions, which themselves can be easily computed using our framework.

► **Theorem 17.** If $\mathcal{C}_1, \dots, \mathcal{C}_r$ are the competitions in $\mathcal{T}$, then $\mathcal{T}$ strictly self-assembles $\text{dom } \alpha$ with probability at least $\prod_{i=1}^r Pr[\mathcal{C}_i]$.

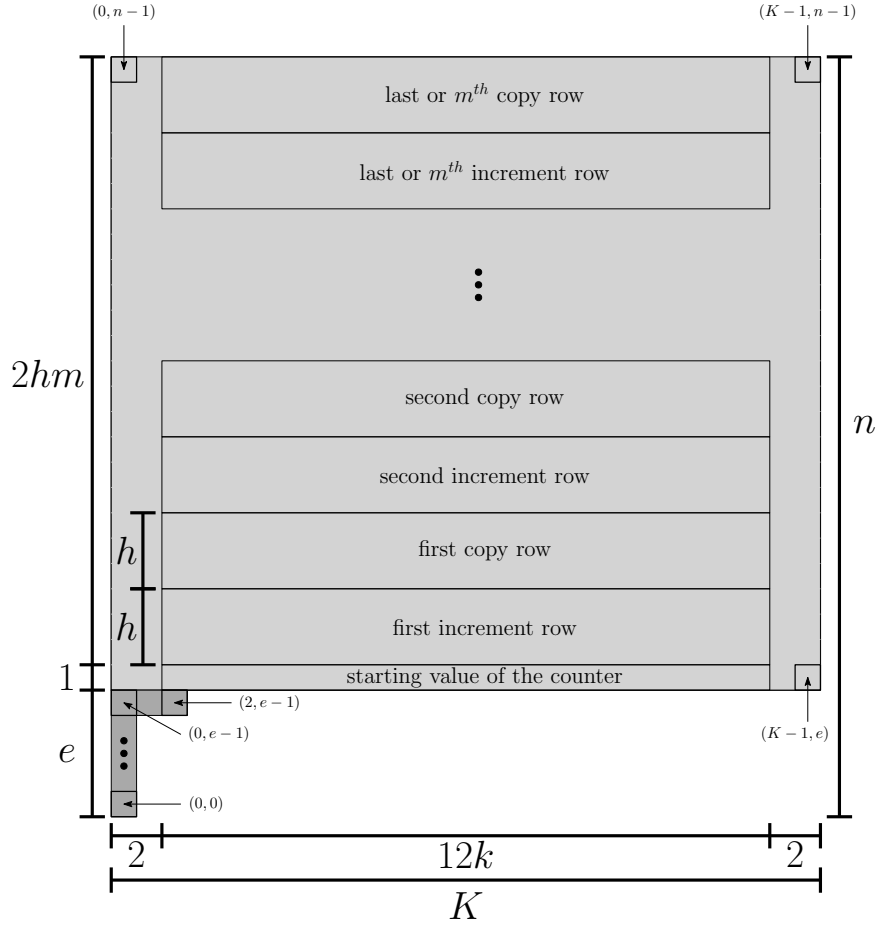
4 Efficient, sequentially non-deterministic self-assembly of an $N \times N$ square with high probability

In this section, we state and sketch the proof of our main “constructive theorem” namely that, for every $N \in \mathbb{Z}^+$ and real $\delta > 0$, there exists a TAS that strictly self-assembles an $N \times N$ square with probability at least $1 - \delta$ and uses only $O(\log N + \log \frac{1}{\delta})$ tile types.

Our construction uses three modified copies of a single template for a zigzag binary counter. This counter template is K tiles wide and n tiles high, with $n \leq N$. It is built out of “gadgets”, which are groups of tiles that self-assemble within a fixed region of space to accomplish a specific task, e.g., non-deterministically read (i.e., guess) a single unknown, geometrically-represented binary value. Since our counter assembles in a zigzag pattern, each one of its values is first incremented, going from right to left, and then copied from left to right. Since one row of tiles is used to hardcode the starting value of the counter and all of the copy and increment gadgets are h tiles high, with $h = \lceil 2 \log N + \log \frac{1}{\delta} \rceil + 5$, the number of values taken on by the counter is equal to $m = \lfloor \frac{n-1}{2h} \rfloor$. Then $e = (n-1) \bmod 2h$ is the number of extra tiles (i.e., rows) needed for the height of our counter template to be exactly

equal to n . The counter template is P-shaped. The stem of the P is an upside-down L whose domain we denote by $L_e = \{0\} \times \{0, \dots, e-1\} \cup \{(1, e-1), (2, e-1)\}$ and depict in dark gray in Figure 6. The “loop” of the P is a $K \times (n-e)$ rectangular counter whose domain we denote by $R_{K,n,e} = \{0, \dots, K-1\} \times \{e, \dots, n-1\}$ and depict in light gray in Figure 6.

The last parameters we need pertain to the binary representation of the counter’s values. We use k to denote the number of bits in the binary expansion of m . Each bit value will be geometrically encoded in a 6-tile-wide gadget corresponding to a competition. Since two tiles are added on each side of the counter’s value and each bit is actually encoded by two gadgets, namely the bit’s value itself and an additional indicator bit used to distinguish the leftmost and rightmost value bits from the other ones, K is equal to $12k + 4$. Finally, the starting value of our counter is set to $s = 2^{\lfloor \log m \rfloor + 1} - m$.



■ **Figure 6** Shape of, and parameters for, our counter template.

To prove the main constructive theorem of this section, we will use the following lemma:

► **Lemma 18.** *Let $N \in \mathbb{Z}^+$, $\delta \in \mathbb{R}^+$ such that $0 < \delta < 1$, $h = \lceil 2 \log N + \log \frac{1}{\delta} \rceil + 5$, $n \in \mathbb{Z}^+$ such that $n \leq N$, and $m = \lfloor \frac{n-1}{2h} \rfloor$. Then there exists a TAS $\mathcal{T} = (T, \sigma, 1)$ satisfying the following conditions:*

1. *If $m > 0$, $s = 2^{\lfloor \log m \rfloor + 1} - m$, k is the number of bits in the binary expansion of m , $K = 12k + 4$, and $e = (n-1) \bmod 2h$, then there exists a finite set $P \subseteq \{0, \dots, K-1\} \times \{0, \dots, n-1\}$, a finite set $Y \subseteq P$ and $w : Y \rightarrow T$ such that:*

- a. P is the set of POCs in $\mathcal{T}$, Y is the set of essential POCs in $\mathcal{T}$ and $Y = P$,
 - b. $\mathcal{T}$ is w -sequentially non-deterministic,
 - c. α is the unique w -correct $\mathcal{T}$ -terminal assembly and $\text{dom } \alpha = L_e \cup R_{K,n,e}$,
 - d. in every w -correct $\mathcal{T}$ -producing assembly sequence resulting in α , no tile placed at any point on the perimeter of α has a positive-strength glue on its outward-facing side, the last one among α 's perimeter points at which a tile is placed is $(K-1, n-1)$, and this point is the only one in $\text{dom } \alpha$ where any tile of this type is placed.
 - e. for every competition $\mathcal{C}$ of $\mathcal{T}$, if π and π' are the corresponding winning and losing paths, respectively, then $|\pi| = 4$ and $|\pi'| = 4h - 10$ (or vice versa),
2. $\mathcal{T}$ strictly self-assembles $\text{dom } \alpha = L_e \cup R_{K,n,e}$ with probability at least $1 - \delta$, and
 3. $|T| = O(\log N + \log \frac{1}{\delta})$.

The reason we set h so high is to allow the P-shaped assembly produced by the TAS in Lemma 18 to be used in a larger construction, e.g., one that strictly self-assembles an $N \times N$ square, which will require a higher value for h in order to guarantee a sufficient probability of correctness. We chose the P-shape of $\text{dom } \alpha$ to be able to invoke Lemma 18 multiple times when assembling an $N \times N$ square.

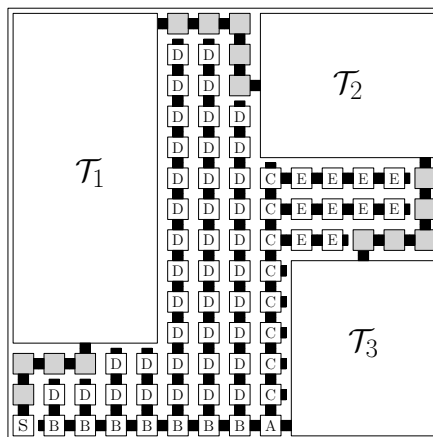
Proof sketch. To prove Lemma 18, we proceed as follows:

1. Prove that the lemma holds in the corner cases when n is too small, i.e., $n < 13$, or when $m = 0$, i.e., when h is too large for a given value of even a sufficiently large n . Assume going forward that $n \geq 13$ and $m > 0$.
2. Define T in terms of “gadgets” of tiles and compute an asymptotic upper bound on its cardinality. The tile types for T will simulate a zigzag binary counter that is initialized with the starting value s , and then alternates between increment steps in which the counter value is incremented by 1 going from right to left, followed by copy steps in which the counter value is copied going from left to right. The counter stops after a copy step in which the copied value is all 0s.
 - a. Create one seed gadget of size $O(e)$, which contains the unique seed tile of $\mathcal{T}$.
 - b. Create $O(k)$ gadgets, each of size $O(1)$, to initialize the counter.
 - c. Create $O(1)$ gadgets, each of size $O(h)$, to increment the counter.
 - d. Create $O(1)$ gadgets, each of size $O(h)$, to copy the value of the counter.
 - e. Consolidate the outcomes of the four previous gadget creation steps to infer that $|T| = O(\log N + \log \frac{1}{\delta})$.
3. Define P , Y and w .
4. Prove that $\mathcal{T}$ is w -sequentially non-deterministic such that α is the unique w -correct $\mathcal{T}$ -terminal assembly and $\text{dom } \alpha = L_e \cup R_{K,n,e}$.
5. Prove that every w -correct $\mathcal{T}$ -producing assembly sequence resulting in α attaches the last tile at the upper-right corner of α and that no tile on the perimeter of α has an outward-facing positive-strength glue.
6. Prove that $\mathcal{T}$ strictly self-assembles $\text{dom } \alpha$ with probability at least $1 - \delta$.
 - a. Use Corollary 16 to prove that, for every competition $\mathcal{C}$ of $\mathcal{T}$, $\Pr[\mathcal{C}] \geq 1 - \frac{\delta}{N^2}$.
 - b. Apply Theorem 17 and use Bernoulli's inequality to bound the product of the probabilities of the $2km \leq N^2$ essential competitions of $\mathcal{T}$. ◀

Lemma 18 can be used to prove the following theorem:

► **Theorem 19.** For each $N \in \mathbb{Z}^+$ and real δ such that $0 < \delta < 1$, there exists a TAS $\mathcal{T}_{N,\delta} = (T_{N,\delta}, \sigma_{N,\delta}, 1)$ that strictly self-assembles $\{0, \dots, N-1\}^2$ with probability at least $1 - \delta$ and such that $|T_{N,\delta}| = O(\log N + \log \frac{1}{\delta})$.

The construction of $\mathcal{T}_{N,\delta}$ follows from three invocations of Lemma 18, a high-level depiction of which is shown in Figure 7.



■ **Figure 7** Construction of $\mathcal{T}_{N,\delta}$ via three invocations of Lemma 18 resulting in $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_3$, respectively. The latter two are rotated clockwise by 90 and 180 degrees, respectively, and then translated appropriately. The ‘S’ tile in the lower left corner denotes $\sigma_{N,\delta}$. The new tile types A, B, C, D and E suffice to fill in the interior region of the square.

5 Conclusion

In this paper, we introduced the notion of sequential non-determinism, which is a general framework for analyzing the probabilistic correctness of a TAS in the PTAM that has a unique correct terminal assembly and an arbitrary number of (potentially infinite) incorrect assemblies. We then demonstrated that it is a sufficiently general framework to allow the efficient self-assembly of an $N \times N$ square at temperature 1 with high probability, i.e., Theorem 19. Note that Theorem 19 improves upon the previous state-of-the-art bound by Cook, Fu and Schweller [5], but our result is still not optimal from an information-theoretic perspective. It is future work to determine whether such an asymptotic lower bound can be achieved in the 2DPTAM.

References

- 1 Leonard M. Adleman, Qi Cheng, Ashish Goel, and Ming-Deh A. Huang. Running time and program size for self-assembled squares. In *STOC*, pages 740–748, 2001. doi:10.1145/380752.380881.
- 2 Nathaniel Bryans, Ehsan Chiniforooshan, David Doty, Lila Kari, and Shinnosuke Seki. The power of nondeterminism in self-assembly. *Theory Comput.*, 9:1–29, 2013. doi:10.4086/T0C.2013.V009A001.
- 3 Harish Chandran, Nikhil Gopalkrishnan, and John H. Reif. Tile complexity of linear assemblies. *SIAM J. Comput.*, 41(4):1051–1073, 2012. doi:10.1137/110822487.
- 4 Qi Cheng, Gagan Aggarwal, Michael H. Goldwasser, Ming-Yang Kao, Robert T. Schweller, and Pablo Moisset de Espanés. Complexities for generalized models of self-assembly. *SIAM Journal on Computing*, 34:1493–1515, 2005. doi:10.1137/S0097539704445202.
- 5 Matthew Cook, Yunhui Fu, and Robert Schweller. Temperature 1 self-assembly: Deterministic assembly in 3d and probabilistic assembly in 2d. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, 2011.

- 6 Erik D. Demaine, Martin L. Demaine, Sándor P. Fekete, Mashhood Ishaque, Eynat Rafalin, Robert T. Schweller, and Diane L. Souvaine. Staged self-assembly: nanomanufacture of arbitrary shapes with $O(1)$ glues. *Natural Computing*, 7(3):347–370, 2008. doi:10.1007/s11047-008-9073-0.
- 7 David Doty. Randomized self-assembly for exact shapes. *SICOMP: SIAM Journal on Computing*, 39(8):3521–3552, September 2010. Preliminary version appeared in FOCS 2009. doi:10.1137/090779152.
- 8 David Doty, Jack H. Lutz, Matthew J. Patitz, Scott M. Summers, and Damien Woods. Intrinsic universality in self-assembly. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science*, pages 275–286, 2009.
- 9 David Doty, Matthew J. Patitz, Dustin Reishus, Robert T. Schweller, and Scott M. Summers. Strong fault-tolerance for self-assembly with fuzzy temperature. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS 2010)*, pages 417–426, 2010. doi:10.1109/FOCS.2010.47.
- 10 David Doty, Matthew J. Patitz, and Scott M. Summers. Limitations of self-assembly at temperature one (extended abstract). In Turlough Neary, Damien Woods, Anthony Karel Seda, and Niall Murphy, editors, *Proceedings International Workshop on The Complexity of Simple Programs, CSP 2008, Cork, Ireland, 6-7th December 2008*, volume 1 of *EPTCS*, pages 67–69, 2008. arXiv:0906.3251.
- 11 Sándor P. Fekete, Jacob Hendricks, Matthew J. Patitz, Trent A. Rogers, and Robert T. Schweller. Universal computation with arbitrary polyomino tiles in non-cooperative self-assembly. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 148–167, 2015. doi:10.1137/1.9781611973730.12.
- 12 Bin Fu, Matthew J. Patitz, Robert T. Schweller, and Robert Sheline. Self-assembly with geometric tiles. In *Automata, Languages, and Programming – 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, pages 714–725, 2012. doi:10.1007/978-3-642-31594-7_60.
- 13 David Furcy, Samuel Micka, and Scott M. Summers. Optimal program-size complexity for self-assembled squares at temperature 1 in 3D. *Algorithmica*, 77(4):1240–1282, 2017. doi:10.1007/S00453-016-0147-6.
- 14 David Furcy and Scott M. Summers. Optimal self-assembly of finite shapes at temperature 1 in 3D. *Algorithmica*, 80(6):1909–1963, 2018. doi:10.1007/S00453-016-0260-6.
- 15 David Furcy, Scott M. Summers, and Hailey Vadnais. Tight bounds on the directed tile complexity of a just-barely $3d\ 2 \times N$ rectangle at temperature 1. In Daniela Genova and Jarkko Kari, editors, *Unconventional Computation and Natural Computation – 20th International Conference, UCNC 2023, Jacksonville, FL, USA, March 13-17, 2023, Proceedings*, volume 14003 of *Lecture Notes in Computer Science*, pages 79–93. Springer, 2023. doi:10.1007/978-3-031-34034-5_6.
- 16 David Furcy, Scott M. Summers, and Christian Wendlandt. Self-assembly of and optimal encoding within thin rectangles at temperature-1 in 3D. *Theoretical Computer Science*, 872:55–78, 2021. doi:10.1016/J.TCS.2021.02.001.
- 17 David Furcy, Scott M. Summers, and Logan Withers. Improved lower and upper bounds on the tile complexity of uniquely self-assembling a thin rectangle non-cooperatively in 3d. *Theory Comput. Syst.*, 67(5):1082–1130, 2023. doi:10.1007/S00224-023-10137-9.
- 18 Oscar Gilbert, Jacob Hendricks, Matthew J. Patitz, and Trent A. Rogers. Computing in continuous space with self-assembling polygonal tiles (extended abstract). In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 937–956, 2016. doi:10.1137/1.9781611974331.CH67.
- 19 Jacob Hendricks, Matthew J. Patitz, Trent A. Rogers, and Scott M. Summers. The power of duples (in self-assembly): It’s not so hip to be square. *Theoretical Computer Science*, 743:148–166, 2018. doi:10.1016/J.TCS.2015.12.008.

- 20 Ming-Yang Kao and Robert T. Schweller. Randomized self-assembly for approximate shapes. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I: Track A: Algorithms, Automata, Complexity, and Games*, volume 5125 of *Lecture Notes in Computer Science*, pages 370–384. Springer, 2008. doi:10.1007/978-3-540-70575-8_31.
- 21 Ján Manuch, Ladislav Stacho, and Christine Stoll. Two lower bounds for self-assemblies at temperature 1. *Journal of Computational Biology*, 17(6):841–852, 2010. doi:10.1089/CMB.2009.0067.
- 22 P.-E. Meunier, M. J. Patitz, S. M. Summers, G. Theyssier, A. Winslow, and D. Woods. Intrinsic universality in tile self-assembly requires cooperation. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 752–771, 2014.
- 23 Pierre-Étienne Meunier and Damien Regnault. Directed non-cooperative tile assembly is decidable. In Matthew R. Lakin and Petr Sulc, editors, *27th International Conference on DNA Computing and Molecular Programming, DNA 27, September 13-16, 2021, Oxford, UK (Virtual Conference)*, volume 205 of *LIPICs*, pages 6:1–6:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.DNA.27.6.
- 24 Pierre-Étienne Meunier, Damien Regnault, and Damien Woods. The program-size complexity of self-assembled paths. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 727–737, 2020. doi:10.1145/3357713.3384263.
- 25 Pierre-Étienne Meunier and Damien Woods. The non-cooperative tile assembly model is not intrinsically universal or capable of bounded turing machine simulation. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 328–341, 2017. doi:10.1145/3055399.3055446.
- 26 Matthew J. Patitz, Robert T. Schweller, and Scott M. Summers. Exact shapes and Turing universality at temperature 1 with a single negative glue. In *Proceedings of the 17th international conference on DNA computing and molecular programming, DNA’11*, pages 175–189, Berlin, Heidelberg, 2011. Springer-Verlag. doi:10.1007/978-3-642-23638-9_15.
- 27 Paul W. K. Rothmund and Erik Winfree. The program-size complexity of self-assembled squares (extended abstract). In *STOC ’00: Proceedings of the thirty-second annual ACM Symposium on Theory of Computing*, pages 459–468, 2000. doi:10.1145/335305.335358.
- 28 Nadrian C. Seeman. Nucleic-acid junctions and lattices. *Journal of Theoretical Biology*, 99:237–247, 1982.
- 29 David Soloveichik and Erik Winfree. Complexity of self-assembled shapes. *SIAM J. Comput.*, 36(6):1544–1569, 2007. doi:10.1137/S0097539704446712.
- 30 Erik Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, California Institute of Technology, June 1998.

Powers and Limitations of Synchronous Self-Assembly

Non-cooperative Assemblies and Limited Synchronization

Florent Becker ✉ 

Laboratoire d'Informatique Fondamentale d'Orléans (UR4022),
Université d'Orléans and INSA Centre-Val-de-Loire, France

Phillip Drake ✉ 

University of Arkansas, Fayetteville, AR, USA

Matthew J. Patitz ✉ 

University of Arkansas, Fayetteville, AR, USA

Ryder Smith ✉ 

Hass Hall Academy, Bentonville, AR, USA

Abstract

In abstract models of algorithmic self-assembly, synchronization between attachments has emerged as a crucial distinction between the classical asynchronous model (aTAM) and a new synchronous model, the syncTAM. This paper presents recent advances in gauging the additional power afforded by the syncTAM. While it is known that the syncTAM and the aTAM are each unable to fully simulate the other, this paper offers evidence that the syncTAM is computationally significantly more powerful than the aTAM, especially in the non-cooperative setting.

The additional power of the non-cooperative syncTAM is witnessed by the following constructions, all impossible in the non-cooperative aTAM: a flagpole, a strict self-assembly of a variant of the discrete Sierpinski triangle, and the ability to build the same assemblies (modulo scale factor) as directed aTAM systems.

The second topic is that of limited synchronization, wherein, when the number of attachments is smaller than some threshold l , they happen synchronously, but attachments in excess of that number must wait. In that context, the precise value of l is crucial, and changes to that value prevent simulation and can change which shapes can be obtained.

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases self-assembly, noncooperative self-assembly, models of computation, tile assembly systems

Digital Object Identifier 10.4230/LIPIcs.DNA.32.2

Related Version *Full Version*: <https://arxiv.org/abs/2605.27604> [3]

Supplementary Material *Software (Source Code)*: [https://self-assembly.net/wiki/index.php/Synchronous_Tile_Assembly_Model_\(syncTAM\)#Examples_and_Simulation_Software](https://self-assembly.net/wiki/index.php/Synchronous_Tile_Assembly_Model_(syncTAM)#Examples_and_Simulation_Software)

Funding *Florent Becker*: Research made at a public, state-funded university, thanks to the guarantee of full academic freedom and perennial research funding.

Phillip Drake: This author's work was supported in part by NSF grant 2329908

Matthew J. Patitz: This author's work was supported in part by NSF grant 2329908

1 Introduction and Outline

Crafting artifacts with intricate small-scale details has been a mark of sophistication in a wide variety of human cultures. Contemporary researchers have turned this endeavor into the field of nanotechnology, with the aim to extend the realm of controlled manipulation



© Florent Becker, Phillip Drake, Matthew J. Patitz, and Ryder Smith;
licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 2; pp. 2:1–2:23

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

down to the molecular and atomic scales. As it turns out, DNA is a very handy vehicle for that, especially as its ability to hold data can be harnessed for algorithmic control of its assembly. Thus was born the field of (DNA) Algorithmic Self-Assembly.

The principle is to come up with basic building blocks which are able to selectively bind and grow patterns of interest. Theoretical models serve to predict and engineer this binding and assembly. There is a variety of them and this paper will be dealing with *abstract* models of self-assembly, which tend to be easier to reason about while giving broad brush insights into the possibilities of molecular self-assembly.

Arguably, the first abstract model of algorithmic self-assembly was the abstract Tile Assembly Model (aTAM) [28]. Since its introduction, a wide variety of results have exhibited its powerful algorithmic potential [4, 9, 19, 25–27].

One striking feature which sets it apart from more classical models of computation is its asynchronous nature. In order to claim correctness for a system in the aTAM, one has to prove that it obeys one’s fancy whatever order the building blocks (known as tiles) come to interact with the growing assembly. On the other hand, there is a natural synchronous variant of the aTAM, called the syncTAM which works synchronously, and thus much closer to more classical models of computation such as cellular automata. The first investigations of the syncTAM in [2] have shown that the two models are uncomparable (each can do things the other cannot). In this paper, we set out to refine that distinction.

It would seem natural that a synchronous model affords its user more control, which in turn means they need fewer resources in order to implement algorithms. In particular, the *temperature* parameter (a.k.a. minimum binding threshold) is a very important source of control in both the aTAM and the syncTAM. Informally, the temperature of an aTAM system tells whether tiles can do a bit of local synchronization through cooperation when they assemble (at temperature 2 and more), or not (at temperature 1). It is well-known in the aTAM that systems with temperature 1 are less powerful than systems with temperature 2 and above. They can be distinguished by intrinsic universality, defined in [9], allowing [21] to state that there are temperature 2 systems which no temperature 1 system can simulate. Moreover, in two dimensions, temperature 1 systems are limited in their computational capacities, as they must be *fragile* or *pumpable* [22]. On the other hand, effectively implementing temperature 2 is a tough experimental challenge. For this reason, there have been many theoretical explorations of variations of the aTAM that exhibit at least some of the power of cooperative, temperature-2, self-assembly at temperature-1 [7, 10–12, 14, 18, 24]. We set out to show that the synchronicity of the syncTAM can act as a kind of substitute for temperature 2 by exploring the power of temperature 1 syncTAM systems. Namely, we show that these systems are able to simulate a flagpole, thus overcoming the simulation obstacle that was identified in [21] for the aTAM. The cooperative synchronization gadget used in this system can act as a basic brick for 2-dimensional computation, allowing the strict self-assembly of a shape close to the Sierpinski Triangle. Sierpinski Triangles have been known since [1] to belong to the class of fractals which cannot be strictly self-assembled in the aTAM [4]. This witnesses that in addition to cooperation and thus computation, the temperature 1 syncTAM is able to *detect the absence* of some part of the assembly. If synchronization was a complete substitute for cooperation, then temperature 1 systems in the syncTAM would be able to build the same assemblies (at least modulo some scale factor) as temperature 2 systems in the aTAM. We show that this is the case in the absence of non-determinism: when restricted to aTAM systems which are directed, the claim holds.

How much of that power remains if, instead of full synchronicity, one assumes that tiles come all at once, *up to some maximal amount l*? This is what we investigate next, introducing the L-syncTAM. Looking for intermediates between the syncTAM and the aTAM is a natural

way to try and make sense of more plausible models of algorithmic DNA self-assembly. The hope is that any property which is robust to changes along the continuum between the syncTAM and the aTAM will hold in experiments. The investigation of the L-syncTAM for various values of l reveals that altering that parameter either up or down affects the dynamics of the model, preventing universal simulation between different values of l , and even affecting which shapes can be assembled.

Section 2 introduces the reader to the aTAM, syncTAM and L-syncTAM models, as well as to intrinsic simulation for these models. Section 3 exposes the power of the syncTAM model by way of the flagpole and Sierpinski triangle constructions. Section 4 examines directed systems in the syncTAM, up to the “simulation” at temperature 1 of directed aTAM systems. Finally, Section 5 is the investigation of limited synchronicity through the L-syncTAM model.

2 Preliminary Definitions and Models

In this section we define the models and terminology used throughout the paper.

2.1 The abstract Tile-Assembly Model

We work within the abstract Tile-Assembly Model [28] in 2 and 3 dimensions. The abbreviation *aTAM* refers to the 2D model. These definitions are borrowed from [15] and we note that [26] and [20] are good introductions to the model for unfamiliar readers.

Let $\mathbb{N}$ be the set of nonnegative integers, for $l, u \in \mathbb{Z}$, let $\llbracket l, u \rrbracket = \{k \in \mathbb{Z} \mid l \leq k < u\}$. For $n \in \mathbb{N}$, let $[n] = \llbracket 0, n \rrbracket = \{0, 1, \dots, n-2, n-1\}$.

Fix Σ to be some alphabet with Σ^* its finite strings. A *glue* $g \in \Sigma^* \times \mathbb{N}$ consists of a finite string *label* and non-negative integer *strength*. There is a single glue of strength 0, referred to as the *null* glue. A *tile type* is a tuple $t \in (\Sigma^* \times \mathbb{N})^4$, thought of as a unit square with a glue on each side. A *tile set* is a finite set of tile types. We always assume a finite set of tile types, but allow an infinite number of copies of each tile type to occupy locations in the $\mathbb{Z}^2$ lattice, each called a *tile*. A *glue set* G_T is the set of all glues associated with a tile set T . Given a direction $d \in \{N, E, S, W\} = \mathcal{D}$ and a tile set T , $G_{T,d}$ is the set of all glues on the edges of tiles in direction d and $g_{t,d}$ is the glue g on tile t in direction d . We write $\text{str}_t(d)$ to denote the strength of $g_{t,d}$.

Given a tile set T , a *configuration* is an arrangement (possibly empty) of tiles in the lattice $\mathbb{Z}^2$, i.e. a partial function $\alpha : \mathbb{Z}^2 \dashrightarrow T$. Two adjacent tiles in a configuration *interact*, or are *bound* or *attached*, if the glues on their abutting sides are equal (in both label and strength) and have positive strength. Each configuration α induces a *binding graph* B_α whose vertices are those points occupied by tiles, with an edge of weight s between two vertices if the corresponding tiles interact with strength s . An *assembly* is a configuration whose domain (as a graph) is connected and non-empty. The *shape* of $X \subset \mathbb{Z}^2$ is $\{X + (x, y) \mid x, y \in \mathbb{Z}\}$, i.e. the set of all its translations. The shape of an assembly α is the shape S_α of its domain: two assemblies have the same shape if they have the same domain modulo translation. An assembly α contains a shape S if there is $s \in S$ such that $s \subset \text{dom } \alpha$. For some $\tau \in \mathbb{Z}^+$, an assembly α is τ -*stable* if every cut of B_α has weight at least τ , i.e. a τ -stable assembly cannot be split into two pieces without separating bound tiles whose shared glues have cumulative strength τ . Given two assemblies α, β , we say α is a *subassembly* of β (denoted $\alpha \sqsubseteq \beta$) if $\text{dom } \alpha \subseteq \text{dom } \beta$ and for all $p \in \text{dom } \alpha$, $\alpha(p) = \beta(p)$ (i.e., they have tiles of the same types in all locations of $\text{dom } \alpha$).

A *tile-assembly system* (TAS) is a triple $\mathcal{T} = (T, \sigma, \tau)$, where T is a tile set, σ is a finite τ -stable assembly called the *seed assembly*, and $\tau \in \mathbb{Z}^+$ is called the *binding threshold* (a.k.a. *temperature*). Given a TAS $\mathcal{T} = (T, \sigma, \tau)$ and two τ -stable assemblies α and β , we say that α *$\mathcal{T}$ -produces β in one step* (written $\alpha \rightarrow_1^{\mathcal{T}} \beta$) if $\alpha \sqsubseteq \beta$ and $|S_\beta \setminus S_\alpha| = 1$. That is, $\alpha \rightarrow_1^{\mathcal{T}} \beta$ if β differs from α by the addition of a single tile. The *$\mathcal{T}$ -frontier* is the set $\partial^{\mathcal{T}} \alpha = \bigcup_{\alpha \rightarrow_1^{\mathcal{T}} \beta} S_\beta \setminus S_\alpha$ of locations in which a tile could τ -stably attach to α . When $\mathcal{T}$ is clear from context we simply refer to these as the *frontier* locations.

We use $\mathcal{A}^T$ to denote the set of all assemblies of tiles in tile set T . Given a TAS $\mathcal{T} = (T, \sigma, \tau)$, a sequence of $k \in \mathbb{Z}^+ \cup \{\infty\}$ assemblies $\alpha_0, \alpha_1, \dots$ over $\mathcal{A}^T$ is called a *$\mathcal{T}$ -assembly sequence* if, for all $1 \leq i < k$, $\alpha_{i-1} \rightarrow_1^{\mathcal{T}} \alpha_i$. The *result* $\lim \alpha$ of an assembly sequence α is the unique limiting assembly of the sequence. For finite assembly sequences, this is the final assembly; whereas for infinite assembly sequences, this is the assembly consisting of all tiles from any assembly in the sequence. We say that α *$\mathcal{T}$ -produces β* (denoted $\alpha \rightarrow^{\mathcal{T}} \beta$) if there is a $\mathcal{T}$ -assembly sequence starting with α whose result is β . We say α is *$\mathcal{T}$ -producible* if $\sigma \rightarrow^{\mathcal{T}} \alpha$ and write $\mathcal{A}[\mathcal{T}]$ to denote the set of $\mathcal{T}$ -producible assemblies. We say α is *$\mathcal{T}$ -terminal* if α is τ -stable and there exists no assembly that is $\mathcal{T}$ -producible from α . We denote the set of $\mathcal{T}$ -producible and $\mathcal{T}$ -terminal assemblies by $\mathcal{A}_\square[\mathcal{T}]$. If $|\mathcal{A}_\square[\mathcal{T}]| = 1$, i.e., there is exactly one terminal assembly, we say that $\mathcal{T}$ is *directed*. When $\mathcal{T}$ is clear from context, we may omit $\mathcal{T}$ from notation.

Given TAS $\mathcal{T}$, we define $\mathcal{S}_\square[\mathcal{T}] = \{S_\alpha \mid \alpha \in \mathcal{A}_\square[\mathcal{T}]\}$ as the set of all shapes of terminal assemblies in $\mathcal{T}$. Clearly, if $\mathcal{T}$ is directed $|\mathcal{S}_\square[\mathcal{T}]| = 1$. However, even if $\mathcal{T}$ is not directed it may be the case that $|\mathcal{S}_\square[\mathcal{T}]| = 1$ (see [6] for an example). In such a case we call the system *shape directed* and say that $\mathcal{T}$ *strictly self-assembles* S_α , where $S_\alpha \in \mathcal{S}_\square[\mathcal{T}]$.

2.2 The Synchronous Tile Assembly Model

The Synchronous Tile Assembly Model (syncTAM) is the variant of the aTAM where at every step of assembly, rather than a single location of the frontier being non-deterministically selected to receive a tile, *all* frontier locations simultaneously receive a tile each. As the frontier of a system can grow arbitrarily large, the number of tiles added per assembly step may also grow arbitrarily large. As in the aTAM, if any single frontier location allows τ -strength binding of tiles of multiple types, one of those types is non-deterministically chosen for each such location.

We formally define the syncTAM in the same way as the aTAM with the exception of the definition of a single step in the assembly sequence. Given a syncTAS $\mathcal{S} = (T, \sigma, \tau)$ and two τ -stable assemblies α and β , we say that α *$\mathcal{S}$ -produces β in one step*, written $\alpha \rightarrow_1^{\mathcal{S}} \beta$, if $\alpha \sqsubseteq \beta$ and $\text{dom}(\beta) \setminus \text{dom}(\alpha) = \partial^{\mathcal{S}}(\alpha)$. Note that like in the aTAM, we define $\partial^{\mathcal{S}}(\alpha)$ to be the set of locations where a tile can τ -stably attach to α . The syncTAM inherits all other definitions and notation from the aTAM.

To distinguish between a tile assembly system (a.k.a. TAS) in the aTAM versus one in the syncTAM, we will use the term *syncTAS* to refer to a TAS in the syncTAM.

2.3 Limited Synchronicity Tile Assembly Model

The Limited Synchronicity Tile Assembly Model (L-syncTAM) is a variant of the syncTAM (introduced in [2]) wherein instead of all frontier locations simultaneously receiving a tile each step, some predefined nonzero integer number of frontier locations each receive a tile. We define this number as the *synchronicity* for a given system, and denote it as $l \in \mathbb{Z}^+$. At every assembly step, l frontier locations are filled (unless the frontier is smaller than l , in which case all frontier locations are filled).

Formally, we define the L-syncTAM similarly to the syncTAM. We call a tile assembly system $\mathcal{L}$ in the L-syncTAM an *L-syncTAS* and define it as $\mathcal{L} = (T, \sigma, \tau, l)$, where l is its synchronicity and T , σ , and τ are defined as previously. As with the syncTAM, given a τ -stable assembly α in $\mathcal{L}$, we define $\partial^{\mathcal{L}}(\alpha)$ to be the set of locations where a tile can τ -stably attach to α , and the L-syncTAM inherits all other definitions and notations from the aTAM, with the following refinement to the definition of an assembly step.

Given two τ -stable assemblies α and β in $\mathcal{L}$, we say that α *$\mathcal{L}$ -produces β in one step*, written $\alpha \rightarrow_1^{\mathcal{L}} \beta$ if $\alpha \sqsubseteq \beta$ and one of the two following are true:

1. $\text{dom}(\beta) \setminus \text{dom}(\alpha) = \partial^{\mathcal{L}}(\alpha)$ and $|\partial^{\mathcal{L}}(\alpha)| \leq l$
2. $\text{dom}(\beta) \setminus \text{dom}(\alpha) \subseteq \partial^{\mathcal{L}}(\alpha)$ and $|\text{dom}(\beta) \setminus \text{dom}(\alpha)| = l$

The first case holds when the frontier is no larger than l . In the second case, when the frontier is larger, the particular subset of frontier locations that receive tiles at a step is non-deterministically selected from the full set of frontier locations. Note that since binding locations are selected only from the current frontier, $\partial^{\mathcal{L}}(\alpha)$, where tiles can bind to the existing assembly α with strength $\geq \tau$, for a tile in location $\vec{l} \in \mathbb{Z}^2$, where $\vec{l} \notin \text{dom} \alpha$, to attach to α it must form one or more bonds with strengths totaling $\geq \tau$ with one or more tiles already in α , regardless of how many bonds it may form with other tiles attaching during the same step.

Note that the aTAM is a special case of the L-syncTAM where $l = 1$, and the syncTAM can be thought of as being a special case where l is allowed to be ∞ .

2.4 Scaled shapes

Given S , a connected set of points in $\mathbb{Z}^2$, we define a version of S *scaled by factor c* , and denote it by S^c , as $S^c = \{(x, y) \in \mathbb{Z}^2 \mid (\lfloor \frac{x}{c} \rfloor, \lfloor \frac{y}{c} \rfloor) \in S\}$. Intuitively, S^c is a version of S expanded by replacing each point of S by a $c \times c$ square of points. Likewise, for a set A of shapes, A^c is defined as $\{S^c \mid S \in A\}$.

Given two TASs $\mathcal{T}$ and $\mathcal{T}'$, if $S_{\square}[\mathcal{T}] = S_{\square}[\mathcal{T}']$, then we say that $\mathcal{T}$ *shape-matches* $\mathcal{T}'$. Similarly, if $S_{\square}[\mathcal{T}]^c = S_{\square}[\mathcal{T}']^d$ for some $c, d \in \mathbb{Z}^+$, we say that $\mathcal{T}$ *scale shape-matches* $\mathcal{T}'$.

2.5 Synchronous intrinsic simulation

Intrinsic universality via intrinsic simulation has been studied in a variety of tile-assembly models (e.g. [8, 9, 15–17, 23]). Essentially, intrinsic simulation deals with one system simulating the behavior of another modulo scale factor, with squares of tiles in the simulating system representing individual tiles of the simulated system. Intrinsic universality (IU) for a model (or class of systems within a model) occurs when there is a single tile set (along with associated functions, to be discussed) that can be used to create a seed structure from its tiles for any of the systems in the model (or class) and then intrinsically simulate them. Now we provide formal definitions which are largely taken from [16] but are modified to incorporate the dynamic of synchronicity of the L-syncTAM.

From this point on, let T be a tile set and let the scale factor be $m \in \mathbb{Z}^+$. An *m-block macrotile* over T is a partial function $\alpha : \mathbb{Z}_m^2 \dashrightarrow T$, where $\mathbb{Z}_m = \{0, 1, \dots, m-1\}$. Let B_m^T be the set of all m -block macrotiles over T . The m -block with no domain is said to be *empty*. For a general assembly $\alpha : \mathbb{Z}^2 \dashrightarrow T$ and $(x', y') \in \mathbb{Z}^2$, define $\alpha_{(x', y')}^m$ to be the m -block macrotile defined by $\alpha_{(x', y')}^m(i_x, i_y) = \alpha(mx' + i_x, my' + i_y)$ for $0 \leq i_x, i_y < m$. For some tile set S , a partial function $R : B_m^S \dashrightarrow T$ is said to be a *valid m-block macrotile representation* from S to T if for any $\alpha, \beta \in B_m^S$ such that $\alpha \sqsubseteq \beta$ and $\alpha \in \text{dom} R$, then $R(\alpha) = R(\beta)$.

For a given valid m -block macrotile representation function R from tile set S to tile set T , define the *assembly representation function*¹ $R^* : \mathcal{A}^S \rightarrow \mathcal{A}^T$ such that $R^*(\alpha') = \alpha$ if and only if $\alpha(x, y) = R\left(\alpha'_{(x,y)}^m\right)$ for all $(x, y) \in \mathbb{Z}^2$. For an assembly $\alpha' \in \mathcal{A}^S$ such that $R^*(\alpha') = \alpha$, α' is said to map *cleanly* to $\alpha \in \mathcal{A}^T$ under R^* if for all non empty blocks $\alpha'_{(x,y)}^m$, $(x, y) + (u_x, u_y) \in \text{dom}(\alpha)$ for some $(u_x, u_y) \in \{(0, \pm 1), (\pm 1, 0)\}$, or if α' has at most one non-empty m -block $\alpha'_{0,0}^m$. In other words, α' may have tiles in macrotile blocks representing empty space in α , but only if that position is adjacent to a macrotile that represents a tile in α . We call such growth “around the edges” of α' *fuzz* and thus restrict it to be adjacent to only valid macrotiles, but not diagonally adjacent (i.e. we do not permit *diagonal fuzz*).

In the following definitions, let $\mathcal{T} = (T, \sigma_T, \tau_T, l_T)$ and $\mathcal{S} = (S, \sigma_S, \tau_S, l_S)$ each be an L-syncTAS, and let R be an m -block representation function $R : B_m^S \rightarrow T$. We will define how $\mathcal{S}$ *synchronously simulates* $\mathcal{T}$ at *spatial scale factor* m and *temporal scale factor* $s \in \mathbb{Z}^+$.

► **Definition 1.** Let $\mathcal{L} = (T, \sigma, \tau, l)$ be an L-syncTAS, and let α and β be two τ -stable assemblies in $\mathcal{L}$. We say that β is *equivalent modulo l* to α , and write $\beta \approx_l \alpha$, if and only if $\exists \gamma$ such that γ is a τ -stable assembly in $\mathcal{L}$ and the following conditions hold:

1. $\alpha \rightarrow_1^{\mathcal{L}} \gamma$
2. $\alpha \sqsubseteq \beta \sqsubseteq \gamma$
3. $\exists i$, where $0 \leq i < l$, such that $|\text{dom}(\alpha)| + i = |\text{dom}(\beta)| = |\text{dom}(\gamma)| - l + i$
4. $\text{dom}(\beta) \setminus \text{dom}(\alpha) \subseteq \partial^{\mathcal{L}}(\alpha)$

Essentially, if $\beta \approx_l \alpha$, β can be thought of as an “intermediate” assembly between α and γ (where α produces γ in one step) where only i (which is $< l$) tiles have been added to α . While such an assembly β is not producible in $\mathcal{L}$ if $i > 0$, it will be useful to define such an assembly in the context of a simulator that may require many tile additions to simulate each individual tile addition of $\mathcal{L}$.

► **Definition 2.** We say that $\mathcal{S}$ and $\mathcal{T}$ have *equivalent productions (under R)*, and we write $\mathcal{S} \Leftrightarrow \mathcal{T}$ if the following conditions hold:

1. $\forall \alpha' \in \mathcal{A}[\mathcal{S}], \exists \alpha \in \mathcal{A}[\mathcal{T}]$ such that $R^*(\alpha') \approx_{l_T} \alpha$
2. $\forall \alpha \in \mathcal{A}[\mathcal{T}], \exists \alpha' \in \mathcal{A}[\mathcal{S}]$ such that $R^*(\alpha') = \alpha$
3. $\{R^*(\alpha') | \alpha' \in \mathcal{A}_{\square}[\mathcal{S}]\} = \mathcal{A}_{\square}[\mathcal{T}]$.
4. For all $\alpha' \in \mathcal{A}[\mathcal{S}]$, α' maps cleanly to $R^*(\alpha')$.

► **Definition 3.** We say that $\mathcal{T}$ *follows* $\mathcal{S}$ (under R), and we write $\mathcal{T} \dashv_R \mathcal{S}$, if $\alpha' \rightarrow^{\mathcal{S}} \beta'$ for some $\alpha', \beta' \in \mathcal{A}[\mathcal{S}]$ implies that $\exists \alpha'', \beta'' \in \mathcal{A}[\mathcal{S}]$ such that $R^*(\alpha') \approx_l R^*(\alpha'')$, $R^*(\beta') \approx_l R^*(\beta'')$ and $R^*(\alpha'') \rightarrow^{\mathcal{T}} R^*(\beta'')$.

► **Definition 4.** We say that $\mathcal{T}$ *synchronously follows* $\mathcal{S}$ (under R with temporal scale factor s), and we write $\mathcal{T} \dashv_R^s \mathcal{S}$, if $\mathcal{T} \dashv_R \mathcal{S}$ and, for every assembly sequence $\vec{\beta} = \beta_0, \beta_1, \beta_2, \dots$ in $\mathcal{S}$, there exists assembly sequence $\vec{\alpha} = \alpha_0, \alpha_1, \alpha_2, \dots$ in $\mathcal{T}$ such that:

1. $\alpha_0 = \sigma$ and $R^*(\beta_0) = \sigma$
2. For every $0 \leq i < |\vec{\alpha}|$, there is $i' \in [i \cdot (s - 1), i \cdot (s + 1)]$ such that $R^*(\beta_{i'}) = \alpha_i$

The next definition essentially specifies that every time $\mathcal{S}$ simulates an assembly $\alpha \in \mathcal{A}[\mathcal{T}]$, there must be at least one valid growth path in $\mathcal{S}$ for each of the possible next steps that $\mathcal{T}$ could make from α which results in an assembly in $\mathcal{S}$ that maps to that next step. While

¹ Note that R^* is a total function since every assembly of S represents *some* assembly of T ; the functions R and α are partial to allow undefined points to represent empty space.

this definition is unfortunately dense, it accommodates subtle situations such as where $\mathcal{S}$ must “commit to” a subset of possible representations in $\mathcal{T}$ before being explicitly mapped, under R , to any one in particular.

► **Definition 5.** We say that $\mathcal{S}$ models $\mathcal{T}$ (under R), and we write $\mathcal{S} \models_R \mathcal{T}$, if for every $\alpha \in \mathcal{A}[\mathcal{T}]$, there exists $\Pi \subset \mathcal{A}[\mathcal{S}]$ where $\Pi \neq \emptyset$ and $R^*(\alpha') = \alpha$ for all $\alpha' \in \Pi$, such that, for every $\beta \in \mathcal{A}[\mathcal{T}]$ where $\alpha \rightarrow^{\mathcal{T}} \beta$, (1) for every $\alpha' \in \Pi$ there exists $\beta' \in \mathcal{A}[\mathcal{S}]$ where $R^*(\beta') = \beta$ and $\alpha' \rightarrow^{\mathcal{S}} \beta'$, and (2) for every $\alpha'' \in \mathcal{A}[\mathcal{S}]$ where $\alpha'' \rightarrow^{\mathcal{S}} \beta'$, $\beta' \in \mathcal{A}[\mathcal{S}]$, $R^*(\alpha'') \approx_{\mathcal{T}} \alpha$, and $R^*(\beta') = \beta$, there exists $\alpha' \in \Pi$ such that $\alpha' \rightarrow^{\mathcal{S}} \alpha''$.

► **Definition 6.** We say that $\mathcal{S}$ synchronously simulates $\mathcal{T}$ (under R and temporal scale factor s) if $\mathcal{S} \Leftrightarrow_R \mathcal{T}$ (equivalent productions), $\mathcal{T} \dashv_R^s \mathcal{S}$ and $\mathcal{S} \models_R \mathcal{T}$ (equivalent dynamics).

3 Shapes

In this section we show how the syncTAM, even at temperature-1, is capable of forms of cooperative behavior and detection of empty locations. This allows the syncTAM to implement a flagpole, the shibboleth shape for cooperation, at temperature 1. This flagpole relies on *implementing* cooperation thanks to synchronization by a *cooperation gadget*. Synchronization also allows for parts of the syncTAM assembly to “probe” regions of space in order to assess whether or not some other part of the assembly has / has not arrived in that region within a fixed window of time. As such, syncTAM spaces may use empty space as input for computation. This sidesteps the gap between weak and strict self-assembly of shapes [4, 5, 13], and enables the strict self-assembly of a (modified) discrete Sierpinski Triangle. Last, the cooperation gadget proves to be enough to build the same assemblies as any *directed* aTAM system with a non-cooperative system running in the syncTAM.

First, we show how temperature-1 syncTAM systems can simulate cooperative attachments.

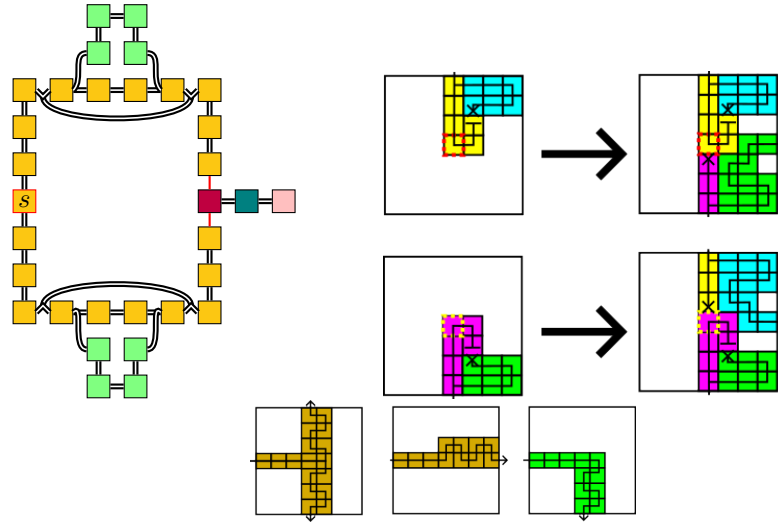
3.1 Simulating a flagpole at temperature-1

From the earliest investigations on temperature 1 and intrinsic universality [21], it has been known that the aTAM temperature 1 system cannot simulate systems which exhibit cooperation, i.e. systems with a temperature at least 2. In particular, let $\mathcal{F}$ be an aTAM system at temperature 2 as shown in Figure 1, which is a slight variation of the one in [21].

As in [21], this system features two arms growing from left to right, one at the top on line $y = 3$, and one at the bottom on line $y = -3$. Each of these arms will turn back towards the $y = 0$ line at a non-deterministic position. If both arms do so in the same column, a three tile flag (in purple, teal and pink) will grow from their meeting point. Our variation adds green tiles, which allow the arms to dally while they move right: for every 4 column they advance, they can do so either straight with the yellow tiles, in 4 time steps, or with a detour through the green tiles, which takes them 6 time steps total. Because of this, even if both arms pick the same column $x = c$ to turn back to the center, they do not necessarily reach position $(c, \pm 1)$ at the same time.

Let F be the set of shapes assembled by $\mathcal{F}$ in the aTAM at temperature 2. Note that in the syncTAM at temperature 2, $\mathcal{F}$ assembles the same set of shapes.

► **Lemma 7.** The TAS $\mathcal{F}$ cannot be scale shape-matched at temperature 1 in the aTAM: there are no integer $c \geq 1$ and TAS $\mathcal{F}_1$ at temperature 1 such that $\mathcal{A}_{\square}[\mathcal{F}_{\infty}] = c \cdot F$.



■ **Figure 1** Our variation $\mathcal{F}$ on the “Flagpole” system from [21], with the TAS on the left. Then, going clockwise, some growth patterns. The first two are in the cooperative gadget (red in the figure to the left), with either the top or bottom neighboring tile arriving first, then the other one arriving and initiating the growth of the flag. If the top arrives first, the flag can only be initiated by the arrival of the path from the bottom because growth from the top consists of two paths. The yellow, if unblocked, blocks the growth of the aqua. The bottom works symmetrically, and if it arrives first its pink path blocks its own green path but also the top’s yellow path, which allows the aqua path to continue, unblocked, to form the flag. Below, the macrotiles for the non-cooperative parts: all paths through the macrotiles are of length 11.

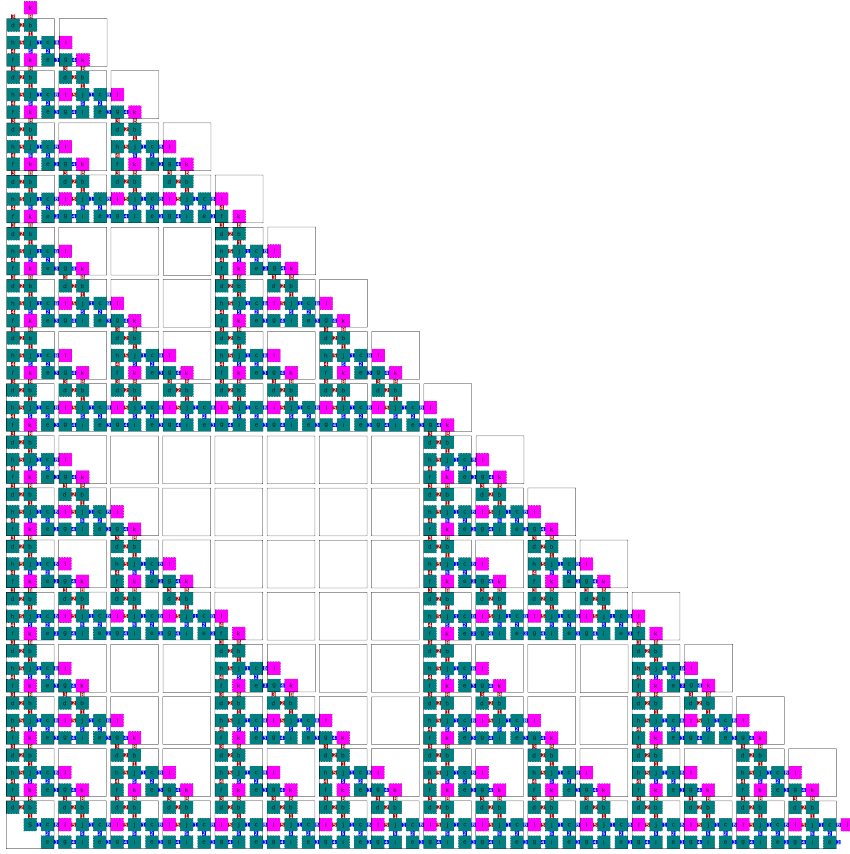
Proof. Since the system $\mathcal{F}$ is a trivial modification of the system in [21], proving Lemma 7 requires only a trivial application of the Window Movie Lemma that was used to show that the original system could not be simulated because any system attempting to simulate it would have to be able to grow assemblies of invalid shapes. Essentially, the horizontal arms can grow to arbitrary lengths and therefore each can be “pumped” once grown past some “pumping” length. If a flag can grow in a case where the top and bottom arm are of the same length, it must be able to do so via glue connections to only the top or bottom arm. By pumping only the arm with the connections to the flag tiles, that arm can be made longer than the other, but it must still be able to grow the flag. This is an invalid shape. ◀

On the other hand, there is a temperature 1 TAS $\mathcal{F}_{\text{sync}}$ which simulates $\mathcal{F}$ in the syncTAM, and thus has $\mathcal{A}_{\square}[\mathcal{F}_{\text{sync}}] = c \cdot F$. A depiction of the system with $c = 7$ and the gadgets that simulate the cooperative behavior of the tile at the base of the flagpole can be seen in Figure 1.

3.2 A modified Sierpinski triangle at temperature-1

In this section we show how a modified version of the Sierpinski triangle at scale factor 3 can self-assemble in the syncTAM. This construction is used to demonstrate the ability of systems in the syncTAM to have gadgets that detect the lack of growth in an area, even without glue cooperativity (i.e. at temperature-1) and continue growth only in that absence of growth.

Let Δ be the shape defined in Figure 2.



■ **Figure 2** A portion of the infinite modified Sierpinski triangle that strictly self-assembles in the syncTAM at temperature 1.

► **Theorem 8.** *The set Δ strictly self-assembles in the syncTAM at temperature 1 and scale 3 but does not strictly self-assemble in the aTAM at any temperature and scale.*

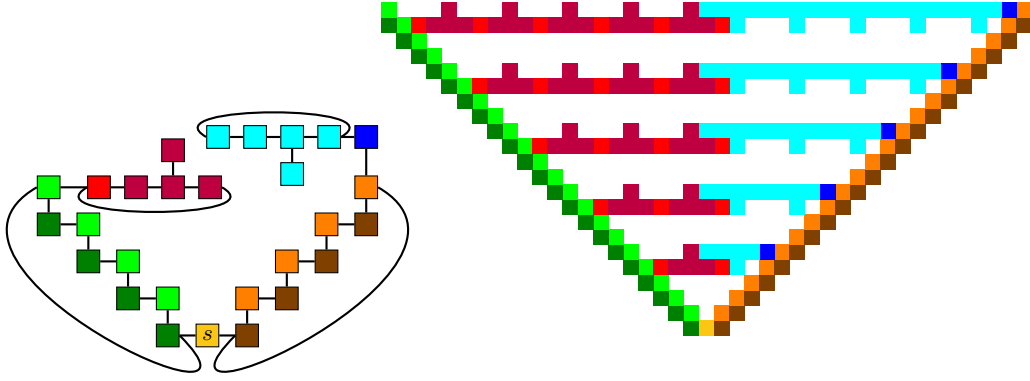
Proof. Lemma 9 shows that Δ does strictly self-assemble at temperature 1 in the syncTAM.

To see that it does not strictly self-assemble in the aTAM, notice that like the actual Sierpinski Triangle, for each k this shape features vertical lines where it has groups of tiles of constant size (here, 2) separated by gaps of size more than k , yet for no vector $\vec{p}$ does it have $\vec{p}$ -periodic, square, non-empty subpatterns of arbitrary size. Then, by the Tree Pump Theorem [5], it cannot be strictly self-assembled in the aTAM. The proof is the same as that of Lemma 31 in that paper, as only these hypothesis on the shape is used. ◀

► **Lemma 9.** *There is a syncTAM system which strictly self-assembles Δ at temperature 1 and scale-factor 3.*

We prove Lemma 9 by providing a syncTAM system which strictly self-assembles Δ .

A portion of the infinite shape can be seen in Figure 2, and the syncTAM system can be downloaded at [http://self-assembly.net/wiki/index.php/Synchronous_Tile_Assembly_Model_\(syncTAM\)](http://self-assembly.net/wiki/index.php/Synchronous_Tile_Assembly_Model_(syncTAM)), which also provides a link to the simulator for simulating the system. Our syncTAM system uses 3×3 *macrotiles* to logically “simulate” the behaviors of the aTAM tiles. The synchronous nature of the syncTAM allows us to use the timing of different portions of the assembly’s growth to guarantee that growth will be blocked from



■ **Figure 3** A system which is directed in the syncTAM but not in the aTAM. The tiles of the temperature 1 syncTAM system $\mathcal{S}$ (the seed is the yellow s tile), and a portion of the infinite shape V assembled by $\mathcal{S}$ in the syncTAM. Growth begins from the seed tile at the bottom and proceeds with the diagonal “arms” growing upward to the left and right, and with red rows growing right from the left arm, and blue rows growing left from the right arm, at every 4th step. The green part is L , the orange and brown part is R , the blue parts are the $B_k^+ \cup T_k^+$, while the red parts are the $B_k^- \cup T_k^-$.

proceeding whenever growth of a stage in one direction reaches the correct length because it crashes into growth from the other direction that has also reached the correct length. In the aTAM, it cannot be guaranteed that the growth of both portions will proceed at identical rates, and thus the necessary blocking may not occur. However, by adding tiles in every frontier location at every step the syncTAM provides such a guarantee and thus provides a powerful tool for designing systems. A detailed description of the system can be found in the appendix of the full version of this paper online [3].

4 Matching Assemblies of Directed aTAM Systems With Non-cooperative syncTAM Systems

Any system which is directed in the aTAM is also directed in the syncTAM, since any syncTAM production must be an aTAM production [2]. In this directed case, it is actually possible to assemble a scaled version of the aTAM system’s terminal assembly. Note that being directed in the aTAM is a stronger condition than being directed in the syncTAM: in order to be directed in the aTAM, a TAS needs to be deterministic in the face of asynchronism of tile additions. For example, the system in Figure 3 is directed in the syncTAM, assembling the production depicted on the figure, while in the aTAM, the meeting point of the two parts of each rung varies based on the particular assembly sequence chosen.

► **Theorem 10.** *If $\mathcal{S}$ is directed in the aTAM, there is a temperature 1 syncTAM system $\mathcal{S}'$ which assembles a scaled version of $\alpha \in \mathcal{A}_{\square}[\mathcal{S}]$, for some scale factor s and macrotile representation function R .*

The proof of this theorem can be found in Appendix A of the appendix, with more detail available in the appendix of the full version of this paper [3]. The construction leverages the cooperation gadget of Section 3.1, using the directedness of $\mathcal{S}$ to argue away timing problems due to the need to simulate the asynchronicity of $\mathcal{S}$.

5 L-SyncTAM Results

First we present Lemma 11, which will be useful for several of the proofs in this section. Lemma 11 shows that when a certain set of requirements are met for subassemblies produced by systems in the aTAM or L-syncTAM, all such subassemblies grow independently of each other.

Given a shape S let $\text{bound}(S)$ refer to the set of unit vectors in $\mathbb{Z}^2$ that form its perimeter, or *boundary*. Given two points $l_1, l_2 \in \mathbb{Z}^2$, let $\text{dist}(l_1, l_2)$ be a function that returns the minimal Manhattan distance between points l_1 and l_2 .

► **Lemma 11** (Eventually Independent Subassemblies). *Let $\mathcal{T} = (T, \sigma, \tau, l)$ be a shape-directed L-syncTAS in the L-syncTAM for any $l \geq 1$ and let S be the shape that $\mathcal{T}$ strictly self-assembles. If there exists a cut C of $\mathbb{Z}^2$ that partitions S into shapes $s_0, s_1, \dots, s_n$ such that:*

1. $\text{dom } \sigma \subseteq s_0$, [The seed is completely contained in s_0]
2. $\forall 0 < i \leq n, |s_i| = \infty$, [Other than perhaps s_0 , all must be infinite]
3. For each $0 < i \leq n$, $|C \cap \text{bound}(s_i)| < \infty$, [C has a finite intersection with the boundary of each except maybe s_0]
4. For all $0 < i \neq j \leq n$, $\forall l_1 \in s_i, \forall l_2 \in s_j, \text{dist}(l_1, l_2) \geq 3$. [The minimum distance between points in any pair of shapes (not including s_0) is at least 3.]
5. For each $0 < i \leq n, C \cap \text{bound}(s_i) = \text{bound}(s_0) \cap \text{bound}(s_i)$, [C contains the entire shared boundary of each s_i with s_0]

then, the following holds: Let $\alpha \in \mathcal{A}_{\square}[\mathcal{T}]$ be any terminal assembly of $\mathcal{T}$ and let $\vec{\alpha}$ be any arbitrary assembly sequence that produces α . Recall that $\text{dom } \alpha = S$. There exists some $f \in \mathbb{N}$ such that for every assembly step of $\vec{\alpha}$ whose index (i.e. location in the sequence) is greater than f , $\forall 0 < i \leq n$ there is a frontier location inside of s_i .

Proof. To prove Lemma 11, we first note that since the seed σ is completely contained in s_0 , no location of s_i initially has a tile, and for all $0 < i \leq n$, the first location in s_i that receives a tile must be adjacent to cut C since that cut separates the subassembly in s_i (which we will call α_i) from the rest of the assembly. For any (necessarily empty) location $\vec{l} \in \mathbb{Z}^2$ to become a frontier location, at least one neighboring location $\vec{l}'$ must have been a frontier location that became filled with a tile. If $\vec{l}$ first becomes a frontier location during some assembly step s , then some neighboring location must have received a tile during step $s - 1$. This means that any location in s_i that receives a tile must have been included in a path of successive, adjacent frontier locations that originated adjacent to the seed and at some point crossed C .

By the conditions of the lemma, the cut C only overlaps with a finite portion of the perimeter of s_i . Also, since each such s_i has infinite size that is filled (in the limit) by tiles, there must be an infinite number of assembly steps during which there exists a frontier location within s_i during $\vec{\alpha}$. For any assembly step s during which there is not a frontier location in s_i , since there must exist a later step where a location in s_i receives a tile, there must exist a later step $s' > s$ where there exists a frontier location adjacent to C , as this is the only boundary into which a path of frontier locations can enter s_i . Since the locations adjacent to C are finite, this can only happen a finite number of times. Therefore, there exists some constant f_i such that after the f_i th assembly step no more tiles are placed adjacent to C , and since the number of tiles placed within s_i is infinite, for the infinite number of steps greater than f_i , there must be frontier locations within s_i .

Finally, since the argument holds for each s_i , each has its own such constant f_i and the f of the lemma is simply the maximum of them. Corollary 12 directly follows from the fact that each shape s_i for $0 < i \leq n$ is separated by a distance greater than that of fuzz. ◀

► **Corollary 12.** *Let $\mathcal{S} = (T, \sigma, \tau, l)$ be an L -syncTAS in the L -syncTAM for any $l \geq 1$. If $\mathcal{S}$ synchronously simulates some $\mathcal{T}$ such that Lemma 11 holds for $\mathcal{T}$, then Lemma 11 holds for $\mathcal{S}$.*

Corollary 12 follows immediately from Lemma 11 since the scaled shape assembled by $\mathcal{S}$ must have a similarly scaled and translated cut C' for which all of the same properties hold (noting that the condition that the distance required between shapes by Item 4 ensures sub-shapes are still disjoint even accounting for fuzz in the simulation), and therefore so does the proof.

5.1 The class of L-syncTAM systems is not IU for any $l > 1$

For any fixed $l \geq 2$, the class of L-syncTAM systems with synchronicity l is not intrinsically universal (for synchronous simulation). For any fixed candidate simulator, there are systems that cannot be simulated by that simulator.

► **Lemma 13.** *For all $l \geq 2$, there is an infinite family $(\mathcal{S}_n^l)_{n \in \mathbb{N}}$ of L -syncTAM systems with synchronicity l such that for each n , $\mathcal{S}_n^l$ has $n + 5$ tiles and cannot be synchronously simulated in the L -syncTAM at synchronicity l with any temporal scale factor s and any spatial scale factor greater than 1.*

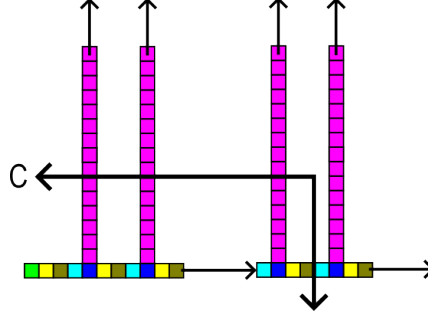
Proof. Fix n . Consider the system of Figure 4, with the green seed tile fixed at $(0, 0)$, and 4 tile types (yellow, gold, aqua, and blue) that grow a row with a periodic sequence of those 4 tile types infinitely far to the right. From each blue tile, a periodic sequence of n unique tile types attach (all shown as pink) infinitely often, to create infinite vertical columns. This system is directed, and strictly-self assembles the shape shown in Figure 4. Call this system $\mathcal{S}_n^l = (T, \sigma, 1, l)$ and note that $|T| = n + 5$.

We will prove Lemma 13 by contradiction, so assume that L-syncTAM system $\mathcal{S}'$ synchronously simulates $\mathcal{S}_n^l$ at scale factor $c > 1$ and some temporal scale factor s . We now inspect a particular assembly sequence in $\mathcal{S}_n^l$ and show that $\mathcal{S}'$ does not synchronously follow $\mathcal{S}_n^l$.

Let $\vec{\alpha}$ be an assembly sequence in $\mathcal{S}_n^l$ that proceeds as follows. The horizontal row grows to a length $> 4((3s + 1)l + (l - 1)) + 2$ by starting from the seed and attaching $(3s + 1)l + (l - 1)$ copies of the 4-tile repeating segment. From each blue tile, a vertical column begins to grow. The columns can grow in any order, resulting in $(3s + 1)l + (l - 1)$ vertical columns which will each ultimately be infinite in height. Note that the horizontal row will continue to grow and initiate further upward columns, but we will not need to include them in our analysis.

There must be an assembly sequence $\vec{\alpha}'$ in $\mathcal{S}'$ that models $\vec{\alpha}$. We let $\vec{\alpha}'$ proceed until it has not only grown sufficiently far horizontally (i.e. $> (c \cdot 4((3s + 1)l + (l - 1)) + 2)$), but also continued growth so that all subassemblies mapping to the $(3s + 1)l + (l - 1)$ vertical arms are of sufficient height that Corollary 12 of Lemma 11 can be applied. This results in cut C separating the $(3s + 1)l + (l - 1)$ vertical columns into shapes $s_1, s_2, \dots, s_{(3s+1)l+(l-1)}$ of the lemma, so that each vertical column is guaranteed to have a frontier location in all further assembly steps. The portion of the horizontal bar containing the seed will be s_0 , and the rest of the assembly to the east of the cut will be $s_{(3s+1)l+l}$. We extend assembly sequence $\vec{\alpha}'$ as follows.

We will refer to each vertical column as V_i for $0 \leq i < (3s + 1)l + (l - 1)$, and to the rightmost $l - 1$ vertical columns of that group as *buffer* columns. Since $\mathcal{S}'$ has synchronicity l , there must be l tiles placed during every assembly step. Therefore, starting with $i = 0$, we perform assembly steps such that each places one tile in each of the buffer columns (using



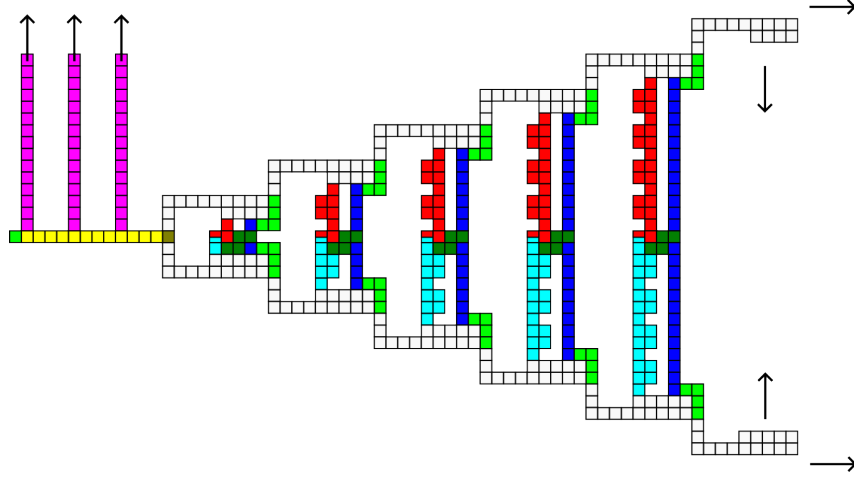
■ **Figure 4** A schematic depiction of a TAS in the L-syncTAM with synchronicity l that cannot be simulated by a system that is IU for the L-syncTAM at synchronicity l . The seed is the green tile on the bottom left. An infinite number of repetitions of 4 tiles attach to its right. From each blue tile, a column grows upward infinitely, with repetitions of a series of n unique tile types attaching. Cut C of Lemma 11 is also shown.

$l - 1$ of the tile placements), and one tile in V_i . We repeat this until there is a frontier location, which we will refer to as a *nonresolving frontier location*, in V_i so that when a tile is placed there, no new macrotile resolves to a tile of $\mathcal{S}_n^l$. That is, assuming the assembly α' in $\mathcal{S}'$ maps under R to some assembly α in $\mathcal{S}_n^l$ (i.e. $R(\alpha') = \alpha$), when a tile is placed in that nonresolving frontier location to create assembly α'' in $\mathcal{S}'$, it is still the case that $R(\alpha'') = \alpha$. This must be possible because the scale factor of the simulation is $c > 1$ and therefore macrotiles must have dimensions > 1 , meaning that any path from the south to the north of a macrotile must be of length > 1 and a macrotile only resolves during a single assembly step. Thus, there must be at least one step during which there is a tile placed that does not cause the macrotile to resolve, and that tile's location is a nonresolving frontier location immediately prior to that.

Once it occurs that a nonresolving frontier location exists in V_i , we increment to $i + 1$ and repeat until nonresolving frontier locations exist in each of the $(3s + 1)l$ leftmost vertical columns. At this point, let the current assembly be β' , and let $\beta = R(\beta')$ be the assembly in $\mathcal{S}_n^l$ to which it maps. We now perform $3s + 1$ assembly steps as follows. For $0 \leq p < 3s + 1$, each step consists of placing a single tile in each V_q , for $lp \leq q < (l+1)p$, into the nonresolving frontier locations. In this way, each column in each set of l columns starting from the left receives a single tile. However, since all locations were nonresolving frontier locations, that means that the assembly sequence proceeded $3s + 1$ steps without any change in the assembly to which it maps in $\mathcal{S}_n^l$. This breaks Definition 4, i.e. the simulated system does not synchronously follow the simulator because it must continue to add l tiles per assembly step, but even allowing for the maximum amount of time within the temporal scaling of s , which is a window of size $3s$, $\mathcal{S}'$ resolves to no additional tiles. This is a contradiction that $\mathcal{S}'$ synchronously simulates $\mathcal{S}_n^l$ and thus Lemma 13 is proven. ◀

► **Theorem 14.** *For all $l \geq 2$, the class of L-syncTAM systems with synchronicity l is not intrinsically universal.*

Proof. Let $l \in \mathbb{N}$. Any fixed L-syncTAM system at synchronicity l with t tile types can only simulate systems with at most $(t + 1)^s$ tiles at scale s , as that is the cardinality of the domain of the representation function. But by Lemma 13, there are infinitely many L-syncTAM systems at synchronicity l which can only be simulated at scale 1. Hence, regardless of the size of the tile set of a fixed system simulator, since it must be finite, no fixed system can simulate all of them and the class of L-syncTAM systems with synchronicity l is not intrinsically universal. ◀



■ **Figure 5** Initial portion of shape S_5 (i.e. S_l for $l = 5$) from the proof of Theorem 15. The seed tile (green) initiates growth of the yellow line which initiates the growth of $l - 2$ arms (pink) that each grow infinitely upward. The gold tile at the end of the yellow row initiates growth of the cone (white) that grows infinitely to the right, with rungs (red, aqua, green, blue, and lime green) that periodically grow between the sides of the cone. A depiction of the growth sequence of the cone can be seen in Figure 7.

5.2 Shapes that cannot self-assemble with less synchronization

We now present a result that shows that for any synchronicity value $l > 1$, there exists a shape which cannot be made by any system with lower synchronicity. This is a much stronger statement than simply saying there is a system with synchronicity l that cannot be simulated by any with lower synchronicity. This instead shows that, even without the constraints of following a system's dynamics, the shape simply cannot be strictly produced.

For any $l > 1$, the shape pictured on Figure 5 strictly self-assembles in an L-syncTAM system with synchronicity l , but not in any system with lower synchronicity.

► **Theorem 15.** *For each $l > 1$, there exists a shape S_l such that:*

1. *There exists L-syncTAM system $\mathcal{L}_l = (T, \sigma, 2, l)$ such that $\mathcal{L}_l$ strictly self-assembles S_l .*
2. *For all $l' < l$, there exists no L-syncTAM system $\mathcal{L}' = (T', \sigma', \tau, l')$ and scale factor $c \in \mathbb{Z}^+$ such that $\mathcal{L}'$ strictly self-assembles $c \cdot S_l$.*

The proof of Theorem 15 relies on observing that any simulator for this system must have $l - 2$ frontier positions for the infinite arms growing upwards. On the other hand, by a pumping argument, it cannot encode the positions within the cone in order to be able to find the middle without keeping pairs of tile additions in sync. Hence synchronicity $\geq l$ is needed in order to assemble that shape. The proof may be found in Appendix B of the appendix, with more detail available in the appendix of the full version of this paper [3].

5.3 Systems that cannot be simulated by systems with greater synchronization

The following theorem states that for any $l \geq 1$, there is a system in the L-syncTAM with synchronicity l that cannot be simulated by any L-syncTAM system whose synchronicity l' is greater than l , i.e. the additional synchronization causes any system to fail to correctly simulate.

► **Theorem 16.** *For each $l \geq 1$, there exists an L -syncTAM system $\mathcal{S}_l$ with synchronicity l such that, for any $l' > l$, there exists no L -syncTAM system $\mathcal{S}'$ with synchronicity l' which simulates $\mathcal{S}_l$.*

Proof. We prove Theorem 16 by first describing a directed L -syncTAS $\mathcal{S}_l$, for arbitrary $l \geq 1$. A schematic depiction can be seen in Figure 6. It consists of a green seed tile fixed at $(0,0)$, and 4 tile types (yellow, gold aqua, and blue) that grow a row with a periodic sequence of those 4 tile types infinitely far to the right. From each blue tile, a single tile (pink) attaches to itself infinitely often, to create infinite vertical columns. This system is similar to that which was shown in the proof of Lemma 13, with the only modification being that each infinite vertical column only contains a single repeating tile type. Call this system $\mathcal{S}_l = (T, \sigma, 1, l)$ and note that $|T| = 6$.

Now, let l' be an integer $> l$. We will prove Theorem 16 by contradiction, so assume that $\mathcal{S}'$ is an L -syncTAS which synchronously simulates $\mathcal{S}_l$ at some temporal scale factor s and spatial scale factor c . Let us now inspect a particular assembly sequence of $\mathcal{S}'$.

Define $k = (3s + 1)l' + (l' - 1)$, a constant whose value will become clear later. Let $\vec{\alpha}$ be an assembly sequence in $\mathcal{S}_l$ that proceeds as follows. The horizontal row grows to a length $> 4k + 2$ by starting from the seed and attaching k copies of the 4-tile repeating segment. From each blue tile, a vertical column begins to grow, resulting in k vertical columns which will ultimately be infinite in height. Note that the horizontal row will continue to grow and initiate further upward columns, but we do not need to include them in our analysis.

There must be an assembly sequence $\vec{\alpha}'$ in $\mathcal{S}'$ that models $\vec{\alpha}$. Let $\vec{\alpha}'$ proceed until it has not only grown sufficiently far horizontally (i.e. $> c \cdot (4k + 2)$), but also continues growth so that all subassemblies mapping to the k vertical columns are of sufficient height that Corollary 12 of Lemma 11 may be applied, with cut C separating k vertical columns into shapes $s_1, s_2, \dots, s_k$, such that each vertical column is guaranteed to have a frontier location in all future assembly steps. The portion of the horizontal bar containing the seed will be s_0 , and the rest of the assembly to the east of the cut will be s_{k+1} . We extend assembly sequence $\vec{\alpha}'$ as follows.

We'll refer to each vertical column as V_i for $0 \leq i < k$, and to the rightmost $l' - 1$ of those as *buffer* columns. Since $\mathcal{S}'$ has synchronicity l' , there must be l' tiles placed during each assembly step. Therefore, starting with $i = 0$, we perform assembly steps such that each places one tile in each of the buffer columns (using $l' - 1$ of the tile placements in each assembly step), and one in V_i . We repeat this until there is a frontier location, which we will refer to as a *resolving frontier location*, in V_i so that there exists a tile type in $\mathcal{S}'$ that may be placed there whose attachment causes a new macrotile to resolve to a tile in $\mathcal{S}_l$. We will refer to this tile as a *resolving tile*. That is, assuming the assembly α' in $\mathcal{S}'$ maps under R to some assembly α in $\mathcal{S}$ (i.e. $R(\alpha') = \alpha$), when a resolving tile is placed in that resolving frontier location to create an assembly α'' in $\mathcal{S}'$, it is the case that $R(\alpha'') \neq \alpha$.

Once it occurs that a resolving frontier location exists in V_i , we increment to $i + 1$ and repeat until resolving frontier locations exist in each of the $(3s + 1)l'$ leftmost vertical columns. At this point, let the current assembly be β' , and let $\beta = R(\beta')$ be the assembly in $\mathcal{S}$ to which it maps. We now perform $3s + 1$ assembly steps as follows. For $0 \leq p < 3s + 1$, each step consists of placing a single resolving tile into the resolving frontier locations in each V_q for $l'p \leq q < (l' + 1)p$. In this way, each column in each set of l' columns starting from the left receives a single tile, causing l' macrotiles to resolve. This breaks Definition 4, i.e. the simulated system does not synchronously follow the simulator because the simulator adds $> l$ tiles in every assembly step for each of $3s + 1$ consecutive steps, causing the dynamics to differ even beyond compensation by the temporal scaling factor s . This is a contradiction of the claim that $\mathcal{S}'$ simulates $\mathcal{S}_l$, and thus Theorem 16 is proven. ◀

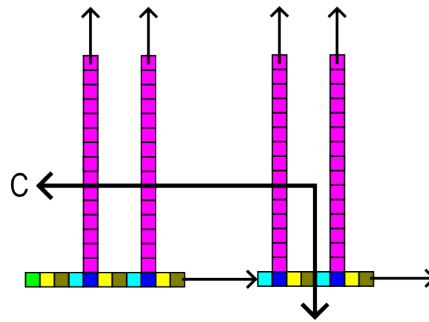


Figure 6 A depiction of a TAS in the L-syncTAM with synchronicity l that cannot be simulated by any TAS in the L-syncTAM with synchronicity $l' > l$. The seed is the green tile on the bottom left, and an infinite number of repetitions of 4 tiles can attach to its right. From each blue tile, a column grows upward infinitely, with copies of the same pink tile type attaching. Cut C of Lemma 11 is also shown.

References

- 1 Kimberly Barth, David Furcy, Scott M. Summers, and Paul Totzke. Scaled tree fractals do not strictly self-assemble. In *Unconventional Computation & Natural Computation (UCNC) 2014, University of Western Ontario, London, Ontario, Canada July 14-18, 2014*, pages 27–39, 2014. doi:10.1007/978-3-319-08123-6_3.
- 2 Florent Becker, Phillip Drake, Matthew Patitz, and Trent Rogers. Synchronous Versus Asynchronous Tile-Based Self-Assembly. In *DNA 2025, Lyon, France, August 2025*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.31.9.
- 3 Florent Becker, Phillip Drake, Matthew J. Patitz, and Ryder Smith. Powers and limitations of synchronous self-assembly, 2026. arXiv:2605.27604.
- 4 Florent Becker, Daniel Hader, and Matthew J Patitz. Strict self-assembly of discrete self-similar fractals in the abstract tile assembly model. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'25), New Orleans, USA*, pages 2387–2466. SIAM, 2025. doi:10.1137/1.9781611978322.80.
- 5 Florent Becker, Daniel Hader, and Matthew J. Patitz. Strict self-assembly of discrete self-similar fractals in the abstract tile assembly model. *Algorithmica*, 88(1):18, 2026. doi:10.1007/S00453-025-01340-W.
- 6 Nathaniel Bryans, Ehsan Chiniforooshan, David Doty, Lila Kari, and Shinnosuke Seki. The power of nondeterminism in self-assembly. *Theory of Computing*, 9:1–29, 2013. doi:10.4086/toc.2013.v009a001.
- 7 Matthew Cook, Yunhui Fu, and Robert T. Schweller. Temperature 1 self-assembly: Deterministic assembly in 3D and probabilistic assembly in 2D. In *SODA 2011: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2011. doi:10.1137/1.9781611973082.45.
- 8 Erik D. Demaine, Matthew J. Patitz, Trent A. Rogers, Robert T. Schweller, Scott M. Summers, and Damien Woods. The two-handed assembly model is not intrinsically universal. In *40th International Colloquium on Automata, Languages and Programming, ICALP 2013, Riga, Latvia, July 8-12, 2013*, Lecture Notes in Computer Science. Springer, 2013. doi:10.1007/978-3-642-39206-1_34.
- 9 David Doty, Jack H. Lutz, Matthew J. Patitz, Robert T. Schweller, Scott M. Summers, and Damien Woods. The tile assembly model is intrinsically universal. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012*, pages 302–310, 2012. doi:10.1109/FOCS.2012.76.

- 10 David Doty, Matthew J. Patitz, Dustin Reishus, Robert T. Schweller, and Scott M. Summers. Strong fault-tolerance for self-assembly with fuzzy temperature. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS 2010)*, pages 417–426, 2010. doi:10.1109/FOCS.2010.47.
- 11 Sándor P. Fekete, Jacob Hendricks, Matthew J. Patitz, Trent A. Rogers, and Robert T. Schweller. Universal computation with arbitrary polyomino tiles in non-cooperative self-assembly. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2015), San Diego, CA, USA, January 4-6, 2015*, pages 148–167, 2015. doi:10.1137/1.9781611973730.12.
- 12 Bin Fu, Matthew J. Patitz, Robert T. Schweller, and Robert Sheline. Self-assembly with geometric tiles. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *LNCS*, pages 714–725. Springer, 2012. doi:10.1007/978-3-642-31594-7_60.
- 13 David Furcy and Scott M. Summers. Scaled pier fractals do not strictly self-assemble. *Natural Computing*, 16(2):317–338, 2017. doi:10.1007/S11047-015-9528-Z.
- 14 Oscar Gilbert, Jacob Hendricks, Matthew J. Patitz, and Trent A. Rogers. Computing in continuous space with self-assembling polygonal tiles. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2016), Arlington, VA, USA January 10-12, 2016*, pages 937–956, 2016.
- 15 Daniel Hader, Aaron Koch, Matthew J. Patitz, and Michael Sharp. The impacts of dimensionality, diffusion, and directedness on intrinsic universality in the abstract tile assembly model. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, January 5-8, 2020*, pages 2607–2624, Salt Lake City, UT, USA, 2020. SIAM. doi:10.1137/1.9781611975994.159.
- 16 Daniel Hader and Matthew J. Patitz. The impacts of dimensionality, diffusion, and directedness on intrinsic cross-model simulation in tile-based self-assembly. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023*, volume 261 of *LIPIcs*, pages 71:1–71:19, Paderborn, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2023.71.
- 17 Jacob Hendricks, Matthew J. Patitz, and Trent A. Rogers. Universal simulation of directed systems in the abstract tile assembly model requires undirectedness. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2016), New Brunswick, New Jersey, USA October 9-11, 2016*, pages 800–809, 2016. doi:10.1109/FOCS.2016.90.
- 18 Jacob Hendricks, Matthew J. Patitz, Trent A. Rogers, and Scott M. Summers. The power of duples (in self-assembly): It’s not so hip to be square. *Theoretical Computer Science*, 743:148–166, 2018. doi:10.1016/J.TCS.2015.12.008.
- 19 James I. Lathrop, Jack H. Lutz, Matthew J. Patitz, and Scott M. Summers. Computability and complexity in self-assembly. *Theory Comput. Syst.*, 48(3):617–647, 2011. doi:10.1007/s00224-010-9252-0.
- 20 James I. Lathrop, Jack H. Lutz, and Scott M. Summers. Strict self-assembly of discrete Sierpinski triangles. *Theoretical Computer Science*, 410:384–405, 2009. doi:10.1016/J.TCS.2008.09.062.
- 21 Pierre-Étienne Meunier, Matthew J. Patitz, Scott M. Summers, Guillaume Theyssier, Andrew Winslow, and Damien Woods. Intrinsic universality in tile self-assembly requires cooperation. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA 2014), (Portland, OR, USA, January 5-7, 2014)*, pages 752–771, 2014. doi:10.1137/1.9781611973402.56.
- 22 Pierre-Étienne Meunier, Damien Regnault, and Damien Woods. The program-size complexity of self-assembled paths. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, June 22-26, 2020*, pages 727–737, Chicago, IL, USA, 2020. ACM. doi:10.1145/3357713.3384263.

- 23 Pierre-Étienne Meunier and Damien Woods. The non-cooperative tile assembly model is not intrinsically universal or capable of bounded turing machine simulation. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 328–341, New York, NY, USA, 2017. ACM. doi:10.1145/3055399.3055446.
- 24 Matthew J. Patitz, Robert T. Schweller, and Scott M. Summers. Exact shapes and Turing universality at temperature 1 with a single negative glue. In Luca Cardelli and William M. Shih, editors, *DNA Computing and Molecular Programming - 17th International Conference, DNA 17, Pasadena, CA, USA, September 19-23, 2011. Proceedings*, volume 6937 of *Lecture Notes in Computer Science*, pages 175–189. Springer, 2011. doi:10.1007/978-3-642-23638-9_15.
- 25 Matthew J. Patitz and Scott M. Summers. Self-assembly of decidable sets. *Natural Computing*, 10(2):853–877, 2011. doi:10.1007/s11047-010-9218-9.
- 26 Paul W. K. Rothmund and Erik Winfree. The program-size complexity of self-assembled squares (extended abstract). In *STOC '00: Proceedings of the thirty-second annual ACM Symposium on Theory of Computing*, pages 459–468, Portland, Oregon, United States, 2000. ACM. doi:10.1145/335305.335358.
- 27 David Soloveichik and Erik Winfree. Complexity of self-assembled shapes. *SIAM Journal on Computing*, 36(6):1544–1569, 2007. doi:10.1137/S0097539704446712.
- 28 Erik Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, California Institute of Technology, June 1998.

A Proof of Theorem 10

► **Theorem 10.** *If S is directed in the aTAM, there is a temperature 1 syncTAM system S' which assembles a scaled version of $\alpha \in \mathcal{A}_{\square}[S]$, for some scale factor s and macrotile representation function R .*

Proof. In the aTAM as well as in the syncTAM, any tile attachment takes place thanks to a set of *predecessors*. The set of directions of those predecessors can be (up to rotation) one of $\{N\}, \{N, E\}, \{N, S\}, \{N, E, S\}, \{N, E, S, W\}$. In this proof, these sets of directions are represented as a binary tree, respectively as follows: N , $N \wedge E$, $N \wedge S$, $(N \wedge E) \wedge S$, $(N \wedge E) \wedge (S \wedge E)$. Likewise, for each set I of predecessors, the value of the input glues of the position is a function $i : I \rightarrow G$, will be considered as the binary tree obtained by mapping i on the leaves of the representation of i . For instance, the input mapping glues n, e, s on the N, E , and S , respectively, $\{N \mapsto n, E \mapsto e, S \mapsto s\} : \{N, E, S\} \rightarrow G$, is represented as the binary tree $(n \wedge e) \wedge s$.

Let $\mathcal{S} = (T, \sigma, \tau)$ be a TAS which is directed in the aTAM, and G its set of glues. Let $g = |G|$ be the number of glues in T . The syncTAM system $\mathcal{S}' = (T', \sigma', 1)$ defined below, with set of glues G' , creates a scaled version of $\alpha \in \mathcal{A}_{\square}[S]$, the terminal assembly of $\mathcal{S}$, in the syncTAM at scale $s = 10g^4 + 1$ with the representation function $R : B_m^{T'} \dashrightarrow T$ also defined below.

Interface between macrotiles and invariants. The set of glues G' of $\mathcal{S}'$ is made of two parts: $E \uplus I$. The set E of *external* glues contains a glue g' (of strength 1) for each $g \in G$ and $d \in \{N, E, S, W\}$. The set I of *internal* glues will be defined below, as the need for those glues arises in the description of the construction. For each tile $t \in T$, there is a unique tile $\hat{t} \in T'$.

Given scale factor s , the macrotiles of $\mathcal{S}'$ that map under R to tiles of $\mathcal{S}$ will be $s \times s$ regions. Let $i^* = \lceil s/2 \rceil + 1$; on each side of a macrotile, the i^* -th position (counting clockwise) is the *input position* in that direction, with the position immediately before it (counting clockwise) being the *output position*. (These are depicted in Figure 7 of the full version of this paper [3] by the arrows pointing into and out of each side.)

There is an invariant associated with the representation of the glues which $\mathcal{S}'$ will uphold:

for any assembly α of $\mathcal{S}'$ where some $s \times s$ square M maps to a tile t under R , there is an α' such that $\alpha \rightarrow_{\mathcal{S}'} \alpha'$ and the sequence $\vec{g}$ of glues (read clockwise) on each side d of M satisfies either $\vec{g}_{i^*} = t_d$ or $\vec{g}_{s-i^*} = t_d$.

That is, every macrotile in $\mathcal{S}'$ that maps to a tile in $\mathcal{S}$ will have (or be able to grow into a macrotile that has) on each side a glue, in either the input or output position, that represents the glue on the corresponding side of the tile in $\mathcal{S}$ that the macrotile maps to.

Because $\mathcal{S}$ is directed in the aTAM, for any attachment of a tile of type t at position z in a production α of $\mathcal{S}$ and any subset of neighbors of z which are filled before z and have glues with binding strength summing to τ or more facing towards z , there must not be any tile other than t that is attachable thanks to this subset of neighbors.

Macrotile Anatomy. Each macrotile is divided into regions. The *input position* is the i^* -th position of each side of the macrotile (counting clockwise), while the *output position* is the $(s - i^*)$ -th. Thus, each input position of a macrotile is aligned with the output position of the neighboring macrotile. For each possible set of inputs I , there is a *computation region*, with room for a *computation zone* for each combination of glues $i \in G^I$. All these zones and regions are disjoint and separated by the *wire region*, to be used for communication between them. As an exception to this rule, the wire region also passes through the computation zones in the $\{N, S\}$ and $\{E, W\}$ computing regions. (These are special cases because they handle the situation when a tile in $\mathcal{S}$ attaches via the cooperation of two input glues across-the-gap, i.e. from N and S or from E and W , as opposed to there being a single input glue or a set of input glues that are all adjacent to each other.) The *center* of the macrotile is the position at $(\lfloor s/2 \rfloor, \lfloor s/2 \rfloor)$; it is outside all regions. The tile there is always one of the unique $\hat{t}$ tiles that corresponds to a tile of T . The representation function is defined by $R(M) = t$ iff the center position of M is occupied by $\hat{t}$.

Input. For each glue $g \in G$ and direction d , there is a set of hard-coded tile types such that from glue g' next to the input position in direction d , a path grows to the computation zone associated with $(d \mapsto g)$ within the d computation region, where it starts the growth of the computation gadget for $(d \mapsto g)$.

Computation gadgets. A computation gadget for a set of inputs I and the values $i \in G^I$ behaves thus: if there is a unique possible attachment in $\mathcal{S}$ with inputs i , say of tile type t , have the macrotile output t (see below for how output proceeds). Otherwise, send i to some computation gadgets having i as a child (within the regions associated with trees having I as a child):

- those with two adjacent inputs
- those which would result in a unique attachment

The computations gadgets for singletons in each direction d are trivial. If glue g has strength at least τ and there is a unique tile type t with glue g on side d , then the gadget is hard-coded to have the macrotile start outputting tile type t (see below). Otherwise, the gadget is hard-coded to forward the input $d \mapsto i$ to the relevant computation gadgets.

For non-singleton sets of input directions, the computation gadget uses a mechanism similar to that of Figure 1 which allows it to wait for both of its inputs to arrive. When both of its inputs have arrived, the behavior is the same as in the singleton case: a unique candidate tile attachment t triggers the output of t , otherwise the input value is transferred to the parent computation gadgets (if there are any).

Output. When a computation gadget logically outputs the attachment of a tile t , it grows a path towards the center of the macrotile and from there to each side, where it passes through the input position before outputting glue t'_d at the output position. The paths from the center to the output positions are designed in such a way that all input and output positions are reached at the same time (i.e. the paths are of the same length, which may require some to “stall” by growing indirectly toward their destinations for short distances).

Wires through the NS and EW computation zones. Wires which need to cross a computation zone within the $\{N, S\}$ or $\{E, W\}$ computation region send two probes, each trying to cross a different input wire of the computation zone then meeting again once they have passed it. The only case where both are blocked is if both inputs are present and they beget a unique tile attachment. In that case, the macrotile is already outputting the value of that attachment (which must be the same value since $\mathcal{S}$ is directed), so the blocking does not prevent correct output.

The claim is that this system builds a scaled version of $\alpha \in \mathcal{A}_{\square}[\mathcal{S}]$. First note that before each output position is reached, all input positions have been blocked. This means that there cannot be a path of attachments between an output position of a macrotile and an input position of the same macrotile.

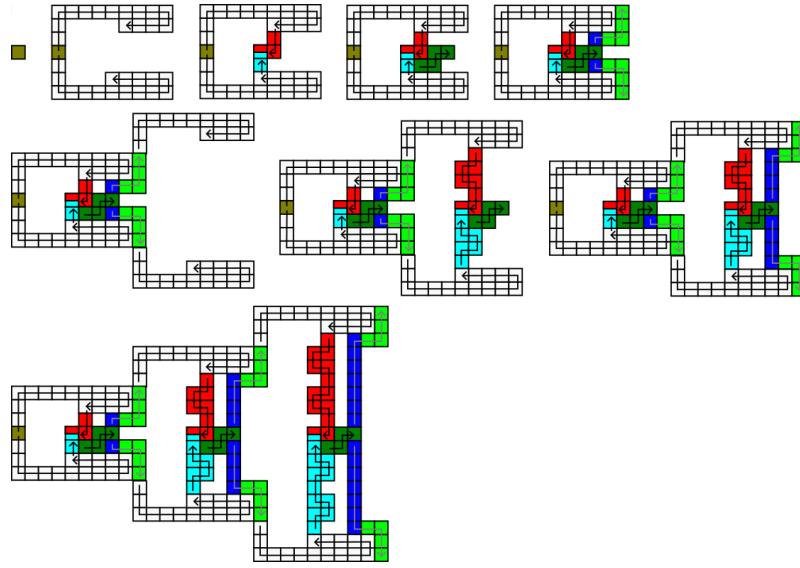
Moreover, conflicts between paths can only arise if two different computing gadgets have found a unique attachment. In this case, because $\mathcal{S}$ is directed, they are racing to output the same tile, so the behavior downstream of the center of the macrotile will be the same as if there was no conflict. It is crucial that $\mathcal{S}$ is directed in the aTAM and not just in the syncTAM, as there might be a race between the macrotile blocking its input positions on output sides and said neighbors outputting a glue there. In $\mathcal{S}$ running in the syncTAM, such a race is always won by whichever sets of inputs gets there first, so it does not preclude directedness. In the aTAM on the other hand, the attachment can be delayed until both sets inputs are complete, so by directedness, they must all agree on the same unique output. Because there cannot be a dependency path from an output position to an input position of the same macrotile location, any sets of inputs of a macrotile of $\mathcal{S}'$ which is involved in a conflict between paths must correspond to a competition between attachments which can happen in the aTAM.

If the seed σ of $\mathcal{S}$ consists of a single tile, then the seed σ' for $\mathcal{S}'$ can be created by simply placing the tile $\hat{t}$ that corresponds to the seed tile type t of σ into the center location of the macrotile representing σ . From that, the output paths will grow to the sides of the macrotile, and then (following the argument above) growth of all subsequent macrotiles will correctly match the growth of the tiles of $\mathcal{S}$ by always growing macrotiles that map, under R , to the tile types in the corresponding locations of the terminal assembly $\alpha \in \mathcal{A}_{\square}[\mathcal{S}]$. If instead σ consists of multiple tiles, then σ' simply consists of the corresponding $\hat{t}$ tiles in the centers of the corresponding macrotile locations, all connected by a path of tiles that cross through and block their adjacent input and output positions. Thus, $\mathcal{S}'$ produces a scaled version of the terminal assembly of $\mathcal{S}$ for scale factor s and macrotile representation function R . ◀

B Proof of Theorem 15

► **Theorem 15.** *For each $l > 1$, there exists a shape S_l such that:*

1. *There exists L -syncTAM system $\mathcal{L}_l = (T, \sigma, 2, l)$ such that $\mathcal{L}_l$ strictly self-assembles S_l .*
2. *For all $l' < l$, there exists no L -syncTAM system $\mathcal{L}' = (T', \sigma', \tau, l')$ and scale factor $c \in \mathbb{Z}^+$ such that $\mathcal{L}'$ strictly self-assembles $c \cdot S_l$.*



■ **Figure 7** A depiction of the growth sequence of the initial part of the **cone** portion of the shape S_5 from Figure 5.

Proof. We prove Theorem 15 by first defining the shape S_l and providing an L-syncTAM system that strictly self-assembles it.

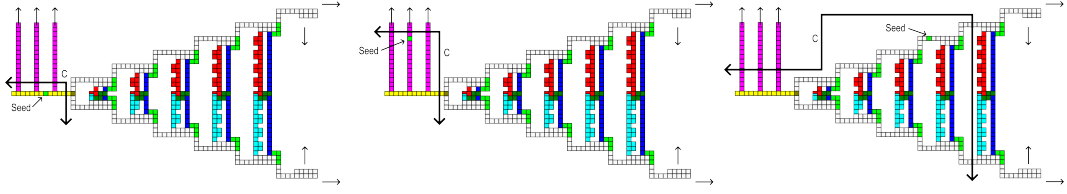
Given some $l > 1$, S_l is an infinite shape consisting of $l - 2$ **arms** that are single-tile-wide columns that each grow upward infinitely, and a **cone** that is a “sideways V” that grows infinitely to the right, with its sides growing further apart and **rungs** periodically growing between the two spreading arms of the V. An initial portion of S_5 (i.e. S_l for $l = 5$) is shown in Figure 5.

The system $\mathcal{L}_l = (T, \sigma, 2, l)$ strictly self-assembles S_l as follows:

1. From the seed, which is a single tile, a set of $4(l - 2)$ tiles grow a row of length $4(l - 2)$ to the right. At each position i to the right of the seed where $(i \bmod 4) == 0$, the yellow tile at that position has a north facing strength-2 glue that attaches to a pink tile type.
2. An infinite number of copies of a single pink tile type attach to the north of the yellow tiles at positions $(i \bmod 4) == 0$ and then to each other, growing $l - 2$ **arms** infinitely upward.
3. To the right of the rightmost yellow tile, a tile of the gold tile type attaches, to initiate growth of the **cone** (whose growth process can be seen in Figure 7).

When growth of S_l begins from the seed, the frontier is initially of size 1. Then, as the yellow row grows and initiates growth of the **arms**, the size of the frontier increases until, when the yellow portion completes, it is $l - 1$ (i.e. one location for each of the $l - 2$ **arms**, and one to the right of the rightmost yellow tile). Then, as the **cone** grows from the gold tile, it consistently has two frontier locations (one on the top half and one on the bottom half) except during the growth of the 4-tile-long green segments in the middle of each **rung**, during which there is 1 frontier location in the **cone**. Thus, once growth of the **cone** has begun, there are always either l frontier locations or $l - 1$ (during the growth of the green segments).

Since there are never more than l frontier locations, the entire frontier can be filled during each assembly step. This ensures that growth of the top and bottom portions of the **cone** can always remain synchronized, and the lengths of the hard-coded sections of the cone are designed so that, even as the rungs grow taller and taller, the red and aqua segments



■ **Figure 8** Examples of the shape S_l from the proof of Theorem 15 being grown in $\mathcal{L}'$ with the seed in different locations and possible cuts (labeled as C in each) used to apply Lemma 11. Note that $\mathcal{L}'$ is allowed to scale S_l to instead self-assemble $c \cdot S_l$, but that doesn't change the location of the cuts or the fact that they still each meet the criteria of Lemma 11 since the scaling only adds a constant amount of thickness across each portion of the cut.

are always guaranteed to collide in the central row. Because of this, the shape S_l is always formed, with bumps extending to the left of the top halves on the left sides of **rungs** and to the right of the bottom halves on the left sides of **rungs**. Therefore, $\mathcal{L}_l$ strictly self-assembles S_l , completing the first portion of the proof.

We now present the second portion of the proof, showing that for all $l' < l$, there exists no L-syncTAM system $\mathcal{L}' = (T', \sigma', \tau, l')$ and scale factor $c \in \mathbb{Z}^+$ such that $\mathcal{L}'$ strictly self-assembles $c \cdot S_l$. We prove this portion by contradiction. Therefore, assume there do exist $l' < l$, $\mathcal{L}' = (T', \sigma', \tau, l')$, and $c \in \mathbb{Z}^+$ such that $\mathcal{L}'$ strictly self-assembles $c \cdot S_l$.

1. $\mathcal{L}'$ is allowed to place its seed anywhere within the shape $c \cdot S_l$. However, since each **arm** and the **cone** are infinite, it must be possible to follow an assembly sequence in $\mathcal{L}'$ that grows portions of each **arm** and the **cone** so that it is possible to make a cut that separates the seed-bearing portion of the assembly from infinite segments of the **arms** and **cone**. (See Figure 8 for examples.)
2. We make a cut C through $c \cdot S_l$ that separates it into sub-shapes $s_0, s_1, \dots, s_{l-1}$, such that s_0 consists of the seed and the yellow segment (plus possibly finite portions of the **arms** and **cone**), s_1 through s_{l-2} each consist of the infinite remainder of one of the **arms** that isn't contained in s_0 , and s_{l-1} consists of the infinite portion of the **cone** that is not contained within s_0 .
3. We now allow $\mathcal{L}'$ to proceed following an arbitrary assembly sequence until the number of steps f , given by Lemma 11 occur. By Lemma 11 it must be the case that, after some assembly step f , each **arm** must always have at least one frontier location, and the **cone** must always have at least one, for a total of at least $l - 1 \geq l'$ frontier locations during all of the subsequent assembly steps (which are infinite in count).
4. From this point on, we choose an assembly sequence, continuing after step f , that proceeds as follows: for every step $f' > f$, a single tile is placed in the **cone**, and $k' - 1$ tiles are placed, each in a unique **arm**.
5. Let g be the number of unique glue types in T' , the tile set of $\mathcal{L}'$. Given the scale factor c at which $\mathcal{L}'$ is self-assembling S_l to achieve $c \cdot S_l$, we let $h = ((g + 1)^{6c} \cdot (6c)! + 1) \cdot 3 + 6$. Justification for this number can be found in the proof of Theorem 1.2 of [21], and it is the count of all possible “window movies” across cuts of the upward or downward growing segments in the locations where they are one (macro)tile wide. This means that this is the maximum possible number of ways that glues can cross a cut across such a segment.
6. We let growth in $\mathcal{L}'$ continue beyond f' until a new period of the **cone** receives its first tile, and then further until the first time after that during which either a red segment grows downward a distance of $2h$ tiles or an aqua segment grows upward a distance of $2h$. This must occur since the **rungs** become arbitrarily tall, and for a **rung** to form, at least one (if not both) of the red and aqua segments must grow at least to the midpoint.

7. We now rewind the assembly sequence until both the red and aqua segments are shorter than $1.5h$ (noting that one of them already may be shorter than that), and at least one is of length $> h$. Since the **cone** is growing by the addition of a single tile during each assembly step, we can apply the Window Movie Lemma (WML) of [21] and the abstract Tile Assembly Model. Without loss of generality, assume that the red segment is of length $> h$. The WML guarantees that since a window movie must have repeated during its growth to the current length, the segment in between the repeated window movies can be “pumped” to create a red segment of arbitrary length (or at least until it crashes into the aqua segment). Since this results in the red segment extending past the midpoint, the shape of the **rung** is invalid in terms of $c \cdot S_l$ with bumps on the left past the midpoint.
8. We can create a new, but valid, assembly sequence that continues to place a single tile in each of $k' - 1$ unique **arms** and a single tile in the red segment of the current period during each step. This results in a **rung** of the shape described, and therefore the self-assembly of a shape that does not match $c \cdot S_l$, which is a contradiction to the statement that $\mathcal{L}'$ strictly self-assembles $c \cdot S_l$. Therefore, no system at synchronicity $l' < l$ can strictly self-assemble S_l at any scale factor, and thus Theorem 15 is proven. ◀

Shape Reachability in Oritatami Is Complete for P

Sota Kano 

The University of Electro-Communications, Tokyo, Japan

Shinnosuke Seki¹  

The University of Electro-Communications, Tokyo, Japan

Abstract

RNA co-transcriptional folding is a process in which an RNA sequence, or *transcript*, folds upon itself while being synthesized nucleotide by nucleotide according to its DNA template. Geary, Rothmund, and Andersen have demonstrated how to program a rectangular tile-like shape into (the DNA template of) an RNA transcript that folds co-transcriptionally into the shape *in vitro*. Using the oritatami model of co-transcriptional folding, we launch the study on the verification of co-transcriptionally folding systems. Shapes are the primary verification target of practical signification; it is abstracted as a set S of points on the 2D triangular grid. Thus, the shape reachability problem asks if the transcript of a given oritatami system goes through all and only the points in S and halts. We demonstrate a logspace reduction from the circuit value problem (CVP), a well-known **P**-complete problem, into a subproblem of the shape reachability whose input oritatami system is promised to be deterministic and at delay 3 (abstraction of the relative speed of local optimization to that of transcription), thus concluding that this subproblem **DSR**(3) is also **P**-complete. What to be verified may specify not only which points to be visited, but also which route should be taken (path reachability), and, moreover, how abstract nucleotides (beads) along the transcript should bind with each other (conformation reachability). We show that as long as the delay is bounded from above by a constant, the conformation reachability can be solved in logspace, and to this problem, path reachability can be reduced in logspace under the promise that a given oritatami system lets beads form as many bonds as possible (maximum arity).

2012 ACM Subject Classification Theory of computation → Complexity classes; Theory of computation → Problems, reductions and completeness; Applied computing → Bioinformatics

Keywords and phrases RNA co-transcriptional folding, Oritatami model, Reachability problems, P-completeness

Digital Object Identifier 10.4230/LIPIcs.DNA.32.3

Supplementary Material *Dataset*: <https://github.com/kabegamikamio/Gates2Oritatami>
archived at `swb:1:dir:4753e8fcffbae60d9d175e40cd82be6ec5b997f3`

Funding *Shinnosuke Seki*: His research is supported in part by Bourse Région Ambition Internationale de Auvergne-Rhône-Alpes, France, No. 00284230-1 (Rokam) and KAKENHI Grant-in-Aid for Scientific Research (C) No. 20K11672.

Acknowledgements We thank Florent Becker for fruitful discussions at the launch of the reported research, and Coda Bourotte for providing insightful comments on the limitations of the Common-rail architecture. Let us take opportunity to express our sincere gratitude towards anonymous referees of **DNA32** for their valuable and encouraging comments to improve the scientific rigor and readability of this manuscript.

¹ Corresponding author



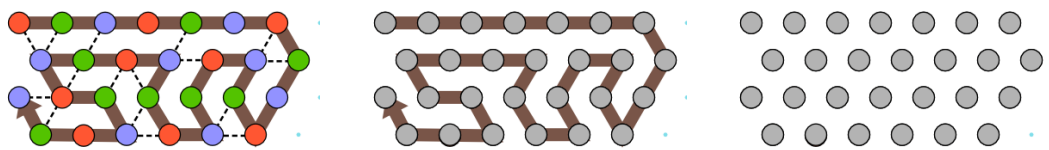


Figure 1 The three types of target considered in the reachability problems: CfR (conformation reachability), PR (path reachability), and SR (shape reachability).

1 Introduction

It is computationally hard to predict thermodynamical folding of RNA [1, 5, 6], wherein an RNA sequence is expected to reach a structure that minimizes free energy, but MFE structures cannot be computed easily unless certain types of non context-free structures, so-called *pseudoknots*, are left out of the picture [23, 27]. RNA sequences also fold isothermally, in particular while being transcribed, that is, synthesized sequentially nucleotide by nucleotide from their DNA templates; the RNA sequences thus synthesized are called *transcripts*. Both this process of *co-transcriptional folding* (CF) and thermodynamical annealing have been engineered as a programming platform for self-assembly of single-stranded RNAs. Annealing is more reliable as of date but seems hardly applicable *in vivo*, where natural RNAs are transcribed and fold to perform various computations, and considerable efforts have been devoted in the last decade to the CF-driven self-assembly of structures and computations [7, 8, 9, 13, 14, 16, 20, 22, 24, 26]. A transcript may not reach its MFE structure co-transcriptionally by being blocked in midway kinetically or topologically. Geary, Rothmund, and Andersen took the initiative in the molecular programming of self-assembly of CF-driven ssRNAs in 2014 [13, 16] by demonstrating how to program a rectangular tile-like shape in a transcript and a sequence of events at the oligonucleic domain-level through which the transcript folds co-transcriptionally into the shape. This methodology has been extended further to 3D wireframe polyhedra *in vitro* [7, 20] and even to arbitrary graphs *in silico/papyro* [8, 24]. Events considered explicitly therein are the hybridization of domains into a helix and the coaxial junction of two helices via a kissing loop (KL) pseudoknot. A target shape is abstracted as a graph G , one of the spanning trees of G is chosen as T , and a transcript is designed such that it traces the contour of T and forms each edge of T as a helix and each *co-edge* (of G but not on T) as a KL. In [20], 5-petal flowers and tetra squares, both of which are 2-dimensional, as well as tetrahedra, have thus been designed and assembled *in vitro* by folding co-transcriptionally and then being annealed afterward. This means that these structures are thermodynamically favored, and they are reachable even solely by co-transcriptional folding according to their AFM images taken prior to post-process annealing. As they are designed so as to fold hierarchically, that is, helices first and then KLs, a certain proportion of successful instances may suggest that global reconfigurations are hardly involved there.

Suppose that the technology of programming a global-reconfiguration-free CF pathway were established. As such a pathway is a coarse-grained specification of events only at the level of oligonucleic domains, a transcript behaves differently depending on what RNA sequences are assigned to its domains. Once fully specified thus at the nucleotide level, the transcript can be verified for an aimed shape efficiently, in linear time in principle. In case of failure, can programmers install a local patch or do they need to overhaul the sequence design? The answer to this question should depend on how much they know about the target shape. If the shape is not connected, a single transcript is certainly not enough; even if it is,

it is computationally hard to test if the shape can be traced, by a Hamiltonian path (see [12]). What if the shape is promised to be traceable, or the programmers are provided with an actual Hamiltonian path? The goal of this paper is to settle these questions in the framework of a reachability test using a formal *oritatami* model of co-transcriptional folding [15]. In this model, a sequence of abstract molecules, or *beads*, folds co-transcriptionally on the triangular grid. This is a discrete-time model; the sequence is transcribed one bead per step, and at every step, only the most nascent δ beads at the end of the thus elongating “transcript” are allowed to reconfigure for more bonds (between adjacent beads), where δ is a system parameter called *delay*. A structure is formalized as a triple of a bead sequence, a self-avoiding path, and a set of bonds between adjacent beads that “like each other,” and called a *conformation*, see Figure 1 (left). Once an oritatami system M is designed, it takes only as many time steps as the length of the transcript of M to see how it folds, or to turn down M because it cannot stabilize a bead uniquely, or *deterministically*. Once the determinism is confirmed, it should be verified if it “reaches” its aimed target (actually, the check of determinism and reachability test can be done in parallel). The system M can be verified for a conformation in logarithmic space, i.e., parallelizable, as long as δ is bounded by a constant (Proposition 6). However, engineers may be informed of the target only partially as noted above, motivating path and shape reachability. Having been promised to be deterministic, M can be tested even for these targets in time exponential in δ but linear in the length of its transcript.

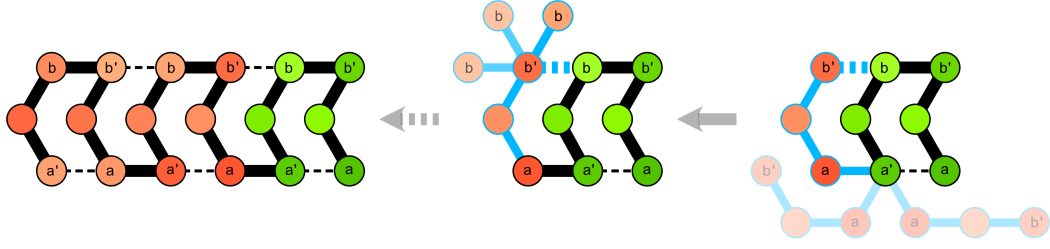
The main contribution of this paper is to prove that the shape reachability problem is **P**-complete; this means that the test is highly unlikely to be parallelizable, even if an input oritatami system is promised to be deterministic and at delay 3 (Theorem 16). The proof consists of demonstrating how to encode in logarithmic space an instance of **CVP** (circuit value problem) [19], which is the most classical **P**-complete problem, primarily as the transcript of an oritatami system to be verified, but also as part of a target shape. This result does not contradict the Turing universality of the oritatami model at delay 3 [25]. Indeed, the point reachability problem is undecidable, but it limits neither time nor space until the transcript reaches a target point, while all the reachability problems studied in this paper delineate the space for computation by an input shape.

2 Preliminaries

Let Σ be a finite set of types of abstract molecules, or *beads*, and let Σ^* be the set of finite sequences of bead types. For a sequence $w = a_1a_2\cdots a_n \in \Sigma^*$ of n beads $a_1, \dots, a_n \in \Sigma$, its length is denoted by $|w|$, i.e., $|w| = n$. We refer to its i -th bead a_i as $w[i]$, and to its subsequence $a_i a_{i+1} \cdots a_j$ as $w[i..j]$; if the subsequence is a prefix of w , that is, $i = 1$, we also use $w[..j]$. This way of referencing is also applied to a self-avoiding path P on the 2-dimensional triangular grid $\mathbb{T} = (\mathbb{Z}^2, \sim)$, where $p \sim q$ means that the grid points p, q are next to each other, that is, $p - q$ is in $\hat{V} = \{(1, 0), (0, 1), (-1, 1), (-1, 0), (0, -1), (1, -1)\}$, which is the set of the six unit vectors of $\mathbb{T}$.

Oritatami [15] is a discrete-time model of co-transcriptional folding on the triangular grid. In the model, a sequence of bead types is transcribed, that is, elongates one bead per step, and folds on the triangular grid in such a way that at every step, only the most nascent δ beads are allowed to reconfigure so as to form as many bonds as possible between adjacent beads without breaking the transcript’s (covalent) backbone or having the transcript intersect with itself, where δ is a parameter of an oritatami system, and called *delay*. Another system parameter called (*binding*) *rule* $\heartsuit$ specifies which types of beads are allowed to bind with

3:4 Shape Reachability in Oritatami Is Complete for P



■ **Figure 2** A glider and its folding at delay 3. Its transcript is of period 6 as a $a \bullet b' b \bullet a' a \bullet \dots$ and its rule is $a \heartsuit a', b \heartsuit b'$.

each other as long as they are at unit distance. That is, $\heartsuit$ is a symmetric relation on Σ . A *(valid) conformation (with respect to $\heartsuit$)* is a triple $c = (w, P, H)$ of a self-avoiding path P on $\mathbb{T}$, a sequence w of as many beads as the points along P , and a set H of pairs of bead indices representing a bond, that is:

$$H \subseteq \{ \{i, j\} \mid P[i] \sim P[j], 1 \leq i, i+2 \leq j < |w|, w[i] \heartsuit w[j] \}.$$

Here, the inequality $i+2 \leq j$ implies that a bond cannot form between two consecutive beads along the transcript, which are bonded covalently rather. By the *length* of c , we mean $|w| (= |P|)$, and denote it by $|c|$. The set of all valid conformations w.r.t. $\heartsuit$ is denoted by $\mathcal{C}_{\heartsuit}$. In the following, we always clarify which rule $\heartsuit$ is under consideration and do not bother specifying with respect to which rule a conformation under consideration is valid. The stability of c is measured as $E(c) = -|H|$; that is, the more bonds a conformation has, the more stable it becomes. Its *arity* is the maximum number of bonds per bead, and formally defined as

$$\max_{1 \leq k \leq |w|} |\{ \{i, j\} \in H \mid i = k \text{ or } j = k \}|.$$

A conformation is said to be of *maximum arity* if its arity is equal to the maximum degree of the underlying grid. The set of all conformations of arity at most α is denoted by $\mathcal{C}_{\heartsuit, \leq \alpha}$, or simply by $\mathcal{C}_{\leq \alpha}$.

Let us introduce two operations between conformations. One is to take a *prefix* of a conformation $c = (w, P, H)$; this operation keeps only a prefix w' of w in the classical sense, leaves the same number of points at the beginning of P , and extracts from H only the bonds between beads on the prefix. The other is elongation. Elongating c by a bead $b \in \Sigma$ yields the following set of conformations:

$$c^{\triangleright b} = \left\{ (wb, Pv, H \cup H') \in \mathcal{C}_{\heartsuit} \mid H' \subseteq \{ \{i, |w|+1\} \mid P[i] \sim v, i \leq |w|-1, w[i] \heartsuit b \} \right\}.$$

For elongating a conformation c rather by a sequence of bead types $b_1 b_2 \dots b_n$, it suffices to first elongate c by the first bead b_1 , then elongate each of the resulting conformations, that is, those in $c^{\triangleright b_1}$, by b_2 , and so on.

An *oritatami system* M is a 4-tuple $M = (t, \heartsuit, \delta, \alpha)$ of a transcript $t \in \Sigma^*$, a rule $\heartsuit$, and two integer parameters δ (delay) and α (arity). An input is given as a *seed* conformation σ of arity at most α . Its computation begins with such a seed, $c_0 = \sigma$, and consists of conformations $c_1, c_2, \dots$ that satisfy:

$$c_i \in \operatorname{argmin}_{c \in c_{i-1}^{\triangleright t[i]} \cap \mathcal{C}_{\leq \alpha}} \left(\min_{c' \in c^{\triangleright t[i+1 \dots i+\delta-1]} \cap \mathcal{C}_{\leq \alpha}} E(c') \right). \quad (1)$$

The i -th bead $t[i]$ is thus stabilized, according to c_i that satisfies (1) (see in Figure 2 how a bead is thus stabilized). With more than one candidate for c_i , M is *nondeterministic on the seed* σ . On the other hand, if for all time steps $i \leq |t|$, c_i is uniquely determined or there is no candidate, M is *deterministic on* σ . Particularly, the lack of candidate aborts M ; this happens when all the neighbors of $t[i-1]$ have been already occupied. Otherwise, the system fully transcribes and stabilizes t as $c_{|t|}$, and this *terminal* conformation serves as its outcome on the input σ . If an oritatami system gets nondeterministic on the seed σ , it may have multiple terminal conformations; let us denote their set by $T(M, \sigma)$; this notation will be abused to refer to the unique terminal conformation if it is clear from the context that M is deterministic and does not abort on σ .

The bead stabilization (1) is local in the sense that it is affected by only at most 6δ beads inside the circle with radius $\delta + 1$ centered around $t[i-1]$, which was already stabilized at the previous step. Let us informally call this region the $(i\text{-th})$ *universe*. The circumference of the circle is too far for nascent beads to be placed, but a bead there can affect the bead stabilization by attracting the nascent segment $t[i..i + \delta - 1]$ unless it has already formed α bonds.

When an oritatami system $M = (t, \heartsuit, \delta, \alpha)$ is given as input, the amount of space to store it matters. If all beads along t are pairwise distinct, then the rule $\heartsuit$ can contain $|\Sigma|^2$ elements; however, this is unrealistic and it is more realistic to treat the size of Σ as a constant. Since δ larger than $|t|$ makes no mathematical sense and α is bounded by the maximum degree of the underlying grid (6 for the triangular grid), when Σ is fixed, M can be stored within $O(|t|)$ bits.

2.1 Parallel feasibility: P-complete vs NC

Class **P** is the set of decision problems that can be solved with a deterministic Turing machine in polynomial time. It is regarded as a class of practically feasible problems.

Nick's class, in short **NC**, is a subset of **P** and consists of the problems which can be solved with a deterministic parallel Turing machine with a polynomial number of processors in polylogarithmic time. Problems in **NC** are regarded as *efficiently parallelised*. L is the class of problems that are computable in logarithmic space. These problems can also be efficiently parallelised. For these complexity classes and Log-space reduction, see [2].

► **Lemma 1** ([3, 17]). $L \subseteq NC$.

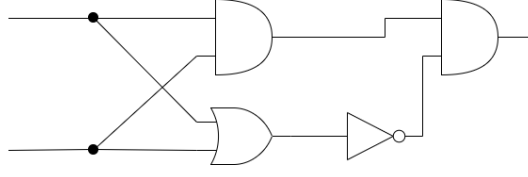
A problem is **P**-complete if it is in **P** and any problem in **P** can be reduced to it with an appropriate reduction such that the reduction itself is efficiently parallelised. **P**-complete problems are believed to be *inherently sequential*; thus, there may not be efficient parallel algorithms to solve them.

2.2 Circuit value problem

The circuit value problem, **CVP**, is the problem to decide the terminal output of a given Boolean circuit consisting of gates for inputs, AND, OR and NOT operations. A **CVP** instance is shown in Figure 3. It is formally defined as follows.

► **Definition 2** (**CVP**).

- Input: a Boolean circuit with $n+1$ gates $G = (g_0, g_1, \dots, g_n)$, where each g_i ($0 \leq i \leq n$) is either an input or a gate of AND, OR or NOT.
- Output: yes if the terminal output g_n is true.



■ **Figure 3** An example of CVP.

► **Lemma 3** ([19]). *CVP is P-complete.*

Just as the SAT is often used to prove NP-completeness, CVP is the standard source for proving P-completeness.

2.3 Reachability problems in Oritatami

Reachability problems in the oritatami model in general ask: given an oritatami system $M = (t, \heartsuit, \delta, \alpha)$, a seed σ , and a “target” as input, are terminal conformations in $T(M, \sigma)$ equal to the target in a problem-dependent sense? As a target, we shall consider conformations, paths, and shapes, which are a set of grid points. Let us first settle the issue of input encoding. Assume that the alphabet Σ is fixed, and recall first that M can be encoded in linear space in $|t|$. A target conformation $c = (w, P, H)$, with $|w| = n$, can be encoded in $O(n)$ bits. Indeed, P consists of as many points as beads in w , its first bead can be placed at $(0, 0)$ without loss of generality, and the rest of P can be rather described as a sequence of $n - 1$ vectors $v_1, v_2, \dots, v_{n-1}$ taken from $\hat{V}$ such that $P[i] = (0, 0) + v_1 + \dots + v_{i-1}$. The set H can contain at most $6n$ bonds, and each of them can be encoded in $\lceil \log 6 \rceil = 3$ bits as an index for the six possible unit vectors on the triangular grid. Lastly, as the seed σ must be a prefix of c (otherwise, how can M reach c , starting from σ ?), it can be encoded as compactly as c . In order for $T(M, \sigma) = C$ to hold, the length of the seed plus $|t|$ must be equal to n . Therefore, the whole input can be encoded in $O(n)$ space. All alternative targets considered in this paper can be obtained by somehow simplifying conformations so that in all problems formalized from now on, the size of an input is $O(n)$.

Logarithmic space in n is enough to retrieve from the above encoding of the conformation c the labels of all the beads next to the i -th point of P . The index $i \leq n$ can be stored in $\lceil \log n \rceil$ bits. The processor computes all the indices $1 \leq m < n$ such that $\sum_{j=i}^m v_j \in \hat{V}$; then the bead $w[j]$ is placed at a point next to $P[i]$; note that there exist at most 6 such j 's. In this way, the processor can even retrieve in $O(\log n)$ space all the pieces of information about the i -th universe necessary and sufficient to decide how the next bead $t[i]$ should be stabilized.

If a conformation is given as a target (CfR in Definition 4), it uniquely determines all intermediate conformations $c_1, c_2, \dots$, and requires M to be deterministic on the seed c_0 . For the other reachability problems (PR and SR in Definitions 7 and 13), it makes at least mathematical sense to formalize their deterministic variants that promise that a given oritatami system folds deterministically on a given seed. It can be tested in $O(6^\delta \cdot |t|)$ time whether M is deterministic or not on a given seed; for $1 \leq i \leq |t|$, the search for c_i considers at most 6^δ elongations of c_{i-1} and if the size of the right-hand side of (1) is 2 or greater, then M is not deterministic.

2.3.1 Conformation reachability (CfR)

Let us begin with the following reachability problem whose target is a conformation.

► **Definition 4** (CfR).

- Input: an oritatami system M , a seed σ , and a conformation c .
- Output: yes if $T(M, \sigma) = c$.

The $O(6^\delta \cdot |t|)$ -time determinism test mentioned above actually simulates the folding by M fully, and the only thing required to solve CfR is to verify that the simulated terminal conformation is equal to c .

► **Proposition 5.** *CfR can be solved in sequential time of $O(6^\delta \cdot n)$; hence, it is fixed-parameter-linear.*

CfR can be solved efficiently in parallel within the class of delay- δ oritatami systems for a fixed δ . Let $\text{CfR}(\delta)$ be the set of CfR instances whose input oritatami system is of delay at most δ ; we will denote likewise the subproblem of other reachability problems by bounding the delay of input oritatami systems by a constant δ . In light of Lemma 1, it suffices to demonstrate how a processor simulates the stabilization of $t[i]$ in the logarithmic space in n . As explained earlier, the i -th universe can be retrieved from the encoding of the target conformation c using $O(\log n)$ space, and can be stored in space $O(6\delta \cdot \log n)$. This universe enables the processor to decide how $t[i]$ should be stabilized. The amount of bits in use remain logarithmic in n as long as δ is bounded by a constant.

► **Proposition 6.** *For all integer δ , $\text{CfR}(\delta)$ is in $\mathbf{L}$.*

2.3.2 Path reachability (PR)

► **Definition 7** (PR).

- Input: An oritatami system M , a seed σ , and a path P .
- Output: Yes if the path of *all* terminal conformations is equal to P .

Unlike the conformation, an oritatami system does not have to follow a path deterministically. If an oritatami system given as input is promised to be deterministic, simulating one run of M is enough; during the run, it suffices to check if $t[i]$ is stabilized at the point $P[i]$ for all time steps $i \leq n$. Let us formalize the deterministic variant of PR as follows:

► **Definition 8** (DPR).

- Input: An oritatami system M and a seed σ such that $|T(M, \sigma)| = 1$, along with a path P .
- Output: Yes if the path of the *unique* terminal conformation is equal to P .

► **Proposition 9.** *DPR can be solved in sequential time of $O(6^\delta \cdot n)$.*

As shown in the conformation reachability, within the class of delay- δ oritatami systems for a fixed δ , this problem can be solved using logarithmic space; let $\text{DPR}(\delta)$ be the set of DPR instances whose input oritatami system is of delay at most δ .

► **Proposition 10.** *For all $\delta \geq 1$, $\text{DPR}(\delta)$ is in $\mathbf{L}$.*

There may exist a nondeterministic oritatami system which always follows the same path, but with different binding patterns. The problem PR is thus motivated at least theoretically. Leaving its computational complexity as an open problem, here we show that an instance of PR with a maximum-arity oritatami system can be reduced to an instance of CfR, and hence, can be solved efficiently in parallel. Let us denote the set of such instances of PR by $\text{PR}_{\max \alpha}$. It makes no sense for a maximum-arity oritatami system to prevent $\heartsuit$ -related beads from

forming a bond. This observation implies that if the system reaches the path P , its terminal conformation is the one obtained from P by binding all the beads that are $\heartsuit$ -related and adjacent in P .

► **Proposition 11.** *For all $\delta \geq 1$, $\text{PR}_{\max \alpha}(\delta) \in \mathbf{L}$.*

Proof. It suffices to reduce an instance of $\text{PR}_{\max \alpha}(\delta)$ to an instance of $\text{CfR}(\delta)$. Let P be the target path and t be the transcript of the input oritatami system with $|t| = n$. For $i \leq n$, the i -th point of P should be labeled with $t[i]$, and this labeling can be done by storing the index i for reference using $\lfloor \log n \rfloor$ bits. In this way, we can determine which type of bead is supposed to stabilize at each point along P . Moreover, it can be determined uniquely how they are supposed to bind with each other because the input oritatami system is promised to be of maximum arity. Thus, an instance of $\text{CfR}(\delta)$ has been obtained. ◀

What if the arity of a given oritatami system is not maximum? In this realistic scenario, once a bead exhausts its capability of forming α bonds, it becomes an obstacle that does not contribute to lowering the free energy, at least by binding, but stands in the way of the transcript. Therefore, it seems necessary to track and memorize how the prefix $t[1..i-1]$ has folded co-transcriptionally so far in order to decide if each bead in the i -th universe is still active or already inert.

This problem is at least in **coNP**. Indeed, if the oritatami system of a given instance can follow another path to the target one, that path serves as a polynomial-size no certificate. It suffices to run the system and ensure that it can follow that path.

► **Proposition 12.** *$\text{PR}(\delta)$ is in **coNP** for all $\delta \geq 1$.*

2.3.3 Shape reachability (SR)

A *shape* is a set of points on $\mathbb{T}$. Given a conformation $c = (w, P, H)$, its shape is a set of points along the path P .

► **Definition 13 (SR).**

- Input: an oritatami system M , a seed σ , and a shape S .
- Output: yes if the shape of *all* the terminal conformations in $T(M, \sigma)$ is equal to S modulo translation.

Note that in this definition, the output verification allows for translation. This is because unlike the conformation or path reachability, the input shape S does not specify where the seed should be. As done for path reachability, here we define the deterministic variant of shape reachability (DSR) by requiring an input oritatami system M and a seed σ to satisfy $|T(M, \sigma)| = 1$.

As done for Proposition 12, we can show that without promise of determinism, the shape reachability problem is in **coNP** under any constant bound on the delay.

► **Proposition 14.** *For all $\delta \geq 1$, $\text{SR}(\delta)$ is in **coNP**.*

In contrast, if a given oritatami system is promised to fold deterministically on a given seed, then the shape reachability can be solved in a quadratic time in n . Based on Proposition 9 on DPR, it should suffice to note that there are $O(n)$ possible arrangements of the seed in order to prove the following result.

► **Proposition 15.** *DSR can be solved in sequential time of $O(6^\delta \cdot n^2)$. Thus, for all $\delta \geq 1$, $\text{DSR}(\delta)$ is in **P**.*

the current count and a carry; independent of the underlying computation, the counter always folds into the same shape (from the *global* bird’s-eye view, one can read the computation by measuring the number of zigzags between two consecutive overflows, where zigzags get wider for 1 extra bit). The novel, programmer-friendly geometry-driven architectures in oritatami for computation [11, 25] do not serve the proof of Theorem 16. Since the hardness of SR appears to stem from the “stealthiness” of intermediate values of computation, simulations of those systems might be efficiently parallelisable.

The primary challenges in the proposed construction include:

- designing new modules to simulate Boolean circuits that conceal interim calculations, unlike “too chatty” existing systems;
- implementing 3D-like wiring crossovers within the 2D oritatami model; and
- developing a reduction algorithm that determines the optimal arrangement of modules and wiring within logarithmic space.

While our gate modules are largely original, they are inspired by the half-adder that intrinsically computes AND operation [10]. As detailed later, this module computes through its folding path rather than its shape. In addition, we developed a crossover module to facilitate multi-level interchanges of wiring. Finally, we propose the Common-rail architecture, which decouples input propagation from logical computation. This enables selective routing of values from the input-carrying rail to individual gates and ensures that the entire reduction is doable in logarithmic space, as the arrangement of modules is done by simple computations in this architecture.

3.1 Common-rail architecture

While the reduction algorithm we shall show in Section 4 must place gate and wiring modules at appropriate positions in the system, determining an optimal layout is challenging due to complex inter-dependencies between modules. To overcome this, we have devised a redundant but systematic architecture called the *common-rail architecture* (Figure 4), where all gate modules are placed diagonally on the basal plate of the system. In this architecture, starting from the seed positioned at the very top, the transcript macroscopically folds into zigzags and proceeds downward, being guided by switch-back modules, which were rather called left and right-turn modules in the zigzag counter [10, 21] from a zig to the next zag and from a zag to the next zig. The transcript is a chain of modules, a functional unit, all of which but the ones for switchback fold into a chevron-like shape of width 6 and height 3 (this is defined as the set of points occupied by the glider depicted in Figure 2). At its end, a terminal module reports the binary output of the circuit by shape.

This architecture has two types of wiring: *common rails* and *lateral rails*. Common rails run vertically from the seed to the bottom of the system, and each of them is dedicated to holding and conveying either an input encoded in the seed or the output of a specific gate. They are implemented by vertical wire modules piled on top of each other. In order to avoid crosstalk, gliders are intercalated in-between. Let us bundle a pillar of vertical wire modules and that of “insulators” together and call it a *column*. The i -th gate g_i , which is an input, AND, OR, or NOT, is placed inside the column i . Lateral rails are horizontal wires branched from common rails to deliver their values through a zig (of height 3) leftward to gates. All the modules along a zig are based on the glider, which is capable of propagating 1-bit of information as the position of its first bead: top for false and bottom for true, and are designed so as to always start and end at the same height, inheriting this capability of propagating 1-bit. Therefore, one logical gate is collaboratively accommodated in two

continuous zigzags to wire common rails into at most two input pins; let us call these zigzags a *row*. As illustrated in Figure 4, the i -th row, which consists of the $(2i-1)$ -th and $2i$ -th zigzags, takes responsibility for the computation by the i -th logical gate. The first input to the gate goes through the upper lateral rail, turns downward, and is fed to the gate from above, while the second input (only for AND and OR) goes through the lower lateral rail and is fed to the gate from the side. Unlike common rails, a lateral rail is placed only when necessary and deployed solely between its source common rail and the destination gate, with no further branching. Let us define a *gate cell* as a structural unit positioned at the intersection of each row and column.

This common-rail architecture enables the reduction algorithm to appropriately place modules with simple conditional branches, ensuring that it can run in logarithmic space.

3.2 Gadget construction for Boolean circuits

The basic design and specification of the modules were redesigned based on the half-adder for counting [10]. We have designed the following five categories of modules:

- gate modules (AND, OR, NOT)
- wire modules
- padding / absorber modules
- the switch-back module
- the terminal module

All but the terminal module fold into a 6-bead width and 3-bead height chevron shape.

Gate modules

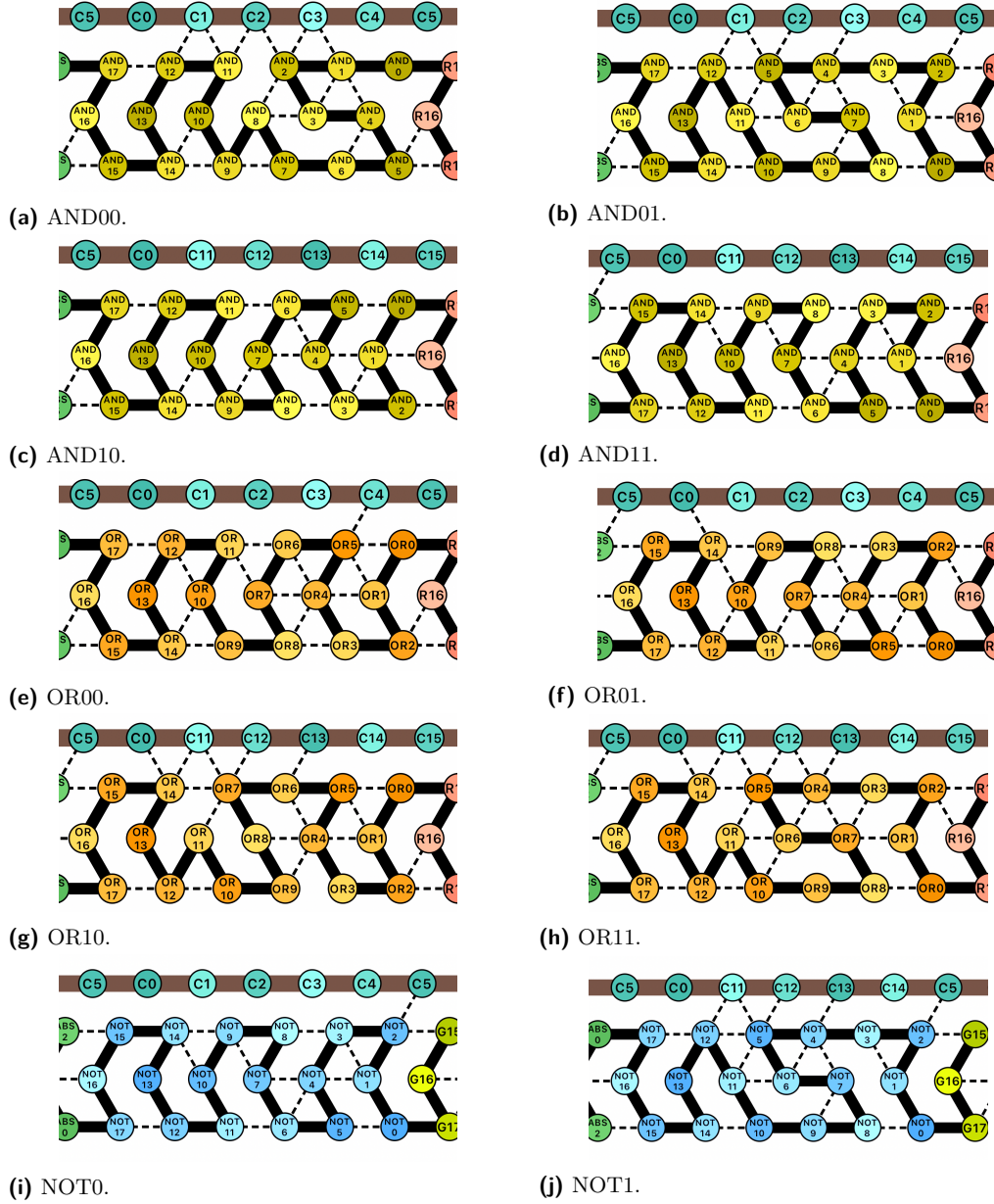
AND and OR gates (Figure 5 (a-d) and (e-h)) receive their first input from above, encoded as how the gate binds with the lateral rail above, while their second input is propagated from the previous module as the starting position of the folding process as: *top* representing false or *bottom* representing true. NOT gate (Figure 5 (i, j)) receives its sole input from above. Although the output specifications vary slightly between modules, they all share two common features: the output value is represented by the presence or absence of bonds between the gate and the common rail below, and the folding process terminates at the bottom position if the output is true, and otherwise at the top.

Wiring

There are five types of wiring in this system:

- vertical wires for common rails
- horizontal wires for lateral rails
- crossovers
- 1-to-2 fan-out wires
- L-shaped wires

Vertical wires (Figure 6) constitute the common rails by being stacked on top of each other. They propagate 1-bit of information downward as absence or presence of bonds between them (no bond means true). Compare Figure 6 (a) with (b) and observe that when receiving false from above, it ends at the top (interpreted as false through lateral rails), or it ends at the bottom otherwise. This property enables this module to be also used to branch a lateral rail from a common rail. It suffices to transcribe a glider after this module; the 1-to-2 fan-out is implemented in this way. Non-bifurcation simple vertical wires can be



■ **Figure 5** All conformations of the AND, OR, and NOT gate modules. (a-d) All four conformations of the AND module on respective four inputs $(x, y) \in \{(0, 0), (0, 1), (1, 0), (1, 1)\}$, where x comes from above while y comes from right. AND_{xy} represents the conformation that computes $x \wedge y$. For instance, AND_{01} is the conformation of the computation $0 \wedge 1 (= 0)$. (e-h) All four conformations of the OR module on (x, y) . (i, j) The two conformations of the NOT module on the respective inputs 0 and 1.

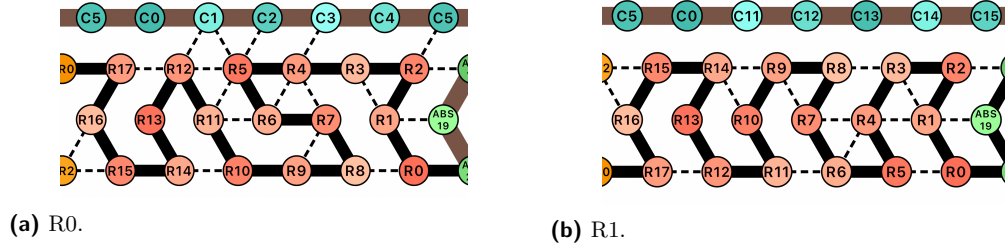


Figure 6 The two conformations of the vertical wire module on the respective two inputs 0 and 1. Note that the vertical wire module extends along the zig-zag path horizontally while it transmits a value vertically.

implemented merely by transcribing an auxiliary absorber module (explained shortly), which immediately cancels out a horizontal top/bottom signal by ending at the bottom no matter what. In fact, this module is the H0c and H1c components of the half-adder in [10]. Due to design constraints of intermodular binding rules, the system involves four “doppelgängers” (R, PA, PB and PC) of vertical wires, which are implemented identically but out of four pairwise-disjoint sets of bead types.

Horizontal wires are just a conventional glider (Figure 2), propagating 1-bit of information as the position of its first bead: top for false and bottom for true.

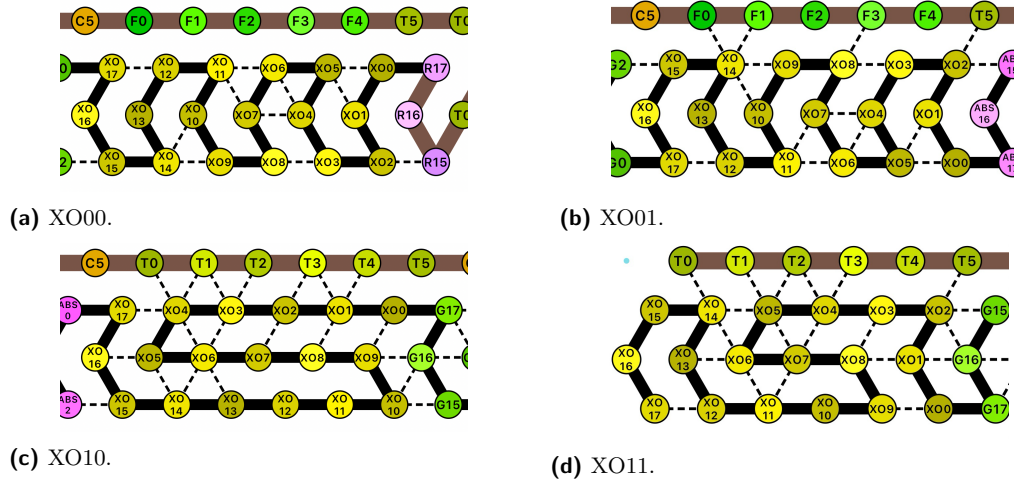


Figure 7 The four conformations of the crossover (XO) module on the respective four inputs $(x, y) \in \{(0, 0), (0, 1), (1, 0), (1, 1)\}$.

Crossover (XO) modules (Figure 7) are placed at the intersections of common and lateral rails to prevent signal collisions. Instead of physically crossing the wires in 3D space, this module achieves a “pseudo-crossover” by folding into one of the four possible conformations depending on the four possible inputs. Adjacent modules then interpret these motifs to retrieve the encoded values for each rail.

Finally, the L-shaped wiring module redirects a signal propagated horizontally downward at the end of a lateral rail in order to be fed to a gate below. While the structure of this module is identical to that of a horizontal wire, the value is interpreted and propagated to the gate by the vertical wire positioned directly beneath it.

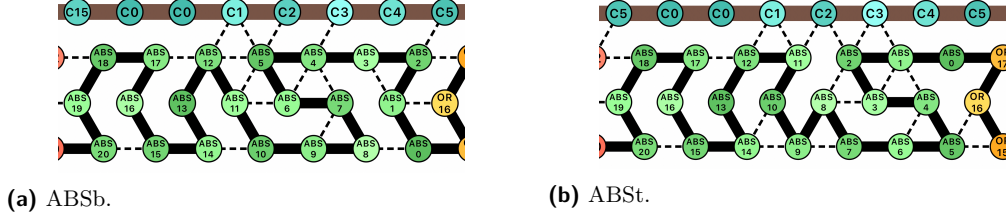


Figure 8 The two conformations of the absorber module, both of which end at the bottom, no matter whether they have started (a) at the bottom or (b) at the top.

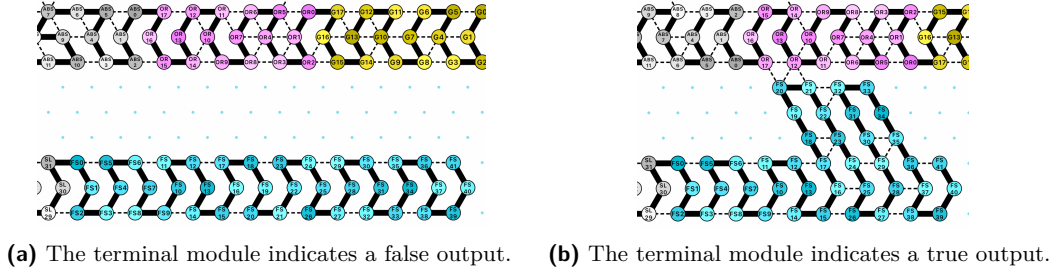


Figure 9 The terminal module and its output conformations. The module translates the final Boolean result into a geometric shape: (a) a horizontal extension represents false, while (b) an upward protrusion represents true. This distinct geometric change at the system's tail enables the encoding of the computation result as a Shape Reachability instance.

Auxiliary modules

The system involves modules that do not directly take part in logical computations but rather ensure the correct geometric growth, enable gates and wiring modules to function properly, and simplify their design. Gliders are used as a padding module to fill the vacant area, which can be found, for instance, to the left of the gates in each row. Absorbers (Figure 8) end at the bottom no matter whether it has started at the top or at the bottom. They are placed at the end of a lateral rail and cancel out the 1-bit of information having propagated therein. They also prevent interference between the computations of gates or wiring modules in neighbouring columns. The switch-back module is positioned at the end of each row to reverse the macroscopic folding direction (from zig to zag, and vice versa).

Terminal module

This is the only module that takes (two) different shapes. As shown in Figure 9, at the end of the system, it protrudes upward if and only if the terminal output of the Boolean circuit is true. As the computation has been fully concealed so far, without this module, the system would yield the unique shape no matter what the seed is, collapsing the reduction.

4 Logspace reduction from CVP to SR

Algorithm 1 receives a Boolean circuit $G = (g_0, g_1, \dots, g_n)$ as input and generates a seed (C_0 in the pseudocode) in Phase 1 and a transcript (T) in Phase 2 both sequentially bit-by-bit. Storing only counters up to n , this algorithm can be executed in $O(\log n)$ space.

■ **Algorithm 1** Logspace construction of seed and transcript from a gate sequence.

Input: A sequence of gates G of length n
Output: Seed C_0 and transcript T

```

1   $C_0 \leftarrow \emptyset, T \leftarrow \emptyset, n_{\text{input}} \leftarrow 0;$ 
   /* Phase 1: Construct seed configuration */
2  for  $i \leftarrow 0$  to  $n - 1$  do
3       $g \leftarrow G_i;$ 
4      if  $g.type = INPUT$  then
5           $n_{\text{input}} \leftarrow n_{\text{input}} + 1;$ 
6          if  $g.value = true$  then
7               $C_0.addFirst(C_t);$ 
8          else
9               $C_0.addFirst(C_f);$ 
10     else
11          $C_0.addFirst(C_{\text{none}});$ 
12  $C_0.add(c_{\text{start}});$ 
   /* Phase 2: Construct transcript */
13 for  $y \leftarrow n_{\text{input}}$  to  $n - 1$  do
14      $g \leftarrow G_y, l \leftarrow \text{left\_index}(g);$ 
   /* First rail: zig path */
15     for  $x \leftarrow 0$  to  $n - 1$  do
16         case  $x = l$  do
17              $T.add(R), T.add(G);$  /* Branch from common to lateral */
18         case  $x = y$  do
19              $T.add(G), T.add(ABS);$  /* Input to gate */
20         case  $x < l$  do
21              $T.add(PA), T.add(ABS);$  /* Common rail */
22         case  $x > y$  do
23              $T.add(G), T.add(ABS);$  /* Glider fill */
24         otherwise do
25              $T.add(XO), T.add(G);$  /* Lateral rail */
26      $T.add(SW);$ 
   /* First rail: zag path */
27     for  $x \leftarrow 0$  to  $n - 1$  do
28          $T.add(ABZ), T.add(PB);$ 
29      $T.add(SB);$ 
   /* Second rail: zig path */
30      $r \leftarrow \text{right\_index}(g);$ 
31     for  $x \leftarrow 0$  to  $n - 1$  do
32         if  $x = y$  then
33              $T.add(g.type), T.add(ABS);$ 
34         else if  $x = j$  then
35              $T.add(PC), T.add(ABS);$ 
36         else if  $g.type \neq NOT \vee x < r$  then
37              $T.add(R), T.add(ABS);$ 
38         else if  $x > y$  then
39              $T.add(G), T.add(ABS);$ 
40         else
41              $T.add(XO), T.add(G);$ 
42     if  $y = n - 1$  then
43          $T.add(SBL);$ 
44         return  $(C_0, T);$ 
45     else
46          $T.add(SB);$ 
   /* Second rail: zag path */
47     for  $x \leftarrow 0$  to  $n - 1$  do
48          $T.add(ABZ), T.add(PC);$ 
49      $T.add(SB);$ 

```

The circuit G is given as a comma-separated sequence of gate definitions enclosed in parentheses. While the gate indices are not explicitly labelled to these brackets, the $(i+1)$ -th element in the sequence uniquely defines gate g_i . The encoding rules are as follows:

- If g_i is an input gate, it is represented as `(true)` or `(false)`.
- If $g_i = g_k \wedge g_j$, `(AND, k, j)`.
- If $g_i = g_k \vee g_j$, `(OR, k, j)`.
- If $g_i = \neg g_k$, `(NOT, k)`.

In Phase 2, a transcript is generated sequentially by concatenating the module specified by the corresponding row and column indices according to the common-rail architecture (Figure 4). For the i -th row, the length of the lateral rails in its first and second zigs are determined by checking from which common rail(s) the gate g_i should be fed with its inputs, while the zags can be actually even hardcoded. The transcript is completed by appending a terminal module at the end.

An executable Java program of this algorithm is shared on <https://github.com/kabegamikamio/Gates2Oritatami>.

5 Concluding remarks

In this paper, we have formulated the conformation, path, and shape reachability problems in the oritatami model, and proved upper and lower bounds on their computational complexity under the assumption that the delay δ be bounded by a constant. In order to prove the **P**-completeness of the deterministic shape reachability problem, we developed a suite of logic gates (AND, OR, NOT) and supplementary modules such as wires and crossovers and demonstrated how to combine them into a delay-3 deterministic oritatami CVP solver in a logarithmic space. With a mechanism to guess Boolean values nondeterministically (see [18]), wiring the proposed logic gates in the proposed Common-rail architecture should enable us to design a delay-3 nondeterministic oritatami UNSAT solver, which would test a hardcoded Boolean formula according to a guessed assignment. If such a solver could be assembled from a given UNSAT instance in a logarithmic space, then $\text{PR}(3)$ would be proved to be **coNP**-complete; recall that $\text{PR}(3) \in \text{coNP}$ (Proposition 12).

The length of a transcript is actually bounded by that of its DNA template, and there is a technical limit on the synthesizable DNA templates. Geary and Andersen have unveiled the idea of transcribing a periodic RNA transcript from a circular DNA template [13]. Most computational oritatami systems adopt their idea and utilize a periodic transcript [11, 15, 21, 25]. Once the length of the period of a transcript is bounded by a constant, a circuit cannot be hardcoded entirely in a transcript anymore, but needs to be also encoded on a seed and retrieved co-transcriptionally.

We have shown how to provide a polynomial-size no-certificate for the path and shape reachability problems (without the promise of determinism), proving that they are in **coNP**. Aside from the computational complexity, it should be interesting to imagine a path or a shape that is intrinsically nondeterministic, that is, cannot be reached by any deterministic oritatami system, but there exists a nondeterministic one all of whose terminal conformations follow the path/are of the shape (see [4] for an analog of this research direction in the abstract tile assembly model).

All these research directions should also be pursued in a more realistic model of RNA co-transcriptional folding.

References

- 1 Tatsuya Akutsu. Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete Applied Mathematics*, 104(1-3):45–62, 2000. doi:10.1016/S0166-218X(00)00186-4.
- 2 Sanjeev Arora and Boaz Barak. *Computational Complexity. A Modern Approach*. Cambridge University Press, 2009.
- 3 Allan Borodin. On relating time and space to size and depth. *SIAM Journal on Computing*, 6(4):733–744, 1977. doi:10.1137/0206054.
- 4 Nathaniel Bryans, Ehsan Chiniforooshan, David Doty, Lila Kari, and Shinnosuke Seki. The power of nondeterminism in self-assembly. *Theory of Computing*, 9:1–29, 2013. doi:10.4086/TOC.2013.V009A001.
- 5 Anne Condon, Monir Hajiaghayi, and Chris Thachuk. Predicting minimum free energy structures of multi-stranded nucleic acid complexes is APX-hard. In *Proceedings of the 27th International Conference on DNA Computing and Molecular Programming*, volume 205 of *LIPIcs*, pages 9:1–9:21. Schloss Dagstuhl, 2021. doi:10.4230/LIPIcs.DNA.27.9.
- 6 Gwendal Ducloz, Ahmed Shalaby, and Damien Woods. Algorithmic hardness of the partition function for nucleic acid strands. In *Proceedings of the 31st International Conference on DNA Computing and Molecular Programming (DNA31)*, volume 347 of *LIPIcs*, pages 1:1–1:23. Schloss Dagstuhl, 2025. doi:10.4230/LIPIcs.DNA.31.1.
- 7 Antti Elonen, Ashwin Karthick Natarajan, Ibuki Kawamata, Lukas Oesinghaus, Abdulmelik Mohammed, Jani Seitsonen, Yuki Suzuki, Friedrich C. Simmel, Anton Kuzyk, and Pekka Orponen. Algorithm design of 3D wireframe RNA polyhedra. *ACS Nano*, 16:16608–16616, 2022.
- 8 Antti Elonen and Pekka Orponen. Designing 3D RNA origami nanostructures with a minimum number of kissing loops. In *Proceedings of the 31st International Conference on DNA Computing and Molecular Programming (DNA30)*, volume 314 of *LIPIcs*, pages 4:1–4:12. Schloss Dagstuhl, 2024. doi:10.4230/LIPIcs.DNA.30.4.
- 9 Antti Elonen, Leon Wimbes, Abdulmelik Mohammed, and Pekka Orponen. Dnaforge: A design tool for nucleic acid wireframe nanostructures. *Nucleic Acids Research*, 52(W1):W13–18, 2024.
- 10 Szilárd Zsolt Fazekas, Hwee Kim, Ryuichi Matsuoka, Shinnosuke Seki, and Hinano Takeuchi. On algorithmic self-assembly of squares by co-transcriptional folding. In *Proceedings of the 33rd International Symposium on Algorithms and Computation (ISAAC 2022)*, volume 248 of *LIPIcs*, pages 37:1–37:15. Schloss Dagstuhl, 2022. doi:10.4230/LIPIcs.ISAAC.2022.37.
- 11 Szilárd Zsolt Fazekas, Naoya Iwano, Yu Kihara, Ryuichi Matsuoka, Shinnosuke Seki, and Hinano Takeuchi. Towards composable computations by RNA co-transcriptional folding: A proof-of-concept demonstration of nested loops in oritatami. *Theoretical Computer Science*, 999:114550, 2024. doi:10.1016/J.TCS.2024.114550.
- 12 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W H Freeman & Co, 1979.
- 13 Cody Geary and Ebbe S. Andersen. Design principles for single-stranded RNA origami structures. In *Proceedings of the 20th International Conference on DNA Computing and Molecular Programming (DNA20)*, volume 8727 of *LNCS*, pages 1–19, 2014. doi:10.1007/978-3-319-11295-4_1.
- 14 Cody Geary, Guido Grossi, Ewan K. S. McRae, Paul W. K. Rothmund, and Ebbe S. Andersen. RNA origami design tools enable cotranscriptional folding of kilobase-sized nanoscaffolds. *Nature Chemistry*, 13:549–558, 2021.
- 15 Cody Geary, Pierre-Étienne Meunier, Nicolas Schabanel, and Shinnosuke Seki. Programming biomolecules that fold greedily during transcription. In *Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016)*, volume 58 of *LIPIcs*, pages 43:1–43:14, 2016. doi:10.4230/LIPIcs.MFCS.2016.43.

- 16 Cody Geary, Paul W. K. Rothmund, and Ebbe S. Andersen. A single-stranded architecture for cotranscriptional folding of RNA structures. *Science*, 345(6198):799–804, 2014.
- 17 Raymond Greenlaw, H. James Hoover, and Walter L. Ruzzo. *Limits to parallel computation: P-completeness theory*. Oxford University Press, 1995.
- 18 Yo-Sub Han, Hwee Kim, Makoto Ota, and Shinnosuke Seki. Nondeterministic seedless oritatami systems and hardness of testing their equivalence. *Natural Computing*, 17(1):67–79, 2018. doi:10.1007/S11047-017-9661-Y.
- 19 Richard E. Ladner. The circuit value problem is log space complete for p. *SIGACT News*, 7(1):18–20, 1975. doi:10.1145/990518.990519.
- 20 Mo Li, Mengxi Zheng, Siyu Wu, Cheng Tian, Di Liu, Yossi Weizmann, Wen Jiang, Guansong Wang, and Chengde Mao. In vivo production of RNA nanostructures via programmed folding of single-stranded RNAs. *Nature Communications*, 9:2196, 2018.
- 21 Kohei Maruyama and Shinnosuke Seki. Counting infinitely by oritatami co-transcriptional folding. *Natural Computing*, 20(2):329–340, 2021. doi:10.1007/S11047-021-09842-6.
- 22 Abdulmelik Mohammed, Pekka Orponen, and Sachith Pai. Algorithmic design of cotranscriptional folding of 2D RNA origami structures. In *Proceedings of the 17th International Conference on Unconventional Computation and Natural Computation (UCNC2018)*, volume 10867 of *LNCs*, pages 159–172. Springer, 2018. doi:10.1007/978-3-319-92435-9_12.
- 23 Ruth Nussinov, George Pieczenik, Jerrold R. Griggs, and Daniel J. Kleitman. Algorithms for loop matchings. *SIAM Journal on Applied Mathematics*, 35(1):68–82, 1978.
- 24 Pekka Orponen, Shinnosuke Seki, and Antti Elonen. Secondary structure design for cotranscriptional 3D RNA origami wireframes. In *Proceedings of the 31st International Conference on DNA Computing and Molecular Programming (DNA31)*, volume 347 of *LIPIcs*, pages 6:1–6:18. Schloss Dagstuhl, 2025. doi:10.4230/LIPIcs.DNA.31.6.
- 25 Daria Pchelina, Nicolas Schabanel, Shinnosuke Seki, and Guillaume Theyssier. Oritatami systems assemble shapes no less complex than tile assembly model (atam). In *Proceedings of the 39th International Symposium on Theoretical Aspects of Computer Science (STACS 2022)*, volume 219 of *LIPIcs*, pages 51:1–51:23. Schloss Dagstuhl, 2022. doi:10.4230/LIPIcs.STACS.2022.51.
- 26 Erik Poppleton, Niklas Urbanek, Taniya Chakraborty, Alesandra Griffo, Luca Monari, and Kerstin Göpfrich. RNA origami: Design, simulation and application. *RNA Biology*, 20(1):510–524, 2023.
- 27 Ahmed Shalaby and Damien Woods. An efficient algorithm to compute the minimum free energy of interacting nucleic acid strands. In *Proceedings of the 52nd International Colloquium on Automata, Languages, and Programming (ICALP2025)*, volume 334 of *LIPIcs*, pages 130:1–130:20. Schloss Dagstuhl, 2025. doi:10.4230/LIPIcs.ICALP.2025.130.

PENSim: A Toolkit for PEN-DNA Systems

Gwendal Ducloz 

École Normale Supérieure de Lyon, LIP (UMR 5668, Équipe MC2), France

Nicolas Schabanel 

CNRS, École Normale Supérieure de Lyon, LIP (UMR 5668, Équipe MC2) and IXXI, France

Abstract

The PEN-DNA toolbox has emerged as a versatile framework for implementing chemical reaction networks with DNA and enzymes, enabling applications ranging from molecular sensing to neural computation. A simulation tool, DACCAD, has provided a standardized approach for translating network schematics into dynamical models, facilitating *in silico* design and validation. However, the evolution of the PEN-DNA toolbox – particularly the transition from inhibitor-based regulation to drain-template-based inactivation – introduces mechanisms that are only partially captured by existing simulation frameworks, limiting the ability to reliably simulate and design recent PEN-DNA systems.

In this context, we introduce PENSim, a simulation package that aims to realign computational modeling with current experimental practices. By extending the repertoire of simulated reactions and incorporating thermodynamic dependencies into kinetic rate calculations via NUPACK, PENSim enables a more faithful representation of modern PEN-DNA systems. In particular, accounting for thermodynamic effects is essential for accurately modeling inactivation via drain templates.

PENSim is validated through qualitative reproduction of existing experimental results, including inactivation-driven bistability, microRNA detection circuits, and linear classifiers based on neural networks. By bridging legacy modeling approaches with recent experimental advances, PENSim provides a step toward more realistic and flexible *in silico* design of systems based on the PEN-DNA toolbox.

2012 ACM Subject Classification Computer systems organization → Molecular computing; Computing methodologies → Molecular simulation; Applied computing → Chemistry

Keywords and phrases PEN-DNA toolbox, DNA circuit simulation, DNA reaction kinetics, Enzymatic DNA reaction kinetics

Digital Object Identifier 10.4230/LIPIcs.DNA.32.4

Supplementary Material *Software (Source Code)*: <https://github.com/GwendalDucloz/PENSim>
archived at `swb:1:dir:ca74bb35d34748ecdb887f1601f05221aabd8233`

Funding This work was supported by the French government under the “France 2030” programme ANR-24-RR11-0001 and ANR-22-CE45-0020-01.

Acknowledgements We thank Guillaume Ginès, Antoine Anastassiades, Quentin Mosagna, Alexis Autour and Nathanaël Aubert-Kato for stimulating discussions.

1 Introduction

Nucleic acid strands – both DNA and RNA – are harnessed to perform a wide range of tasks, spanning chemical [11, 32, 40], mechanical [7, 34], biomedical [12, 18, 27], and computational [5, 21, 36] applications. In particular, their ability to form base pairs enables the design of 2D or 3D nanostructures, which in turn confer precise functional behaviors.

In the context of computing, several paradigms have emerged. Some rely exclusively on DNA, as for instance, tile and brick assembly models [9, 36], which perform computation by dynamically expanding a DNA-based network. Alternatively, another DNA-only approach relies on chemical reaction networks, where computation is achieved through toehold-mediated strand displacement [5, 13].



© Gwendal Ducloz and Nicolas Schabanel;

licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 4; pp. 4:1–4:24

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In recent years, a novel paradigm has emerged: the *PEN-DNA toolbox* [4, 10, 20], which integrates DNA with enzymatic processes to implement chemical reaction networks. Specifically, this framework exploits three types of enzymes, forming the so-called *Polymerase-Exonuclease-Nickase Dynamic Network Assembly (PEN-DNA)* toolbox (see Section 2 for details). This framework has been successfully applied to engineer biochemical oscillators [20], model predator-prey dynamics [10], and, more recently, construct small neural networks [21]. The ability to implement neural networks is a proof of concept that the PEN-DNA toolbox is functionally complete and may be capable of Turing-complete computations under idealized conditions (e.g. unbounded precision and infinite memory). Beyond computational applications, the PEN-DNA toolbox also shows applications in biomedical contexts, such as the detection of microRNAs, enzymes, and other small molecules [11, 12, 18, 32].

When working with any DNA model, *in silico* simulation is a critical first step to assess computational systems before advancing to *in vitro* validation. Therefore, each paradigm is supported by a dedicated simulation framework: kTAM for tile assembly models [35], systems based on ordinary differential equations or continuous-time Markov chains for toehold-mediated strand displacement systems [15, 39], and DACCAD for the PEN-DNA toolbox [2, 3]. These frameworks not only enable the prediction and optimization of nucleic acid-based systems but also bridge the gap between theoretical design and experimental implementation. Developing and refining such tools is therefore essential, as it allows simulations to increasingly reflect biochemical reality and better guide experimental outcomes.

Contribution

This work presents PENSIm, a new toolkit for the PEN-DNA toolbox that addresses limitations of the existing DACCAD framework. PENSIm enhances the simulation capabilities for systems utilizing the latest PEN-DNA toolbox features with two main contributions.

First, it expands the scope of simulable reactions by implementing mechanisms introduced in recent versions of the PEN-DNA toolbox, including reporter templates, EvaGreen fluorescence, inactivation via drain templates, irreversible conversion, and support for multiple versions of the same signal strand [19, 21, 29]. These mechanisms are central to modern PEN-DNA systems – such as neural networks and microRNA sensing – but are not supported in available versions of DACCAD, thereby limiting the simulation and design of such systems. Furthermore, PENSIm includes experimental code modules supporting additional reactions, such as those involving non-phosphorylated and non-protected templates, reflecting a design choice toward extensibility and future integration of emerging mechanisms. Importantly, PENSIm is designed to complement rather than replace DACCAD, as it does not implement inhibitor-based regulation from earlier versions of the toolbox, which is no longer used in recent studies.

Second, PENSIm improves the fidelity of simulations by introducing a *thermodynamically grounded approach to kinetic rate computation*. Rather than relying on uniform or user-defined kinetic parameters, the simulator derives reaction rates directly from biochemical properties using NUPACK and a kinetics nearest neighbor model [26, 38]; including sequence, temperature, enzyme, and buffer conditions. This enables a more accurate representation of DNA hybridization and enzymatic processes, which is critical for capturing behaviors driven by inactivation mechanisms and for modeling complex systems such as enzymatic neural networks [19, 21].

By enabling simulation from DNA sequences and experimental parameters, PENSIm provides a practical tool for experimentalists designing new PEN-DNA circuits. The simulator, along with validation notebooks and usage examples, is openly accessible in the accompanying

Git repository (<https://github.com/GwendalDucloz/PENSim>). It is validated through the qualitative reproduction of experimentally observed behaviors of previously implemented systems spanning dynamical circuits, molecular sensing, and neural computation [19, 21, 29].

Section 2 introduces the mechanisms of the PEN-DNA toolbox and their representation as a graph-like schematic. DACCAD, and related work on simulating PEN-DNA toolbox systems are reviewed in Section 3. Our contribution begins in Section 4, where we detail the dynamics of key subreactions – DNA hybridization and enzymatic reactions – and present our method for computing their kinetics. Section 5 outlines the code architecture, provides usage tutorials, and describes the species considered (DNA single strands and complexes), along with the construction of their governing equations. Validation of the simulator is presented in Section 6. An appendix includes additional features of the PEN-DNA toolbox, supplementary figures, and an overview of the set of equations solved by PENSim.

2 The PEN-DNA toolbox

A system built using the PEN-DNA toolbox, called *PEN-DNA system*, is composed of DNA strands and three types of enzymes: Polymerase, Exonuclease, and Nickase(s). All these components interact simultaneously through various reactions that produce, transform, and degrade DNA strands. Generally, PEN-DNA systems are designed so that the concentrations of all DNA species either reach a periodic behavior [10, 20] or equilibrium [19, 21].

2.1 Reactions of the PEN-DNA toolbox

Five types of mechanisms are designed within the PEN-DNA toolbox, described in Figure 1. They are the *conversion*, the *reporting*, the *inactivation*, the *degradation*, and the *killing conversion*. Each reaction, except degradation, needs a specific DNA strand, called *template*.

Conversion

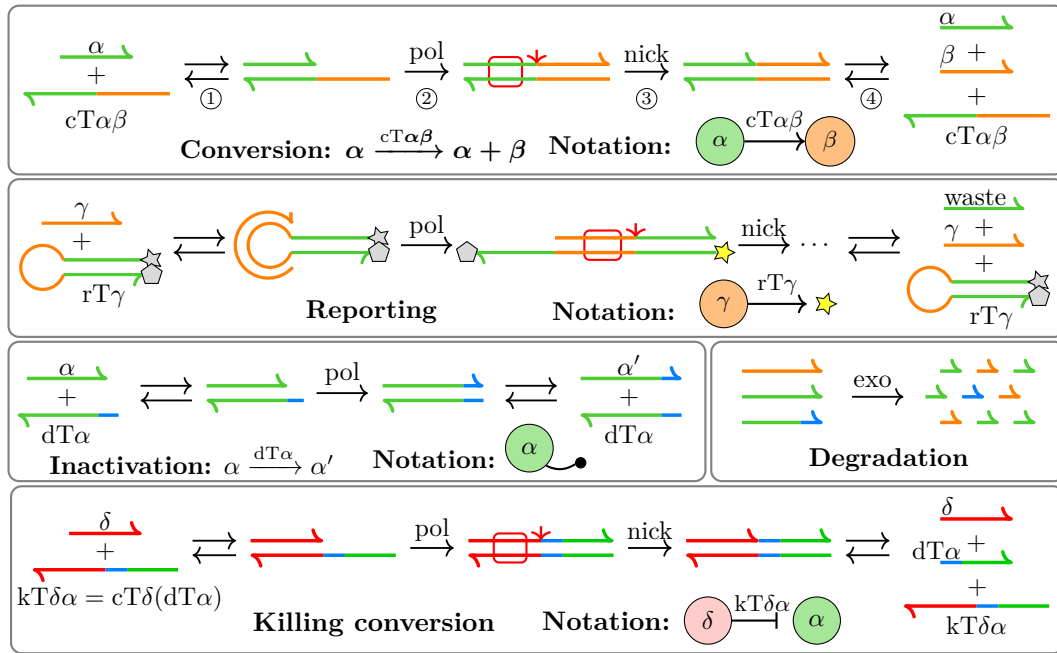
A **converter template (cT)** enables the production of a new DNA strand β , called the *output strand* or *output signal*, from an initial DNA strand α , called *input strand* or *input signal*. The conversion $\alpha \xrightarrow{cT\alpha\beta} \alpha + \beta$, happens as illustrated in Figure 1:

- ① The strand α binds to the converter template $cT\alpha\beta$.
- ② The polymerase binds to the resulting complex and extends α according to the complementary sequence of the template $cT\alpha\beta$.
- ③ The nickase binds to the complex on its recognition sequence and cleaves the extended strand at a specific position, separating α from the newly synthesized part β .
- ④ At the working temperature, both strands α and β can unbind from the template $cT\alpha\beta$.

After step 3, if α and β are still attached to the template, the polymerase can bind again to the 3' end of α and extend it, dislocating the strand β . This intermediate process is equivalent to step 2, but provides an output signal β by polymerase displacement.

When the output signal of a converter template is the same as its input signal, we refer to it as an *autocatalytic template*.

Some converter templates are special, extending first their input signal into a longer one. Since this new input has a higher melting temperature, we consider that it never unbinds from the template at the working temperature, as illustrated in Figure 6 (Appendix). We refer to this kind of template as *irreversible template*.



■ **Figure 1** Schematic of the main reactions underlying PEN-DNA systems: conversion, reporting, inactivation, degradation, and killing conversion. Abbreviations pol, nick and exo denote the enzymatic actions of polymerase, nickase and exonuclease. A red arrow $\rightarrow$ indicates the cleavage position of a nickase enzyme, and a red rectangle its recognition sequence. A star $\star$ represents an emitting fluorophore, while a pentagon $\blacklozenge$ represents its associated quencher.

Some converter templates may also feature a truncated 5' end to address thermodynamic or kinetic constraints, as illustrated in Figure 7 (Appendix). This truncation introduces a bias: when a signal has two binding options – one with a full-length template and one with a shortened template – the interaction with the full-length version will be thermodynamically favored [19].

Inactivation and killing conversion

A **drain template (dT)** inactivates a strand α by converting it into an extended form, denoted α' , via the reaction $\alpha \xrightarrow{dT\alpha} \alpha'$. While α' can still bind to the converter templates, it cannot be extended by the polymerase anymore because its 3' end is misaligned with the template, thereby preventing further conversion, as illustrated in Figure 8 (Appendix).

A converter template whose output strand is a drain template is called **killer template (kT)**. A schematic is presented in Figure 1.

Reporting

A **reporter template (rT)** functions as a molecular beacon, designed to signal the presence of a specific strand [14]. As illustrated in Figure 1, it adopts a hairpin secondary structure at the working temperature. In this conformation, the quencher is positioned adjacent to the fluorophore, effectively preventing fluorescence. However, when a reported strand binds to the hairpin loop, it is extended by the polymerase. This extension opens the hairpin, spatially separating the quencher from the fluorophore and enabling fluorescence.

The reporter template can also switch off if the reported strand is no longer present in solution. To achieve this, the extended reported strand is cleaved by the nickase into two short fragments, which are sufficiently small to unbind at the working temperature. Once detached, these fragments are degraded by the exonuclease, allowing the reporter template to refold into its hairpin loop. This refolding repositions the quencher near the fluorophore, quenching fluorescence once again.

The reporting process is not the only mechanism for tracking the evolution of a PEN-DNA system. *EvaGreen*, a green fluorescent nucleic acid dye, is also commonly used to quantify the amount of double-stranded DNA species present in the system.

Degradation

All DNA strands in PEN-DNA systems are subject to degradation by exonuclease, which cleaves nucleotides sequentially from the strand ends. The *ttRecJ* enzyme is commonly employed, degrading strands from their 5' ends [23]. To prevent degradation, DNA strands can be chemically protected using three phosphorothioate bases at their 5' ends. This modification replaces one of the oxygen atoms in the DNA backbone with a sulfur atom, making the strand less recognizable and resistant to enzymatic cleavage [33].

2.2 Graph representation

To represent a PEN-DNA system, a graph-like schematic is used. The notation – illustrated in Figure 1 – is inspired by gene regulatory networks. A simple arrow ($\rightarrow$) denotes a conversion, a black dot ($\hookrightarrow$) represents a drain template, a T-shaped arrow ($\dashrightarrow$) indicates a killing conversion, and an arrow pointing to a star ($\star$) refers to a reporting.

3 Related work

PEN-DNA systems are simulated by modeling the interactions between all species over time using ordinary differential equations (ODEs). Specifically, the general reactions described in Section 2 can be decomposed into two fundamental sub-reactions: *DNA hybridization* and *enzymatic processes*.

While many articles on PEN-DNA systems include supplementary sections on differential equations for simulation purposes, these sections are often dense and difficult to interpret due to different and complex notations [19, 20, 21].

To standardize the simulation of any PEN-DNA system, Aubert-Kato et al. developed *DACCAD* (DNA Artificial Circuit Computer-Assisted Design) [2]. This tool automates the generation of differential equations from the graph representation of a PEN-DNA system and solves them, significantly accelerating the design process. *DACCAD* has proven to be both consistent and reliable, qualitatively reproducing experimental results such as oscillators and switchable memory systems [2, 20, 23].

However, since the development of *DACCAD*, the PEN-DNA toolbox has evolved into a new version, replacing inhibitor-based regulation with drain template inactivation. Drain templates are preferred because they irreversibly convert active strands into non-functional species and provide a tunable degradation pathway that improves control over network dynamics [19].

It has been reported that *DACCAD* is extensible to support advanced features like reporter or drain templates. To our knowledge, although some of the corresponding equations have been described in detail [3], their practical implementation remains unavailable. Additionally,

a previous attempt to simulate drain templates in DACCAD revealed an unexpected behavior: a bistable system switched off only after spiking a large concentration of drain species over several minutes, rather than through a single spike of 60 nM of drain [3].

Furthermore, the latest version of the toolbox introduces additional complexities, including the use of two distinct nicking enzymes (Nb.BsmI and Nt.BstNBI), irreversible templates, and different versions of the same signal strand [12, 21]. These innovations represent new reactions that would be valuable to simulate, further highlighting the need for updated simulation capabilities.

DACCAD faces another limitation in its computation of hybridization rates. Initially, it assigns a uniform binding rate to all signals and derives unbinding rates from the dissociation constant, as detailed in Section 4. The tool requires users to input the dissociation constant for each signal, assuming that all rates associated with a given signal are identical. This simplification was primarily due because all converter templates and signals were used in their full-length version, which was sufficient to simulate systems based on the first version of the PEN-DNA toolbox. However, unbinding rates depend on the binding sequence length, leading to differences between full-length converter templates and their 3' shortened counterparts. This distinction has been identified as a key factor in implementing inactivation processes: the dissociation rate between a signal and its drain template differs from that between the same signal and its converter template [19]. Moreover, even the same signal can be present under shortened or elongated forms, to speed up certain reactions, as it has been used in some studies [12, 29]. To ensure consistency with the latest experimental findings, simulations should now account for these discrepancies by defining a specific kinetic rate for each pair of interacting DNA strands in the system.

A final limitation pertains to the kinetics rates of enzymatic processes. In DACCAD, users can adjust the Michaelis-Menten constants of all enzymatic processes. Nevertheless, these constants are not derived from an enzyme concentration, which could be more intuitive and user-friendly. Additionally, these rates should be derived from multiple parameters, including salt concentrations, temperature, and template sequences.

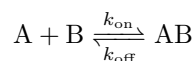
In the following sections, we present PENSim, a toolkit for generating and solving the ordinary differential equations (ODEs) of PEN-DNA systems. This tool addresses most of the limitations of previous approaches and enables qualitative reproduction of the behavior of complex and recent systems, which are based on inactivation mechanisms.

4 Kinetics and rates computation

In this section, we describe how the kinetic rates used in PENSim are computed. We first present DNA hybridization dynamics, then enzymatic reactions, and finally discuss their limitations.

4.1 DNA hybridization

The overall chemical equation for DNA hybridization between strands A and B is



where AB is the DNA duplex formed by A and B, and k_{on} and k_{off} are the *association* and *dissociation* rates.

The association reaction is first-order in each reactant, described by $\frac{d[AB]}{dt} = k_{\text{on}}[A][B]$ with k_{on} typically ranging between 7×10^5 and $3 \times 10^7 \text{ M}^{-1} \cdot \text{s}^{-1}$ [1, 39].

The dissociation reaction is also first-order, described by $\frac{d[AB]}{dt} = -k_{\text{off}}[AB]$ with k_{off} typically ranging between 6×10^{-6} and $4 \times 10^3 \text{ s}^{-1}$ [1, 39].

At equilibrium, the concentration of AB is constant and $k_{\text{on}}[A][B] = k_{\text{off}}[AB]$. The *dissociation constant* K_D , which quantifies the equilibrium binding affinity, is given by:

$$K_D := \frac{[A][B]}{[AB]} = \frac{k_{\text{off}}}{k_{\text{on}}} \quad (1)$$

Under the quasi-equilibrium hypothesis, k_{on} can be derived from k_{off} (or vice versa), if the dissociation rate is known. In DACCAD, a global k_{on} value of $3.3 \times 10^6 \text{ M}^{-1} \cdot \text{s}^{-1}$ and a signal-specific K_D of $74.328 \times 10^{-9} \text{ M}$ are used by default, unless otherwise specified by the user.

4.1.1 An automatized computation

Decades of research into the mechanism of hybridization have led to a deep understanding of its thermodynamics, resulting in the development of detailed computational models [16, 30, 38]. However, many questions remain regarding its kinetics and dynamics.

Recent advances include coarse-grained computational models that describe hybridization as a two-step process: nucleation of a critical base-pair stretch followed by rapid zippering [8, 22]. Tools like Multistrand, which simulates nucleic acid kinetics using continuous-time Markov chains, enable the exploration of folding pathways and reaction rates [17, 31].

Another approach involves developing nearest-neighbor (NN) thermodynamic models to compute association and dissociation rates directly from the sequence, similar to how free energy is calculated [26, 30]. This approach is based on the *Eyring equation*, which computes the rate of a chemical reaction:

$$k = \frac{k_B T}{h} \exp\left(-\frac{\Delta G^\ddagger}{RT}\right) \quad (2)$$

where:

- T is the temperature,
- k_B is the Boltzmann constant,
- h is the Planck constant,
- R is the gas constant,
- $\Delta G^\ddagger$ is the Gibbs energy of activation.

The Gibbs energy of activation can be computed using the thermodynamic equation:

$$\Delta G^\ddagger = \Delta H^\ddagger - T\Delta S^\ddagger \quad (3)$$

where $\Delta H^\ddagger$ is the enthalpy of activation and $\Delta S^\ddagger$ is the entropy of activation.

The nearest-neighbor model designed by Rejali et al. [26] allows the computation of the enthalpy and entropy of activation from the DNA sequence. The article provides two NN models for association and two for dissociation, each corresponding to one of two buffer conditions: 1.0 M NaCl or 2.2mM of MgCl₂.

For dissociation, the models demonstrated high accuracy: in 1.0 M NaCl, predicted dissociation rates were within 3-fold of literature data. However, for association, predictions were less accurate, with NN predictions differing from experiments by $35 \pm 30\%$ even on the training data. This discrepancy is attributed to weak temperature dependence and overparameterization of the NN terms.

Given that the thermodynamics of hybridization are well-studied and models are well-established, we aim to ensure that our computed rates are consistent with the dissociation constant K_D . To achieve this, we use only one of the nearest-neighbor (NN) models – either for association or dissociation – and derive the complementary rate from the K_D value computed using NUPACK [38].

Although NN models for association are less accurate, and exhibit weaker dependence on thermodynamic parameters – such as sequence length and temperature – we chose to derive k_{off} from k_{on} in PENSIm. When the reverse approach is taken, even small errors in the computation of k_{off} are amplified upon division by K_D , rendering k_{on} overly sensitive to variations in sequence length and temperature. A detailed comparison of these approaches is available in the file `resources/comparison_computation_kon_koff.ipynb` in the Git repository and in Figure 9 (Appendix).

4.1.2 Coaxial stacking slowdown

At a nick, adjacent bases undergo collaborative stabilization through *coaxial stacking*, which slows the release of both input and output strands. This effect depends on the bases located at the nick site. Early studies in DACCAD estimated realistic slowdown values based on the data available at the time [2, 24, 37]. However, due to experimental constraints, these studies could only measure stacking interactions between base *pairs*, not between individual bases. This has prevented knowing stacking energies between A and C for example, because in the context of a duplex this would be necessarily paired with stacking of the hybridized bases (T and G in this example).

Recently, a study using a centrifuge force microscope advanced our understanding of coaxial stacking energy by directly measuring the unbinding slowdown [25]. Building on these findings, we incorporated the updated values into our model, automating the computation to account for temperature and nick sequence using the formula:

$$\text{stack_slowdown} = \exp\left(\frac{\Delta G_{\text{stack}}}{RT}\right)$$

This refinement shifts the reference slowdown value in DACCAD from 0.2 to 0.035, as initial works had underestimated the coaxial stacking energy.

4.1.3 Limitations

Despite the improved accuracy of our hybridization kinetic rate computations, several uncertainties need to be pointed out. First, the NN model was calibrated specifically for 1.0 M NaCl, which does not match the buffer conditions used in PEN-DNA systems – which usually contain 0.07 M monovalent ions and 0.0125 M bivalent ions. Note that although these buffer conditions do not match, this is not a major concern for qualitative predictions, since all association rates k_{on} are relatively close from each other. System behavior is primarily determined by the dissociation rates, which are computed through NUPACK using the actual buffer concentrations. For example, a DNA strand will bind at nearly the same speed to a shortened converter template and to its full-length counterpart, but it will unbind much more quickly from the shortened version.

Second, regarding again association rates, the NN model neglects the impact of secondary structures on kinetics, prioritizing computational speed over accuracy. More sophisticated approaches, such as Multistrand or molecular dynamics simulations, could refine these computations [1, 17].

Finally, chemical modifications to the DNA backbone – such as 5' phosphorothioate – are known to affect both kinetic and thermodynamic constants [1, 6]. Since these modifications are employed in the PEN-DNA toolbox to prevent degradation, the release of output signals may be accelerated. Additionally, the presence of netropsin, used to stabilize PEN-DNA systems, may further alter all kinetic rates.

4.2 Enzymatic reactions

All enzymatic processes in this study are described using Michaelis–Menten kinetics, under the quasi-equilibrium hypothesis [28]. In the case of several reactions using the same enzyme ($S_i \xrightarrow{E} P_i$), their competitive behavior is modeled by the rate equation $\frac{d[P_i]}{dt} = \text{enz}_i \cdot [S_i]$, where the *effective enzymatic activity*, enz_i , is given by:

$$\text{enz}_i = \frac{k_{\text{cat},i} \cdot [E]_{\text{total}}}{K_i \cdot \left(1 + \sum_j \frac{[S_j]}{K_j}\right)}$$

where:

- $[P_i]$ and $[S_i]$ are the product and substrate concentrations (in nM),
- $k_{\text{cat},i}$ is the *catalytic rate constant* associated to the i -th enzymatic process,
- K_i is the *Michaelis constant* (in nM) associated to the i -th enzymatic process,
- $[E]_{\text{total}}$ is the total enzyme concentration (in nM, or in units/mL).

For simplicity, PENSIm assumes that all catalytic rate constants ($k_{\text{cat},i}$) and Michaelis constants (K_i) are identical for a given enzyme, except for the polymerase. For this enzyme, we distinguish two cases: whether it is displacing the output strand or not. Rates associated with this special case are denoted with the suffix `_displ` in subsequent sections.

4.2.1 Rates computation

In PENSIm, we adopted the same enzymatic kinetic constants as those used in DACCAD processes involving polymerase, exonuclease, and the Nt.BstNBI nickase enzyme, based on the study by Padirac et al. [2, 23]. However, we modified the approach to compute $V_{\text{max}} = k_{\text{cat}} \cdot [E]_{\text{total}}$ directly from the enzyme concentration, enhancing user-friendliness.

Additionally, we incorporated the distinct kinetic constants of the Nb.BsmI nicking enzyme, which differ from those of Nt.BstNBI, as measured by Montagne et al. [19]. All kinetic rates are summarized in Table 1 (Appendix).

4.2.2 Limitations

All enzymatic rates depend on temperature, salt conditions and template sequence [11, 18]. While we currently use constant rates for simplicity, this approach will be refined in future work, as precise method for measuring enzymatic rates are available [11]. Additionally, the polymerase rates described here apply to the Bst DNA polymerase. To our knowledge, no measurements exist for the Vent(exo-) polymerase, which is also used in PEN-DNA systems [18, 21].

Intuitively, these remarks primarily limit quantitative predictions. However, one should also note that, since enzymatic parameters are fixed as constants, this likely impacts qualitative explorations of temperature and buffer conditions as well.

5 Overview of PENSIm

5.1 Installation and code organization

PENSIm is a piece of software developed in Python. It relies on basic packages but also on NUPACK for computing the kinetic constants [38]. PENSIm is publicly available on this Git repository: <https://github.com/GwendalDucloz/PENSIm>.

PENSIm can be installed by following the instruction in the README. The main code is organized into three files:

- `strands.py` defines the DNA strands classes, as described in the following subsection.
- `rates.py` manages all the kinetics rates, as detailed in Section 4.
- `PEN_System.py` is the core file for system setup, ODEs generation, simulation execution, and result visualization (including graph-like representations).

5.2 Species and equations

As previously described, the system is simulated using an ordinary differential equation (ODE) of the form: $\frac{dx}{dt} = f(x)$, where each coordinate x_i of the variable vector x represents the concentration of a species within the PEN-DNA system. This species may be a single strand or a complex of strands.

Four classes of species are defined in the `strands.py` file:

- **Signal:** Can exist in two states:
 - `alone`: Free in solution.
 - `drained`: The Signal once inactivated by a drain template, free in solution.
- **Drain:** Can exist in three states:
 - `alone`: Free in solution.
 - `in`: Bound to the target signal.
 - `ext`: Bound to the inactivated signal.
- **Template:** Can exist in six states:
 - `alone`: Free in solution.
 - `in`: Bound to its input signal.
 - `out`: Bound to its output signal.
 - `ext`: Bound to its extended input signal (via polymerase activity).
 - `both`: Bound to both input and output signals.
 - `in_drained`: Bound to an inactivated input strand.

Note 1: The state where the template is bound to an inactivated output strand was not implemented, as it is suspected not to significantly inhibit the system.

Note 2: Irreversible Template includes an additional state called `load`. It corresponds to the initial binding step of the input signal to the template before irreversible extension. The `in` state refers to the irreversible binding of the extended input (illustrated in Figure 6).

- **Reporter Template:** Shares the same states as reversible templates, except for `in_drained`, which cannot open the hairpin structure of reporter templates.

The ODEs for all species are computed by decomposing each reaction into either (de)hybridization or enzymatic processes (see Section 4). An overview of the resulting system of equations is provided in Appendix D (Appendix), while the full set of equations is available in the `resources/` folder within the accompanying Git repository.

Importantly, in addition to hybridization and enzymatic processes, we must account for a polymerase default behavior: even in the absence of an input signal, conversion can be spontaneously initiated by spurious polymerization or impurities [19]. This phenomenon, referred to as *leak*, is modeled in PENSIm by defining a leak rate, which quantifies the fraction of polymerase molecules initiating spurious reactions.

5.3 Tutorials

Tutorials of use are available in the `tutorials/` directory of the accompanying Git repository. The notebook `tutorial1_basic.ipynb` guides users through basic simulations of a PEN-DNA system, covering:

1. **System definition:** Specifying experimental conditions (temperature, salt concentrations, enzyme concentrations, and polymerase leak rate) and the main species (signals, drain templates, converter templates, and reporter templates).
2. **Simulation execution** over a predefined time range.
3. **Result visualization:** Plotting species concentrations and fluorescence (reporter templates/EvaGreen dye).
4. **System validation:** Checking kinetic rates and graph-based coherence.

This tutorial illustrates a simple system with the mechanisms presented in Figure 1 (conversions, inactivation, killing conversions, reporting).

Two additional tutorials explore other simulable scenarios, including systems with multiple versions of the same signal strands, templates degraded by exonucleases, and 3' shortened templates extended by polymerase.

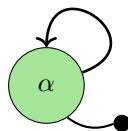
6 Validation

In this section, we validate PENSIm across three representative classes of PEN-DNA systems: dynamical systems (bistability), sensing circuits (microRNA detection), and computational architectures (neural networks). Unless otherwise specified, all simulation parameters are directly sourced from the referenced articles. The only tuned parameter is the polymerase leak rate, which is generally set to 6×10^{-7} , unless specified otherwise. All simulations are available in the `validation/` folder within the Git repository.

6.1 Bistable systems and inactivation process

A key contribution of this article is the ability to simulate the inactivation process (described in Section 2). Our first validation demonstrates that the simulated process exhibits the same behavior as experimentally observed in the study from Montagne et al. [19]. The simulation is available in the notebook `validation/drain_validation.ipynb` in the accompanying Git repository.

Specifically we simulate a system comprising a single signal α , its autocatalytic template, and a drain template, as schematized below:



Our simulations qualitatively confirm the following main behaviors, as detailed in the original article:

1. Self-reproduction and equilibrium of α

In the absence of drain templates, α self-reproduces through conversion until it reaches equilibrium, where its production is balanced by degradation via exonuclease. However, this amplification spontaneously starts even without α , due to polymerase leak.

Drain templates are introduced to capture and inactivate self-produced α , thereby slowing the self-activation. Beyond the *critical drain concentration*, a bifurcation occurs, stabilizing the $[\alpha] = 0$ nM state. Past this point, the system becomes *bistable*, exhibiting either no α or a stable positive concentration.

The critical drain concentration is determined by defining the *start time* (C_t) as the time at which the concentration of α reaches 20% of its maximal value. This concentration corresponds to the point where $1/C_t$ reaches 0, with $1/C_t$ set to 0 in the absence of amplification. This process is illustrated in Figure 2.

2. Impact of drain template length

The longer the drain template, the slower the inactivated strand (α') dissociates, reducing the efficiency of the inactivation process. Consequently, longer drain templates require a higher critical drain concentration to achieve bistability, as revealed in Figure 10 (Appendix).

3. Effect of shortened 3' end in the autocatalytic template

Thermodynamic predictions indicate that a shortened 3' end in the autocatalytic template enhances the binding affinity of α for the drain template (which exhibits a full length interaction). As a result, inactivation becomes more efficient, and the critical drain concentration decreases as the 3' end of the autocatalytic template is shortened. This behavior is presented in Figure 11 (Appendix).

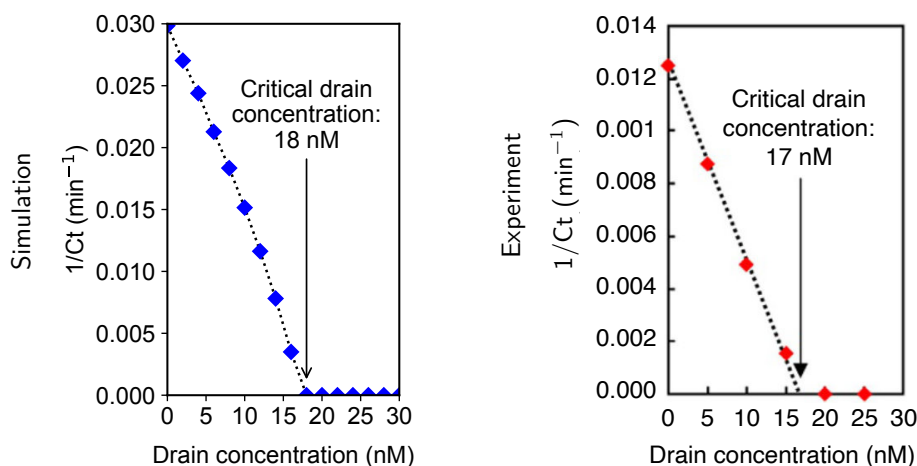
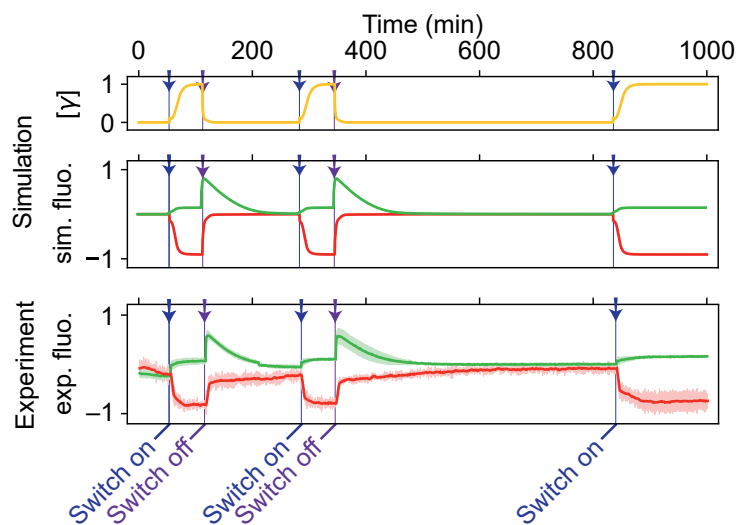


Figure 2 $1/C_t$ versus drain concentration for simulated (left, blue) and experimental data (right, red) adapted from Montagne et al. [19]. In the simple autocatalytic network, self-initiation occurs spontaneously at a drain concentration of 0. As the drain template concentration increases, it first delays and eventually suppresses this self-initiation. In this simulation, the leak rate has been set to 2×10^{-7} .

Furthermore, in an analog bistable system using γ instead of α , we simulated the system's switch-on and switch-off behavior by triggering either γ (on) or $dT\gamma$ (off). The simulations successfully reproduced the expected γ concentration dynamics and the experimentally measured fluorescence signals (EvaGreen and red Cy3.5), as shown in Figure 3.

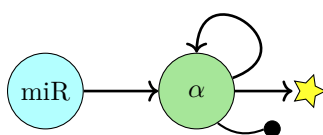
This result contrasts with previous attempts in DACCAD, where simulations of drain templates were reported to result in unexpected behavior: the system switched off only after spiking a large concentration of drain species over several minutes [3]. This discrepancy likely arose because (de)hybridization kinetics were modeled identically for both signal-drain template and signal-converter template interactions.



■ **Figure 3 Simulation and experimental comparison of bistable system dynamics.** The first two curves (top) are from our simulation, triggered by spiking either 20 nM of γ (switch-on) or 340 nM of drain γ (switch-off). The yellow curve represents the normalized concentration of γ , while the red and green curves correspond to normalized EvaGreen and red Cy3.5 fluorescences. The third curve (bottom) is adapted from Montagne et al. [19], where the system is toggled using 20 nM of γ and 60 nM of drain γ .

6.2 Sensing systems: microRNA detection

Ginès et al. recently developed PUMA, an isothermal, digital detection method for microRNAs (miR) using a PEN-DNA circuit [12, 18, 29]. This system integrates an irreversible converter template, an autocatalytic template, a drain template and a reporter template, as illustrated below:



Qualitatively, our simulations demonstrate the system's ability to detect microRNA concentrations as low as femtomoles, validating the use of different nickases in the same system (Nt.BstNBI and Nb.BsmI), reporting, and irreversible templates implementations. The simulation files are available in `validation/PUMA_simulation.ipynb` on the Git repository, and the result is presented in Figure 4.

Moreover, our model validates the presence of modified signal variants, where additional bases are truncated at the 5' end and extended at the 3' end, while maintaining compatibility with the autocatalytic template. Specifically, the converter template produces the strand called $\alpha+2-2$, which binds to the autocatalytic template with two bases downstream of the nicking position. This modification biases the thermodynamic, favoring the input region of the autocatalytic template, relative to the drain template, thereby altering the kinetics.

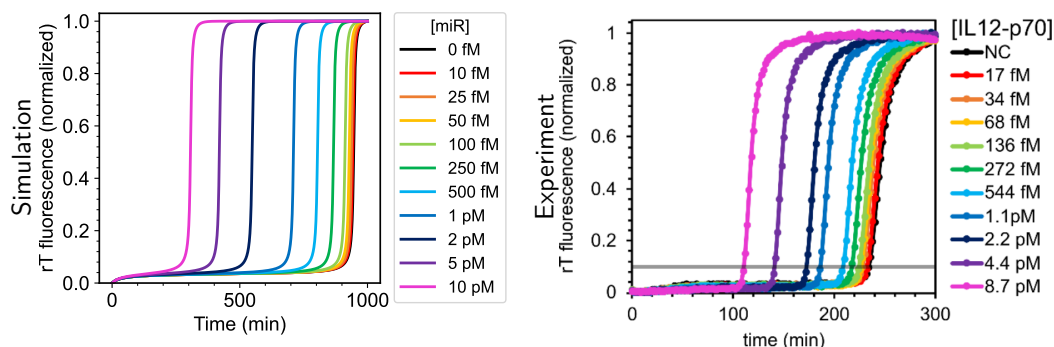


Figure 4 Simulated (left) and experimental (right) PUMA system. The experimental data are adapted from Masurier et al. [18]. The simulation was conducted at 50°C using concentrations of 0.5 nM converter template, 50 nM autocatalytic template, 9.66 nM drain template (compared to 6 nM in the experimental data), and 50 nM reporter template. Measurements were taken over a time range of 1000 minutes (extended from 300 minutes in the experiment). In both cases, the detection limit of the system is approximately 100 fM.

6.3 Neural network simulation

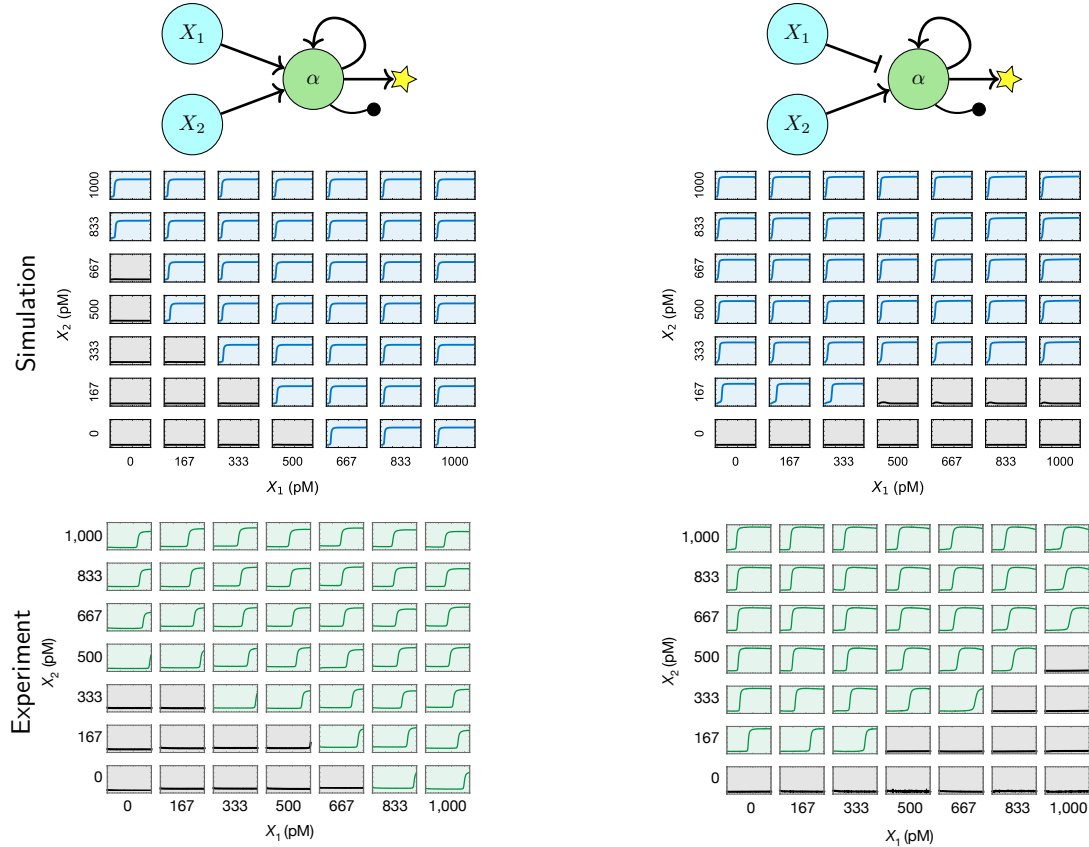
The PEN-DNA toolbox has been demonstrated to enable the implementation of neural networks [21]. From a theoretical perspective, this proof of concept is significant, as it establishes that PEN-DNA systems are functionally complete and may be capable of Turing-complete computations under idealized conditions (e.g. unbounded precision and infinite memory). Thus, the ability to accurately simulate neural networks serves as a critical validation for our simulator.

The capability of PENSIm to simulate neural networks is demonstrated in the notebook `validation/neural_network_simulation.ipynb` (available in the Git repository) and is illustrated in Figure 5. The simulator qualitatively reproduces the behavior of positive-weighted linear classifiers (exhibiting a negative slope) with respect to converter and drain template concentrations. It also approximates the behavior of negative-weighted classifiers (exhibiting a positive slope) with respect to temperature, aligning with experimental results reported by Okumura et al. [21]. From a logic gate perspective, positive-weighted classifiers function as AND gates, while negative-weighted classifiers behave as AND-NOT gates.

7 Conclusion

The PENSIm package enables the simulation of complex reactions based on the PEN-DNA toolbox, expanding the scope of previously simulable systems on DACCAD by incorporating new mechanisms used in recent studies. It also enhances quantitative precision through a novel method for computing kinetic rates, grounded in thermodynamic properties such as DNA sequences, temperature, and salt concentrations. This advancement addresses many of the limitations inherent in previous tools.

PENSIm has been validated by simulating previously experimentally characterized systems, including bistable systems [19], microRNA detection circuits [29], and linear classifiers based on neural network architectures [21]. In all cases, PENSIm produced qualitative results that faithfully replicate experimental behaviors. This makes it a valuable tool for designing and validating experiments *in silico* before laboratory implementation, while also allowing users to explore the impact of critical parameters such as DNA lengths and temperature.



■ **Figure 5 Simulation (top, blue) and experimental (bottom, green) results for linear classifiers.** Simulations were conducted using the same configuration as the experiments, from which the bottom figure is adapted [21]. The only deviation is the polymerase concentration, which was set to half the experimental value, as PENSIm was calibrated using Bst polymerase rates, whereas the experiments used Vent (exo-) polymerase – known to be slower. Another difference is that the right system was simulated at 46°C instead of 45°C. The **left** system represents a positive-weighted classification, while the **right** system represents a negative-weighted classification. The underlying PEN-DNA systems are schematized at the top. Qualitative behaviors align between simulations and experiments, with the positive-weighted classification showing strong quantitative agreement. However, the negative-weighted classification exhibits less quantitative agreement, as the slope is significantly flatter than in experiments.

However, PENSIm is not yet a quantitative oracle. While it significantly improves qualitative accuracy and modeling scope, several challenges remain to achieve fully quantitative simulations. Intuitively, the major obstacles, from most to least important, are:

1. Enzymatic reactions are insufficiently characterized. In our simulator, kinetic rates are currently modeled as dependent solely on enzyme concentrations, despite their known dependence on temperature and salt conditions. The existing rates were measured under different experimental conditions (e.g., buffer, temperature) than those used in recent systems. Additionally, enzymes such as Vent(exo-) polymerase (now often used instead of Bst) have not been measured in any PEN-DNA buffer. Enzymatic activity is also highly sequence-dependent, with rates varying by up to threefold between templates [18].
2. To our knowledge, polymerase leak rates are currently uncharacterized. In the previous simulations, this rate was set to 6×10^{-7} , as this value yields convincing results with respect to enzymatic activities. While this parameter did not affect the qualitative agreement of the simulations, it remains a major obstacle for achieving quantitative accuracy.
3. Hybridization rates are still not fully understood. Our approach does not account for secondary structures or chemical modifications, which may significantly affect reaction dynamics. Moreover, the computation of association rates in PENSIm was calibrated for a specific buffer condition, differing from the usual conditions in PEN-DNA systems.
4. Non-specific interactions, such as cross-talks and secondary structures, are not currently modeled, further limiting quantitative accuracy.

Future work

To enhance the capabilities of PENSIm, we plan to pursue the following improvements:

- Develop a web-based graphical user interface to improve accessibility for non-expert users.
- Refine enzymatic reaction kinetics to improve quantitative predictions and temperature dependencies. Additionally, conduct a study of the polymerase leak rate to properly define and model it.
- Perform experiments to calculate quantitative indicators comparing simulation results with wet lab data. In particular, focus on kinetic predictions and sensitivity limits as key targets.
- Optimize the ODE solver to accelerate simulations for large-scale systems, enabling rapid exploration of parameter spaces and experimental conditions. Additionally, conduct experiments on complex architectures to assess simulation scalability.
- Enable automated species definition from input sequences, allowing users to simulate systems directly from raw DNA sequences.
- Integrate sequence design tools to facilitate seamless transitions from theoretical models to experimental implementation.
- Incorporate additional side effects, such as exonuclease inhibition by free templates [23], and support for new reactions to align with further updates in the PEN-DNA toolbox.

References

- 1 Brennan Ashwood and Andrei Tokmakoff. Kinetics and dynamics of oligonucleotide hybridization. *Nature Reviews Chemistry*, 9(5):305–327, May 2025. doi:10.1038/s41570-025-00704-8.
- 2 Nathanaël Aubert, Clément Mosca, Teruo Fujii, Masami Hagiya, and Yannick Rondelez. Computer-assisted design for scaling up systems based on DNA reaction networks. *Journal of The Royal Society Interface*, 11(93):20131167, April 2014. doi:10.1098/rsif.2013.1167.

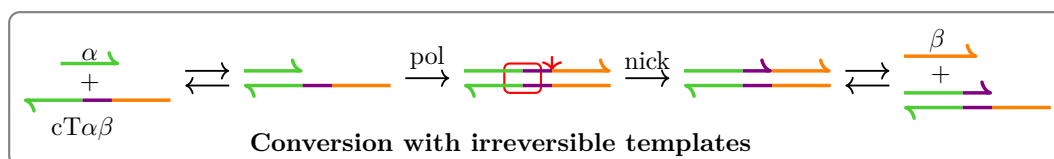
- 3 Nathanael Aubert-Kato and Leo Cazenille. Designing dynamical molecular systems with the PEN toolbox. *New Generation Computing*, 38(2):341–366, May 2020. doi:10.1007/s00354-020-00089-w.
- 4 Alexandre Baccouche, Kevin Montagne, Adrien Padirac, Teruo Fujii, and Yannick Rondelez. Dynamic DNA-toolbox reaction circuits: A walkthrough. *Methods*, 67(2):234–249, 2014. doi:10.1016/j.ymeth.2014.01.015.
- 5 Shulian Bai, Bangtian Xu, Jiangling Wu, and Guoming Xie. Series or parallel toehold-mediated strand displacement and its application in circular RNA detection and logic gates. *Biosensors and Bioelectronics*, 241:115677, 2023. doi:10.1016/j.bios.2023.115677.
- 6 Antonio Carlesso, Johanna Hörberg, Giuseppe Deganutti, Anna Reymer, and Pär Matsson. Structural dynamics of therapeutic nucleic acids with phosphorothioate backbone modifications. *NAR Genomics and Bioinformatics*, 6(2):lqae058, May 2024. doi:10.1093/nargab/lqae058.
- 7 Mathias Centola, Erik Poppleton, Sujay Ray, Martin Centola, Robb Welty, Julián Valero, Nils G. Walter, Petr Šulc, and Michael Famulok. A rhythmically pulsing leaf-spring DNA-origami nanoengine that drives a passive follower. *Nature Nanotechnology*, 19(2):226–236, February 2024. doi:10.1038/s41565-023-01516-x.
- 8 M. E. Craig, D. M. Crothers, and P. Doty. Relaxation kinetics of dimer formation by self complementary oligonucleotides. *Journal of Molecular Biology*, 62(2):383–401, December 1971. doi:10.1016/0022-2836(71)90434-7.
- 9 Constantine G. Evans and Erik Winfree. Physical principles for DNA tile self-assembly. *Chem. Soc. Rev.*, 46:3808–3829, 2017. doi:10.1039/C6CS00745G.
- 10 Teruo Fujii and Yannick Rondelez. Predator–prey molecular ecosystems. *ACS Nano*, 7(1):27–34, 2013. doi:10.1021/nn3043572.
- 11 Guillaume Gines, Rocío Espada, Adèle Dramé-Maigné, Alexandre Baccouche, Nicolas Larrouy, and Yannick Rondelez. Functional analysis of single enzymes combining programmable molecular circuits with droplet-based microfluidics. *Nature Nanotechnology*, 19(6):800–809, June 2024. doi:10.1038/s41565-024-01617-1.
- 12 Guillaume Gines, Roberta Menezes, Kaori Nara, Anne-Sophie Kirstetter, Valerie Taly, and Yannick Rondelez. Isothermal digital detection of microRNAs using background-free molecular circuit. *Science Advances*, 6(4):eaay5952, 2020. doi:10.1126/sciadv.aay5952.
- 13 Robert F. Johnson and Lulu Qian. Simplifying chemical reaction network implementations with two-stranded DNA building blocks. In Cody Geary and Matthew J. Patitz, editors, *26th International Conference on DNA Computing and Molecular Programming (DNA 26)*, volume 174 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:14, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.2020.2.
- 14 Ievgen Kurylo, Guillaume Gines, Yannick Rondelez, Yannick Coffinier, and Alexis Vlandas. Spatiotemporal control of DNA-based chemical reaction network via electrochemical activation in microfluidics. *Scientific Reports*, 8(1):6396, 2018. doi:10.1038/s41598-018-24659-7.
- 15 Matthew R. Lakin, Simon Youssef, Filippo Polo, Stephen Emmott, and Andrew Phillips. Visual DSD: a design and analysis tool for DNA strand displacement systems. *Bioinformatics*, 27(22):3211–3213, October 2011. doi:10.1093/bioinformatics/btr543.
- 16 Ronny Lorenz, Stephan H Bernhart, Christian Höner zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. ViennaRNA package 2.0. *Algorithms for Molecular Biology*, 6(1):26, 2011. doi:10.1186/1748-7188-6-26.
- 17 Jordan Lovrod, Boyan Beronov, Chenwei Zhang, Erik Winfree, and Anne Condon. Revisiting hybridization kinetics with improved elementary step simulation. In Ho-Lin Chen and Constantine G. Evans, editors, *29th International Conference on DNA Computing and Molecular Programming (DNA 29)*, volume 276 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:24, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.29.5.

- 18 Antoine Masurier, Rémi Sieskind, Guillaume Gines, and Yannick Rondelez. DNA circuit-based immunoassay for ultrasensitive protein pattern classification. *Analyst*, 149:5052–5062, 2024. doi:10.1039/D4AN00728J.
- 19 Kevin Montagne, Guillaume Gines, Teruo Fujii, and Yannick Rondelez. Boosting functionality of synthetic DNA circuits with tailored deactivation. *Nature Communications*, 7(1):13474, 2016. doi:10.1038/ncomms13474.
- 20 Kevin Montagne, Raphael Plasson, Yasuyuki Sakai, Teruo Fujii, and Yannick Rondelez. Programming an in vitro DNA oscillator using a molecular networking strategy. *Molecular Systems Biology*, 7(1):466, 2011. doi:10.1038/msb.2010.120.
- 21 Shu Okumura, Guillaume Gines, Nicolas Lobato-Dauzier, Alexandre Baccouche, Robin Deteix, Teruo Fujii, Yannick Rondelez, and Anthony J Genot. Nonlinear decision-making with enzymatic neural networks. *Nature*, 610(7932):496–501, 2022. doi:10.1038/s41586-022-05218-7.
- 22 Thomas E. Ouldrige, Petr Sulc, Flavio Romano, Jonathan P. K. Doye, and Ard A. Louis. DNA hybridization kinetics: zippering, internal displacement and sequence dependence. *Nucleic Acids Research*, 41(19):8886–8895, October 2013. doi:10.1093/nar/gkt687.
- 23 Adrien Padirac, Teruo Fujii, and Yannick Rondelez. Bottom-up construction of in vitro switchable memories. *Proceedings of the National Academy of Sciences*, 109(47):E3212–E3220, 2012. doi:10.1073/pnas.1212069109.
- 24 Ekaterina Protozanova, Peter Yakovchuk, and Maxim D. Frank-Kamenetskii. Stacked-unstacked equilibrium at the nick site of DNA. *Journal of Molecular Biology*, 342(3):775–785, September 2004. doi:10.1016/j.jmb.2004.07.075.
- 25 Jibin Abraham Punnoose, Kevin J. Thomas, Arun Richard Chandrasekaran, Javier Vilcapoma, Andrew Hayden, Kacey Kilpatrick, Sweta Vangaveti, Alan Chen, Thomas Banco, and Ken Halvorsen. High-throughput single-molecule quantification of individual base stacking energies in nucleic acids. *Nature Communications*, 14(1):631, February 2023. doi:10.1038/s41467-023-36373-8.
- 26 Nick A. Rejali, Felix D. Ye, Aisha M. Zwitter, Caroline C. Keller, and Carl T. Wittwer. Nearest-neighbour transition-state analysis for nucleic acid kinetics. *Nucleic Acids Research*, 49(8):4574–4585, May 2021. doi:10.1093/nar/gkab205.
- 27 Anna Romanov, Grant A. Knappe, Larance Ronsard, Christopher A. Cottrell, Yiming J. Zhang, Heikyung Suh, Lauren Duhamel, Marjan Omer, Asheley P. Chapman, Katie Spivakovsky, Patrick Skog, Claudia T. Flynn, Jeong Hyun Lee, Oleksandr Kalyuzhnyi, Alessia Liguori, Molly F. Parsons, Vanessa R. Lewis, Josue Canales, Boris Reizis, Ryan D. Tingle, Torben Schiffner, William R. Schief, Daniel Lingwood, Mark Bathe, and Darrell J. Irvine. DNA origami vaccines program antigen-focused germinal centers. *Science*, 391(6785):eadx6291, 2026. doi:10.1126/science.adx6291.
- 28 Yannick Rondelez. Competition for catalytic resources alters biological network dynamics. *Phys. Rev. Lett.*, 108:018102, January 2012. doi:10.1103/PhysRevLett.108.018102.
- 29 Yannick Rondelez and Guillaume Gines. Programmable ultrasensitive molecular amplifier for digital and multiplex microRNA quantification. *Methods in Molecular Biology*, 2630:89–102, 2023. Chapter. doi:10.1007/978-1-0716-2982-6_7.
- 30 John SantaLucia Jr and Donald Hicks. The thermodynamics of DNA structural motifs. *Annual Review of Biophysics*, 33:415–440, 2004. doi:10.1146/annurev.biophys.32.110601.141800.
- 31 Joseph M. Schaeffer, Chris Thachuk, and Erik Winfree. Stochastic simulation of the kinetics of multiple interacting nucleic acid strands. In *DNA*, 2013. URL: <https://api.semanticscholar.org/CorpusID:17591933>.
- 32 Vasily Shenshin, Camille Lescanne, Guillaume Gines, and Yannick Rondelez. A small-molecule chemical interface for molecular programs. *Nucleic Acids Research*, 49(13):7765–7774, 2021. doi:10.1093/nar/gkab470.

- 33 C.A. Stein, Chrisanthi Subasinghe, Kazuo Shinozuka, and Jack S. Cohen. Physicochemical properties of phosphorothioate oligodeoxynucleotides. *Nucleic Acids Research*, 16(8):3209–3221, April 1988. doi:10.1093/nar/16.8.3209.
- 34 Y. Wang, I. Baars, I. Berzina, I. Rocamonde-Lago, B. Shen, Y. Yang, M. Lolaico, J. Waldvogel, I. Smyrlaki, K. Zhu, R. A. Harris, and B. Högberg. A DNA robotic switch with regulated autonomous display of cytotoxic ligand nanopatterns. *Nature Nanotechnology*, 19(9):1366–1374, September 2024. doi:10.1038/s41565-024-01676-4.
- 35 Erik Winfree and Renat Bekbolatov. Proofreading tile sets: Error correction for algorithmic self-assembly. In Junghuei Chen and John Reif, editors, *DNA Computing*, pages 126–144, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. doi:10.1007/978-3-540-24628-2_13.
- 36 Damien Woods, David Doty, Cameron Myhrvold, Joy Hui, Felix Zhou, Peng Yin, and Erik Winfree. Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. *Nature*, 567(7748):366–372, March 2019. doi:10.1038/s41586-019-1014-9.
- 37 Peter Yakovchuk, Ekaterina Protozanova, and Maxim D. Frank-Kamenetskii. Base-stacking and base-pairing contributions into thermal stability of the DNA double helix. *Nucleic Acids Research*, 34(2):564–574, January 2006. doi:10.1093/nar/gkj454.
- 38 John N. Zadeh, Christopher D. Steenberg, Justin S. Bois, Blake R. Wolfe, Matthew B. Pierce, Abbas R. Khan, Raymond M. Dirks, and Niles A. Pierce. NUPACK: Analysis and design of nucleic acid systems. *Journal of Computational Chemistry*, 32(1):170–173, 2011. doi:10.1002/JCC.21596.
- 39 David Yu Zhang and Erik Winfree. Control of DNA strand displacement kinetics using toehold exchange. *Journal of the American Chemical Society*, 131(47):17303–17314, December 2009. doi:10.1021/ja906987s.
- 40 Fangzhou Zhao, Ziyi Sun, and Hanadi F. Sleiman. DNA-templated assembly of metalloprotein mimics with hydrolytic activity. *Journal of the American Chemical Society*, 147(40):36365–36375, October 2025. doi:10.1021/jacs.5c10056.

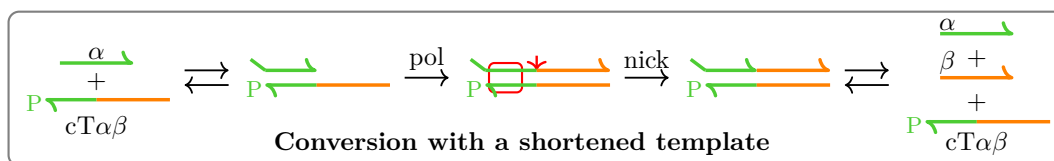
A Additional mechanisms and figures about the PEN-DNA toolbox

The mechanisms presented in Section 2 can incorporate additional features. For example, the converter template can act as an *irreversible* template, if it includes an extra domain between its input and output region (depicted in violet in Figure 6). In this configuration, once the polymerase extends the input strand, the template becomes “loaded”. Since the nickase cleaves after the extended domain, the resulting extended α strand (comprising green and violet domains) remains stably bound to the template at the working temperature. This irreversible template continuously produces the β output signal.

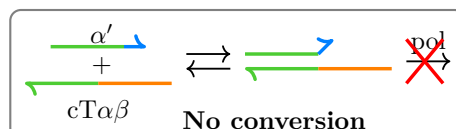


■ **Figure 6 Schematic of the conversion with irreversible template.** The converter template contains an extra domain (violet) that irreversibly binds the input once it is extended.

Another feature of the converter template involves its 3' end, which can be shortened by a couple of bases. While unbinding from shortened templates occurs faster than from standard templates, the overall binding is less thermodynamically favorable. Such shortened templates are typically used in inactivation processes, to favor pairing with the drain template. It is important to note that the converter template is usually protected by a 3' phosphorylation (represented by a “P” in Figure 7) to prevent the polymerase from re-extending the strand to its full length.



■ **Figure 7 Schematic of the conversion with a shortened template.** A P signifies a 3' phosphorylation, which prevents the polymerase from extending the template.



■ **Figure 8 Schematic of the interaction between an inactivated strand and a converter template.** Even though the strand can bind to the template, it cannot be extended by the polymerase since its 3' end (in blue) does not properly match the template.

B Kinetics supplement

As discussed in Section 4, deriving association rates by deriving them from dissociation rates via NUPACK may introduce errors. Given that K_D and k_{off} exhibit an exponential dependence on thermodynamic parameters, this approach can lead to incorrect behavior in k_{on} . For example, Figure 9 illustrates this exponential temperature dependency. Further exploration of these behaviors is available in the notebook `comparison_computation_kon_koff.ipynb` in the `resources/` folder in the Git repository.

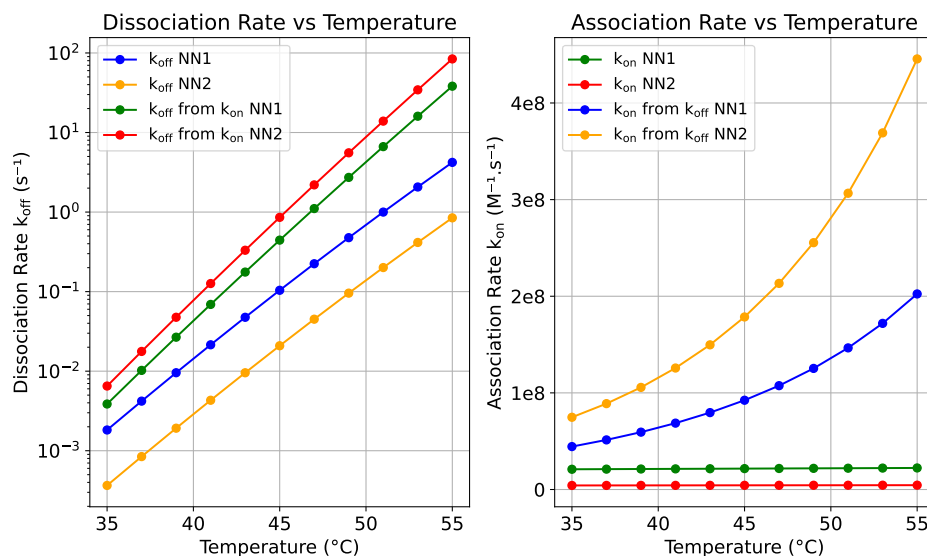
C Validation supplement

As explained in Section 6.1 and illustrated in Figure 10, the critical drain concentration required to achieve bistability increases with the drain template length. Experimentally, however, this behavior does not fully align with theory, as a single mismatched nucleotide may not be sufficient to completely prevent polymerase extension of the input signal [19].

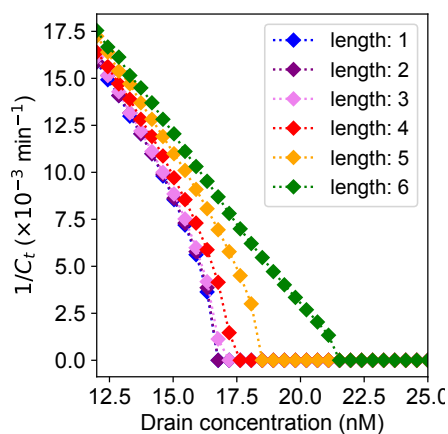
The simulation closely matches the measured behavior for 3'-shortened converter templates, as shown in Figure 11: the critical drain concentration decreases as the shortening size increases. However, even at 43°C (instead of 45°C), the system fails to self-start in the absence of the drain template for a shortening of 3 nucleotides. This is primarily because the input signal unbinds faster than the polymerase can extend it. This effect may be addressed in future work, once temperature-dependent enzyme activities will be incorporated.

D ODE construction

Following the schematic in Figure 1, the ODE function $f(x)$ is computed by decomposing each reaction into either (de)hybridization or enzymatic processes (see Section 4). This decomposition results in a system of equations, which are briefly outlined here. The rest of the full set of reactions is provided in the `resources/` folder in the Git repository, along with its modifications for complex features.



■ **Figure 9** Comparison of the four ways to compute the hybridization rates applied to the DNA sequence TAATCGGCAGGT. For dissociation (and association), two nearest-neighbor (NN) models from Rejali et al. [26] are employed, along with two additional models derived from the NN models for association (and dissociation), as explained in Section 4. NN1 refers to the model designed for 1 M NaCl, while NN2 refers to the model developed for 2.2 mM MgCl_2 . The association rates, when derived from the dissociation ones, suffer from an exponential dependency on the temperature.



■ **Figure 10** Simulated behavior of $1/C_t$ versus drain concentration for drain templates of varying lengths. The legend indicates the length of the inactivation tail of the drain templates. The critical drain concentration increases with drain template length, indicating reduced inactivation efficiency for longer templates.

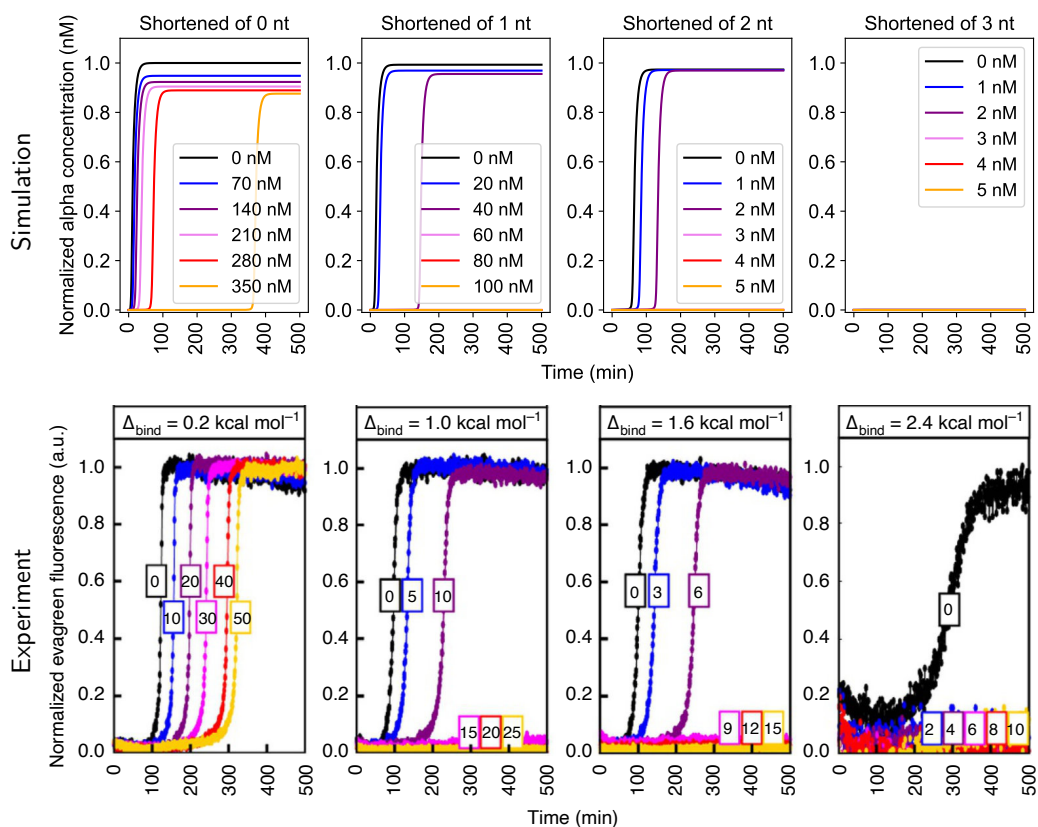


Figure 11 Normalized α concentration over time for templates of varying shortened lengths. The top panel shows results from our simulations (at 43°C), while the bottom panel presents experimental data adapted from Montagne et al. [19] (at 45°C). The critical drain concentration decreases with template length, reflecting enhanced inactivation efficiency for shorter templates.

We use the following notation:

- **s** denotes standalone signals, **s_{dr}** inactivated signals, **d** drain, and **tmp** stands for template, with **s_{in}** and **s_{out}** as its input and output signals.
- **[X]** denotes the concentration of species **X**.
- **k_{on}(X, Y)** and **k_{off}(X, Y)** are the binding and unbinding rates between species **X** and **Y**.
- **leak(tmp)** is the leak rate of template **tmp**.
- **stack(tmp)** is the unbinding penalty coming from the stability increase from the coaxial base-stacking interaction between the input and output signals. **stack** is computed as detailed in Section 4.
- **exo**, **nick**, and **pol** represent the effective enzymatic activities of exonuclease, nickase, and polymerase enzymes, respectively. **pol_displ** is the specific effective activity of the polymerase when displacing the output signal strand.
- **polV([polymerase], T) = [polymerase] · k_{cat}^{pol}**, where **k_{cat}^{pol}** is the catalytic rate constant of the polymerase.
- **polK** is the Michaelis constant of the polymerase and **polK_displ** the one for the polymerase when displacing the output signal strand.
- All concentrations and enzyme activities depend on time, but we omit the (**t**) term for readability.

alone signal equation:

$$\text{flux_in}(s) = \sum_{\substack{\text{tmp whose} \\ \text{input is s}}} \left(\begin{array}{ll} k_{\text{off}}(s, \text{tmp}) \cdot [\text{tmp}_{\text{in}}(s)] & (\text{s unbinds from in}) \\ + k_{\text{off}}(s, \text{tmp}) \cdot \text{stack}(\text{tmp}) \cdot [\text{tmp}_{\text{both}}(s)] & (\text{s unbinds from both}) \\ - k_{\text{on}}(s, \text{tmp}) \cdot [s] \cdot [\text{tmp}_{\text{alone}}] & (\text{s binds to alone}) \\ - k_{\text{on}}(s, \text{tmp}) \cdot [s] \cdot [\text{tmp}_{\text{out}}] & (\text{s binds to out}) \end{array} \right)$$

$$\text{flux_out}(s) = \sum_{\substack{\text{tmp whose} \\ \text{output is s}}} \left(\begin{array}{ll} k_{\text{off}}(s, \text{tmp}) \cdot [\text{tmp}_{\text{out}}(s)] & (\text{s unbinds from out}) \\ + k_{\text{off}}(s, \text{tmp}) \cdot \text{stack}(\text{tmp}) \cdot [\text{tmp}_{\text{both}}(s)] & (\text{s unbinds from both}) \\ - k_{\text{on}}(s, \text{tmp}) \cdot [s] \cdot [\text{tmp}_{\text{alone}}] & (\text{s binds to alone}) \\ - k_{\text{on}}(s, \text{tmp}) \cdot [s] \cdot [\text{tmp}_{\text{in}}] & (\text{s binds to in}) \\ + \text{pol_displ} \cdot [\text{tmp}_{\text{both}}] & (\text{pol displacement}) \\ + \text{leak}(\text{tmp}) \cdot \text{polV}([\text{pol}], T) \cdot [\text{tmp}_{\text{alone}}] & (\text{polymerase leak}) \end{array} \right)$$

$$\begin{aligned} \frac{d[s]}{dt} = & \text{flux_in}(s) + \text{flux_out}(s) && (\text{interactions with templates}) \\ & + \sum_{\substack{\text{d whose} \\ \text{input is s}}} \left(\begin{array}{ll} k_{\text{off}}(s, \text{d}) \cdot [\text{d}_{\text{in}}] & (\text{s unbinds from the drain}) \\ - k_{\text{on}}(s, \text{d}) \cdot [s] \cdot [\text{d}_{\text{alone}}] & (\text{s binds to the drain}) \end{array} \right) \\ & - \mathbb{1}_{\{\text{s not protected}\}} \cdot \text{exo} \cdot [s] && (\text{exonuclease degradation}) \end{aligned}$$

alone drain equation:

$$\begin{aligned}
 \frac{d}{dt}[\text{d}_{\text{alone}}] = & k_{\text{off}}(\text{s}_{\text{dr}}, \text{d}) \cdot [\text{d}_{\text{ext}}] && (\text{inactivated signal unbinding from drain}) \\
 & + k_{\text{off}}(\text{s}, \text{d}) \cdot [\text{d}_{\text{in}}] && (\text{signal unbinding from in}) \\
 & - k_{\text{on}}(\text{s}, \text{d}) \cdot [\text{s}] \cdot [\text{d}_{\text{alone}}] && (\text{signal binding to alone}) \\
 & - k_{\text{on}}(\text{s}_{\text{dr}}, \text{d}) \cdot [\text{d}_{\text{alone}}] \cdot [\text{s}_{\text{dr}}] && (\text{inactivated signal binding to alone}) \\
 & + \text{flux_out}(\text{d}) && (\text{created drains by conversion}) \\
 & - \mathbb{1}_{\{\text{d not protected}\}} \cdot \text{exo} \cdot [\text{d}_{\text{alone}}] && (\text{exonuclease degradation})
 \end{aligned}$$

both template equation (for reversible or irreversible):

$$\begin{aligned}
 \frac{d}{dt}[\text{tmp}_{\text{both}}] = & k_{\text{on}}(\text{s}_{\text{in}}, \text{tmp}) \cdot [\text{s}_{\text{in}}] \cdot [\text{tmp}_{\text{out}}] && (\text{input binding to out}) \\
 & + k_{\text{on}}(\text{s}_{\text{out}}, \text{tmp}) \cdot [\text{s}_{\text{out}}] \cdot [\text{tmp}_{\text{in}}] && (\text{output binding to in}) \\
 & - k_{\text{off}}(\text{s}_{\text{in}}, \text{tmp}) \cdot \text{stack}(\text{tmp}) \cdot [\text{tmp}_{\text{both}}] && (\text{input unbinding from both}) \\
 & - k_{\text{off}}(\text{s}_{\text{out}}, \text{tmp}) \cdot \text{stack}(\text{tmp}) \cdot [\text{tmp}_{\text{both}}] && (\text{output unbinding from both}) \\
 & - \text{pol_displ} \cdot [\text{tmp}_{\text{both}}] && (\text{polymerase binding to both}) \\
 & + \text{nick} \cdot [\text{tmp}_{\text{ext}}] && (\text{nicking enzyme binding to ext})
 \end{aligned}$$

Effective polymerase activity:

$$\text{pol} = \frac{\text{polV}([\text{polymerase}], T)}{\text{polK}(T) \cdot \left(1 + \sum_{\text{tmp}} \frac{[\text{tmp}_{\text{in}}]}{\text{polK}(T)} + \sum_{\text{d}} \frac{[\text{d}_{\text{in}}]}{\text{polK}(T)} + \sum_{\text{tmp}} \frac{[\text{tmp}_{\text{both}}]}{\text{polK_displ}(T)}\right)}$$

Enzymatic kinetic constants

■ **Table 1** Table of the enzymatic kinetic constants used in PENSIm, along with the temperature they have been measured at. (u.a.) means $\text{nM} \cdot \text{min}^{-1} \cdot (\text{units/mL})^{-1}$.

Enzymes	k_{cat}	K (nM)	Reference Temperature
pol	41.1 (u.a.)	80	42°C
pol_displ	8.2 (u.a.)	5.5	42°C
exo	6 min^{-1}	440	42°C
Nt.BstNBI	1.8 (u.a.)	30	42°C
Nb.BsmI	0.19 (u.a.)	9	45°C

Geometric Constraint Optimization for Localized Strand Displacement Reactions

Matthew R. Lakin¹   

Department of Computer Science, University of New Mexico, Albuquerque, NM, USA

Department of Chemical & Biological Engineering, University of New Mexico,
Albuquerque, NM, USA

Abstract

Localized molecular circuits offer practical advantages over those implemented using components freely diffusing in bulk solution, such as computation speed and component reuse. A common framework for implementing such circuits uses DNA strand displacement reactions with components localized via tethering to a DNA origami tile. While a number of papers have demonstrated the capability of such circuits, design tools for enumerating localized reactions and analyzing their behavior are relatively scarce. The key difficulty in modeling such circuits is that the geometric constraints imposed by the tethering of specific components at specific points on the tile surface are critical in determining whether or not a particular reaction may occur. In previous work, we deployed simple techniques based on random sampling of the structure space in an attempt to find geometric structures for candidate reaction products that satisfy all of the geometric constraints. In this paper, we show that this approach can be enhanced by using an optimization algorithm that takes initial guessed structures that fail to satisfy certain constraints and attempts to refine them into structures that do satisfy all of the constraints. We illustrate this approach on simple example reactions as well as a strand displacement-based signal transmission example from the literature. This work thus advances the state of the art in modeling tools for localized molecular circuits.

2012 ACM Subject Classification Computer systems organization → Molecular computing

Keywords and phrases Localized circuits, reaction enumeration, DNA strand displacement, constraint solving, geometry, molecular computing

Digital Object Identifier 10.4230/LIPIcs.DNA.32.5

Supplementary Material *Software (Source code)*: <https://github.com/matthewlakin/LocalizedEnumeratorOptimization> [24]

archived at `swb:1:dir:095f551c8228cf74f1548237675d8391b3963452`

Funding This material is based upon work supported by the National Science Foundation under grants 1518861, 1814906, and 2044838.

Acknowledgements The author thanks Abigail Carlson for early work on a preliminary version of the system reported in this paper.

1 Introduction

Molecular computing based on DNA strand displacement has been shown to be a powerful tool for implementing computation at the nanoscale. Networks of components diffusing freely in bulk solution have been used for many years to experimentally implement a range of computational paradigms, including digital logic circuits [38], population protocols [9], and recurrent neural networks [10, 39], including those trainable *in situ* [11].

In addition, several groups have developed variants of strand displacement systems in which some, or all, of the circuit components are localized to the surface of a DNA nanostructure. Typically this means the face of a two-dimensional DNA origami tile [40], where staple extension provides a simple means of programming the positions of individual

¹ Corresponding author



© Matthew R. Lakin;

licensed under Creative Commons License CC-BY 4.0

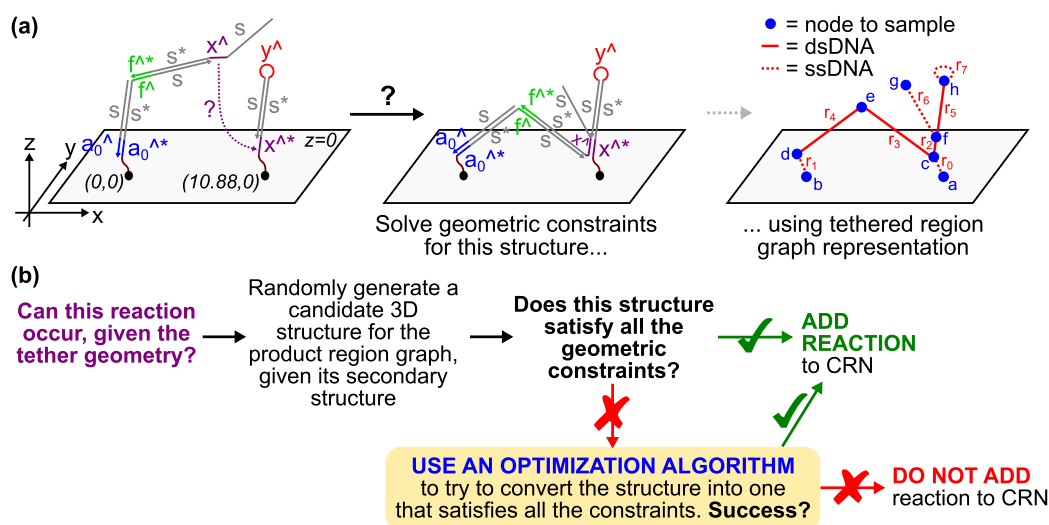
32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 5; pp. 5:1–5:23

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 1** Geometric reaction enumeration for tethered DNA strand displacement networks. (a) Our approach uses an algorithm to determine whether a particular localized DNA interaction is geometrically feasible: first the putative product secondary structure is determined, then a candidate three-dimensional physical structure for this product is generated, based on a “region graph” representation that translates the strand graph representation language into one more suited for geometric solving. If a geometrically plausible structure for the region graph can be found, we determine that the reaction may proceed based on geometry. (b) In this paper we improve the reliability of this process by taking a structure that initially fails this plausibility test and attempting to convert it into one that *is* plausible, using non-linear optimization routines.

components on the surface. In particular, previous work from the Seelig group [8,33] and the Reif group [5–7] has demonstrated the construction of multi-step signal processing cascades and logic circuits using hairpin-based strand displacement components. This approach to DNA computing offers potential advantages over the purely solution-phase approach. In particular, localized reactions can proceed more rapidly than solution-phase interactions because the reactants are tethered in close proximity, meaning that there are fewer degrees of freedom for reactant diffusion. Furthermore, nucleotide sequences can be reused between components because their position on the tile surface encodes their position in the circuit, unlike in solution-phase systems where each distinct logic gate must be implemented with sequences that are sufficiently distinct to prevent undesired crosstalk.

The issue of molecular geometry is typically not directly considered in reaction enumeration in solution-phase molecular circuits. The well-mixed assumption means that spatial considerations do not arise, and it is assumed that components may approach each other in whatever orientation is required in order to interact. For example, early versions of the Visual DSD system [28,31] restricted to a particular class of linear heteropolymer structures that, while expressive, did not allow for the kinds of structure where geometric constraints might stymie reaction progress, such as loops. Those were admitted in later versions of the tool that used a more expressive strand graph syntax [36], but certain geometric considerations were handled via *ad hoc* side conditions on reaction rules, for example, to prevent any binding interaction where one single-stranded domain is enclosed within a hairpin loop. This was not ideal because it failed to capture more subtle interactions of geometry with molecular design.

The collection of design tools available for modeling and analyzing the behavior of tethered DNA strand displacement systems is thus somewhat less than developed than that for more standard approaches to molecular computing. Some early work on the Visual DSD system

did introduce a syntax for representing tethered structures attached to DNA origami tiles, along with a manual system for annotating which components attached to the tile could interact with each other [27]. However, that approach is unsatisfactory because it places the burden for solving the key aspect of this reaction enumeration problem onto the user, namely, how to solve the geometric constraints inherent in tethered molecular circuits. In previous work, we tackled this problem using various approaches to automate the solving of the geometric constraint problems inherent in the enumeration of localized DNA strand displacement reaction networks. These constraints often boil down to determining whether two localized components are tethered close enough to each other on the surface to interact, as illustrated in Figure 1(a), though in the case of hairpin-based systems there are also often geometric constraints imposed by sequestering toeholds in a tight hairpin loop. In our earliest work [29], we used the Z3 SMT solver [14] to solve some instances of constraint problems arising in geometric reaction enumerators. However, that approach had performance issues due to the theoretically complete but sometimes extremely inefficient process of solving constraints modulo theories of real arithmetic [21].

We therefore adopted a conceptually simpler approach of repeated, randomized sampling of the structure space [23]. We used this approach to search for geometric structures for putative product species that satisfies the constraints implied by their secondary structures. Using a simple biophysical model, we successfully integrated this approach into a reaction enumerator over strand graphs for non-tethered species [22] and showed that it could successfully solve the geometric constraints involved in hairpin binding and remote toehold DNA strand displacement [19]. Most recently, we applied the same approach to a structure sampling enumerator for tethered DNA strand displacement systems [25]. However, the naïve approach to structure sampling taken in those papers means that it can sometimes take large numbers of attempts to find a structure that satisfies all of the constraints.

In this work, we attempt to mitigate some of the randomness in those approaches by using optimization algorithms to attempt to turn sampled structures that do not satisfy all of the constraints into ones that do. This approach is outlined in Figure 1(b). Our goal is to increase the probability of each attempt finding a satisfying structure, if one exists. We can then try fewer sampling attempts, though each one may do a lot more work. In the sections that follow, we define the key concepts and outline our implementation’s approach to optimization-based constraint solving. We then apply the algorithm to a simple tethered hybridization reaction and to both two- and three-stator versions of the hairpin-based signal propagation line of Chatterjee et al [8]. In particular, the three-stator variant is very difficult for the naïve structure sampling approach to enumerate because it involves two loop constraints that must be satisfied simultaneously. The results from our algorithm in the three-stator case also find a number of structures that are geometrically plausible in our biophysical model but which are not accounted for in the idealized chemical reaction networks reported for those systems. This suggests that localized strand displacement systems may be more susceptible to off-target interactions, driven by the proximity and flexibility of the reactants. Our work thus improves existing capabilities for enumeration of reaction networks for localized molecular computing systems and also suggests directions for future research into circuit designs.

2 Tethered geometric reaction enumeration

This paper builds upon our previous geometric reaction enumerators using structure sampling [22] for tethered strand graphs [25]. We recapitulate the key points below, and refer the reader to those papers for further details.

2.1 Tethered strand graphs

A tethered strand graph is a strand graph [36] in which some strands are annotated with a tether at either the 5' or 3' end. We assume that all tethers on a single strand are attached to the same tile; interactions between two tiles are thus prohibited. Each component tethered to a particular tile is represented as a single connected strand graph; we use a higher-level construct to group all tethers attached to a single tile. (Here we assume, as in our previous work [22], that vertices and sites in tethered strand graphs are indexed starting from zero; this contrasts with the original strand graph paper [36] where indexing started from one.) The fundamental definition of a tethered strand graph is presented below, reproduced from [25]:

► **Definition 1 (Tethered strand graphs.).** For $V \subseteq \mathbb{N}$ and $\text{length} : V \rightarrow \mathbb{N}$, we define the following helper functions:

- $\text{legal}_S(V, \text{length}) = \{(v, n) \in \mathbb{N}^2 \mid v \in V \wedge n \leq \text{length}(v)\}$ to be the set of legal sites, where a site $s = (v, n)$ is a pair of natural numbers such that v is a vertex (representing a single strand) and n is a position on that vertex (representing a sequence domain).
- $\text{legal}_E(V, \text{length}) = \{S \subseteq \text{legal}_S(V, \text{length}) \mid |S| = 2\}$ is the set of legal edges, where an edge $e = \{s_1, s_2\}$ is a set containing two sites, such that $s_1 \neq s_2$.

A **tethered strand graph** $G = (V, \text{length}, \text{color}, \text{tether5}', \text{tether3}', A, \text{toehold}, E)$, where:

- $V = \{1, \dots, N\}$ is the set of vertices; each vertex is a natural number.
- $\text{color} : V \rightarrow \mathbb{N}$ is a function assigning a natural number “color” to each vertex.
- $\text{length} : V \rightarrow \mathbb{N}$ is a function assigning a length to each vertex, such that $\text{color}(v_1) = \text{color}(v_2) \implies \text{length}(v_1) = \text{length}(v_2)$. That is, the length is really a function of the color.
- $\text{tether5}' : V \rightarrow ((\mathbb{N} \times \mathbb{N}) \uplus \{\perp\})$ and $\text{tether3}' : V \rightarrow ((\mathbb{N} \times \mathbb{N}) \uplus \{\perp\})$ are functions assigning either a pair of (x, y) -coordinates to the corresponding tethered strand terminus of each vertex, or $\perp$ if that strand terminus is not tethered, such that at most one of $\text{tether5}'(v)$ and $\text{tether3}'(v)$ is not $\perp$, for any v . That is, at most one terminus of each strand can be tethered. Note that vertices of the same color may have different tethers, as the tether is a property of the individual strand not the vertex color.
- $A \subseteq \text{legal}_E(V, \text{length})$ is a set of admissible edges, such that $\text{color}(e_1) = \text{color}(e_2) \implies (e_1 \in A \iff e_2 \in A)$ and such that $(\text{legal}_S(V, \text{length}), A)$ forms a bipartite graph, which ensures that admissible edges connect complementary domains. Furthermore, for an edge $e = \{(v_1, n_1), (v_2, n_2)\}$ we define its color to be: $\text{color}(e) = \{(\text{color}(v_1), n_1), (\text{color}(v_2), n_2)\}$.
- $\text{toehold} : A \rightarrow \mathbb{B}$ is a function that returns true on all admissible edges between toehold domains and false on all admissible edges on long domains, such that $\text{color}(a_1) = \text{color}(a_2) \implies \text{toehold}(a_1) = \text{toehold}(a_2)$.
- $E = \{e_1, \dots, e_M\} \subseteq A$ is a set of current edges, such that $(i \neq j) \implies e_i \cap e_j = \emptyset$. That is, at any given instant there can be at most one edge connected to any given site.

2.2 Geometric reaction enumeration for tethered strand graphs

We now define reaction enumeration over tethered strand graphs. The first component of this definition is a transition relation $G \rightarrow G'$ between pairs of tethered strand graphs; this is as defined in our previous work [25] and is reproduced as Figure A.1 for convenience. These rules specify the fundamental kinds of transition that may occur in our system: binding of complementary domains, toehold unbinding, and three- and four-way branch migration. Each rule has a premise of the form $\text{plausible}(G')$, which requires that the candidate product structure G' must be shown to be “geometrically plausible” for the transition to be permissible. To define the chemical reaction network (CRN) semantics, we distinguish two kinds of species:

- *free species* S are connected strand graphs that contain no tethers;
- *tile species* $[[\bar{S}]]$ represent a single DNA origami tile with attached components, where $\bar{S} = S_1, \dots, S_n$ is a finite collection of distinct, connected strand graphs, each of which contains at least one tether.

Then, on top of the $\rightarrow$ relation between individual tethered strand graphs we build a transition relation $\Rightarrow$ between collections of free or tile species. In addition to unimolecular reactions, we permit bimolecular interactions either between two free species as reactants, or between one free and one tile species. We do not permit interactions between two distinct tiles. Furthermore, uni- or bimolecular interactions between species tethered to the same tile are lifted to unimolecular reactions at the level of the tile species. The rules defining this relation were introduced in our previous work [25]; we reproduce them here as Figure A.2.

2.3 Forming geometric constraint sets from strand graphs

The first step toward forming the set of geometric constraints associated with a strand graph G is to form the associated region graph, $regiongraph(G) = (RGV, RGE)$. The rules defining this process are reproduced from our previous work [22] for convenience, as Appendix B. The region graph is an alternative representation of the structure in which vertices, rather than representing domains, now represent the *boundaries* between domains, at which the structure has flexibility. (Figure 1(a) illustrates the region graph produced for an example tethered structure.) We must provide a three-dimensional (x, y, z) coordinate for each vertex of the region graph in order to test for plausibility of the corresponding structure. The edges of the region graph represent contiguous segments of either single- or double-stranded DNA. In our simplified biophysical model, we assume that double-stranded regions are short enough relative to the persistence length of DNA duplexes to be modeled as rigid rods, whereas single-stranded regions are assumed to be infinitely flexible, as are all joints between regions.

We note that this process implicitly implements the “hybridization constraints” that we used in previous work [30], as domains hybridized to each other end up being collapsed into the same vertices of the region graph (which we consider acceptable because we neglect the width of the DNA double helix in our simplified biophysical model). Thus, constraints that remain to be solved may take five different forms:

- **Double-stranded region length constraints:** $eqdistconstr(N, N', d)$, which specifies that the distance between vertices N and N' must be precisely equal to d ;
- **Single-stranded region length constraints:** $maxdistconstr(N, N', d)$, which specifies that the distance between vertices N and N' must be at most d ;
- **Nick angle constraints:** $nickconstr(N, N', N'', \theta)$, which specifies that the magnitude of the angle of deviation from one of N or N' to the other, when passing through the nick, is less than or equal to θ (helpful when analyzing certain examples);
- **Tether location constraints:** $tetherconstr(N, (x, y))$, which specifies that the vertex N must be tethered at the coordinates $(x, y, 0)$ on the tile, whose surface is assumed to form the $z = 0$ plane of its local coordinate system; and
- **Tile constraints:** $zcoordconstr(N)$, which specifies that the z -coordinate of the vertex N must be at least zero, that is, not on the “wrong” side of the tile.

We now assume that the input strand graph G has been converted into the corresponding region graph, $regiongraph(G) = (RGV, RGE)$. To decide which kind of constraint to use for a given edge of a region graph, we write $dse(RGE)$ for the subset of edges from the region graph that correspond to *double-stranded* regions and $sse(RGE)$ for the subset of edges from the region graph that correspond to *single-stranded* regions. We write $edgelenh(e)$ for the

length in nanometers associated with the region graph edge e , which corresponds to the *total* length for double-stranded domains and the *maximum* length for single-stranded domains. Then, the constraint set for a given strand graph is as follows.

► **Definition 2** (Constraint sets for a given strand graph). *We define the set of constraints, $\mathcal{C}(G)$, that corresponds to the strand graph G , as follows:*

$$\begin{aligned} \mathcal{C}(G) = & \{eqdistconstr(N, N', d) \mid \{N, N'\} \in dse(RGE) \wedge edgelenh(\{N, N'\}) = d\} \cup \\ & \{maxdistconstr(N, N', d) \mid \{N, N'\} \in sse(RGE) \wedge edgelenh(\{N, N'\}) = d\} \cup \\ & \{nickconstr(N, N', N'', \theta) \mid \{N, N'\} \in dse(RGE) \wedge \{N', N''\} \in dse(RGE)\} \cup \\ & \{tetherconstr(N, (x, y, 0)) \mid ((v, 0), 5') \in N \wedge tether5'(v) = (x, y)\} \cup \\ & \{tetherconstr(N, (x, y, 0)) \mid ((v, length(v) - 1), 3') \in N \wedge tether3'(v) = (x, y)\} \cup \\ & \{zcoordconstr(N) \mid N \in RGV\} \end{aligned}$$

2.4 Constraint satisfaction for geometric constraint sets

A **geometric structure** $\mathcal{S}$ for a tethered strand graph G is a mapping from the set RGV of vertices in the corresponding region graph $regiongraph(G)$ to 3D Cartesian coordinates. If $\mathcal{S}$ maps vertex N to coordinate (x, y, z) , then we write $\mathcal{S}(N) = (x, y, z)$. A candidate structure $\mathcal{S}$ satisfies a finite set of constraints $\{c_1, \dots, c_n\}$, which we write as $\mathcal{S} \models \{c_1, \dots, c_n\}$, if $\mathcal{S}$ simultaneously satisfies each individual constraint c_i in the set. We overload notation and write this as $\mathcal{S} \models c$, where constraint satisfaction will be defined for each of the different kinds of constraint specified above. First, however, a few auxiliary functions: we write $dist(C_1, C_2)$ for the Euclidean distance between the 3D coordinates C_1 and C_2 . Also, we write $devangle(\mathcal{S}(C), \mathcal{S}(C'), \mathcal{S}(C''))$ for the magnitude of the angle between the vector $C' - C$ and $C'' - C$, when placed tail-to-tail. We can now define constraint satisfaction.

► **Definition 3** (Constraint satisfaction by geometric structures). *We define constraint satisfaction for the different kinds of constraint by cases, as follows.*

- If $c \equiv eqdistconstr(N, N', d)$ then $\mathcal{S} \models c \iff dist(\mathcal{S}(N), \mathcal{S}(N')) = d$.
- If $c \equiv maxdistconstr(N, N', d)$ then $\mathcal{S} \models c \iff 0 \leq dist(\mathcal{S}(N), \mathcal{S}(N')) \leq d$.
- If $c \equiv nickconstr(N, N', N'', \theta)$ then $\mathcal{S} \models c \iff 0 \leq devangle(\mathcal{S}(N), \mathcal{S}(N'), \mathcal{S}(N'')) \leq \theta$.
- If $c \equiv tetherconstr(N, (x, y, 0))$ then $\mathcal{S} \models c \iff \mathcal{S}(N) = (x, y, 0)$.
- If $c \equiv zcoordconstr(N)$ then $\mathcal{S} \models c \iff \exists x, y, z. \mathcal{S}(N) = (x, y, z) \wedge z \geq 0$.

We say that a tethered strand graph G is *plausible*, which we write as $plausible(G)$, iff there exists a geometric structure $\mathcal{S}$ such that $\mathcal{S} \models \mathcal{C}(G)$ holds. *The key contribution of this paper is improved algorithms for finding a satisfying geometric structure for a given region graph, and hence for testing geometric plausibility.*

3 Enhanced constraint solving over tethered region graphs

3.1 Energy functions over geometric structures

To allow for iterative improvement of an initially unsatisfying geometric structure, we need a metric to define how far away from satisfaction the current geometric structure actually is. Therefore, in this section we define an *energy function* over geometric structures for a given tethered strand graph with respect to its constraint set. We note that this is not an energy in the strict sense, but will merely serve as a loss function to be optimized over below.

► **Definition 4** (Energies of geometric structures). *Given a finite set of constraints $\{c_1, \dots, c_n\}$ and a geometric structure $\mathcal{S}$, we define the function energy as follows:*

$$\text{energy}(\mathcal{S}, \{c_1, \dots, c_n\}) = \sum_{i \in \{1, \dots, n\}} \text{energy}(\mathcal{S}, c_i)$$

where the energy associated with each individual constraint is defined by cases on the different kinds of constraint introduced above:

- If $c \equiv \text{eqdistconstr}(N, N', d)$ then $\text{energy}(\mathcal{S}, c) = (\text{dist}(\mathcal{S}(N), \mathcal{S}(N')) - d)^2$.
- If $c \equiv \text{maxdistconstr}(N, N', d)$ and $\text{dist}(\mathcal{S}(N), \mathcal{S}(N')) \leq d$ then $\text{energy}(\mathcal{S}, c) = 0$. Otherwise, $\text{energy}(\mathcal{S}, c) = (\text{dist}(\mathcal{S}(N), \mathcal{S}(N')) - d)^2$.
- If $c \equiv \text{nickconstr}(N, N', N'', \theta)$ and $\text{devangle}(\mathcal{S}(N), \mathcal{S}(N'), \mathcal{S}(N'')) \leq \theta$ then $\text{energy}(\mathcal{S}, c) = 0$. Otherwise, $\text{energy}(\mathcal{S}, c) = (\text{devangle}(\mathcal{S}(N), \mathcal{S}(N'), \mathcal{S}(N'')) - \theta)^2$.
- If $c \equiv \text{tetherconstr}(N, (x, y, 0))$ then $\text{energy}(\mathcal{S}, c) = \text{dist}(\mathcal{S}(N), (x, y, 0))^2$.
- If $c \equiv \text{zcoordconstr}(N)$ and $\mathcal{S}(N) = (x, y, z)$ and $z \geq 0$ then $\text{energy}(\mathcal{S}, c) = 0$. If $z < 0$ then $\text{energy}(\mathcal{S}, c) = 10^{100}$.

► **Lemma 5** (Energies are non-negative). *For all finite constraint sets $\{c_1, \dots, c_n\}$ and geometric structures $\mathcal{S}$:*

$$\text{energy}(\mathcal{S}, \{c_1, \dots, c_n\}) \geq 0.$$

Proof. This follows straightforwardly from the fact that the total energy is the sum of the energies associated with each individual constraint, $\text{energy}(\mathcal{S}, c_i)$. Each of these is either positive or zero, which can be seen by case analysis over constraints as defined above. ◀

► **Lemma 6** (Satisfaction iff energy is zero). *For all finite constraint sets $\{c_1, \dots, c_n\}$ and geometric structures $\mathcal{S}$:*

$$\mathcal{S} \models \{c_1, \dots, c_n\} \iff \text{energy}(\mathcal{S}, \{c_1, \dots, c_n\}) = 0.$$

Proof. By Lemma 5, the energy values for individual constraints cannot be negative. Therefore, the only way the energy of the entire constraint set can be zero is when the energy associated with every individual constraint is zero. We then proceed by case analysis on the different kinds of constraint, comparing the relevant cases from the definitions of constraint satisfaction by $\mathcal{S}$ and of the energy function. If a constraint involves an inequality, then the energy is zero in the region satisfying the inequality and the square of the magnitude by which the inequality is violated. For constraints involving an equality, the energy is just the square of the magnitude of the error when comparing the actual result of the calculation to that required to satisfy the constraint. Thus, in every case, the constraint is satisfied precisely when the energy function evaluates to zero, which gives the result. ◀

Given the properties established above, it follows that minimizing such an energy function over a strand graph using a numerical optimization routine thus provides a possible route to convert an unsatisfying geometric structure into a satisfying one.

3.2 Optimizing energies of sampled structures

Given an energy function defined over structures for a given set of geometric constraints, we can consider optimizing these structures in an attempt to minimize the energies to zero which, by Lemma 6, would correspond to a structure that simultaneously satisfies all of the constraints. This raises the question of how to choose the starting structure for the

optimization process. Here we draw on our previous work on structure sampling over region graph structures [22, 23, 25], which yields structures that satisfy most of the constraints by construction, leaving some constraints (those that close a loop, either explicitly in the strand graph or implicitly via the underlying origami tile) to be checked *post hoc*.

We refer the reader to those papers for the details of this process; briefly, sampling begins from a randomly chosen graph of maximum degree (in the case of free species) or a randomly chosen tether (in the case of a species tethered to a tile). An adjacent region is chosen at random to be sampled for its directionality unit vector and, in the case of single-stranded regions, its length. (The length of double-stranded regions is fixed.) Double-stranded regions are always sampled before single-stranded regions, when available, as the single-stranded regions are the ones that can “take up the slack” to enable constraints to be satisfied when a loop is eventually closed. Crucially, this iterative sampling process determines, as an offshoot of its operation, an ordering on the regions that will be critical in our approach below.

In early experiments on this system we generated an initial structure via sampling and, if that structure failed to satisfy all of the constraints, ran an optimization procedure on the (x, y, z) -coordinates of the region graph vertices directly. While this could, in some cases, produce energies very close to zero, the resulting structures never actually satisfied the original constraints. This was because the energy penalty for modifying the length of a (supposedly fixed-length) double-stranded region was often worth paying to minimize the overall energy of the system. We therefore could not sufficiently constrain the lengths of these double-stranded regions so that the resulting structure was geometrically plausible without subsequent adjustment.

This led us to reimagine our optimization algorithm by lifting it to the level of structure sampling operations. Rather than operating on the coordinates of the structure directly, the optimization variables became those that guide the structure sampling process, *assuming a particular ordering on the sampled regions*. To specify this process, here we reformulate the structure sampling process in a slightly more formal manner in terms of *sampling actions*.

► **Definition 7** (Sampling actions.). *A sampling action A is one of the following:*

- **fixVertex** (N, C) , *which means that the 3D Cartesian coordinates of region graph vertex N are set to be C . This is used to establish which vertex is at the origin in the case of a free species, and to establish which vertices are tethers in the case of a tile species;*
- **doubleStranded** (N, v, d, N') , *which means that the coordinates for the unsampled vertex N' are obtained from those from the previously-sampled vertex N by moving along the sampled directionality unit vector v for a distance d equal to the fixed length of the corresponding double-stranded region linking N to N' ; and*
- **singleStranded** $(N, v, d, d_{\max}, N')$, *which means that the coordinates for the unsampled vertex N' are obtained from those from the previously-sampled vertex N by moving along the sampled directionality unit vector v for a distance d sampled from between zero and the maximum length $d_{\max}$ of the corresponding single-stranded region linking N to N' .*

Using the structure sampling algorithms from our previous work [25], it follows that a single structure sampling run over a region graph $\text{regiongraph}(G) = (RGV, RGE)$ produces a randomly chosen ordered list $\bar{A}$ of sampling actions, such that every vertex $N^* \in RGV$ is sampled precisely once – that is, appears in precisely one sampling action in $\bar{A}$ of the form **fixVertex** (N^*, C) for some C , or of the form **doubleStranded** (N, v, d, N^*) or **singleStranded** $(N, v, d, d_{\max}, N^*)$ for some $N, v, d, d_{\max}$.

Given a region graph and a sequence of sampling actions generated from it during a structure sampling attempt, the corresponding geometric structure for that region graph can be reconstructed. Furthermore, from this sequence of sampling actions an ordered sequence

of floating-point parameters can be extracted that is suitable as input to an optimization algorithm. This sequence can be derived by traversing the sequence $\bar{A}$ and appending the appropriate parameters for each sampling action to the sequence in turn, as follows:

- For a sampling action `fixVertex( $N, C$ )`, no new parameters are added.
- For a sampling action `doubleStranded( $N, v, d, N'$ )`, new parameters $[v_x, v_y, v_z]$, the components of the directionality vector v , are added.
- For a sampling action `singleStranded( $N, v, d, d_{max}, N'$ )`, new parameters $[v_x, v_y, v_z, d]$, the ordered components of the directionality vector v and the sampled length d , are added.

Via a similar mechanism, bounds for these parameters can be extracted: for the components of directionality unit vectors, the lower bounds can be set to -1 and the upper bounds can be set to $+1$. For the sampled lengths of single-stranded regions, the lower bound is 0 and the upper bound is the d_{max} value from the sampling action.

The key advantage of this approach is that, by modifying these parameters within their bounds, our optimization attempts tend to preserve satisfiability of as many of the individual geometric constraints as possible. This is because instead of optimizing the coordinates directly, we optimize the underlying structural parameters from which the coordinates are derived. In particular, the fixed lengths of double-stranded domains are not modified. This allows us to run the optimization routine over an initial failed structure sampling attempt before taking the resulting sequence of modified parameters and, in conjunction with the original sequence of sampling actions, reconstructing the corresponding geometric structure for these optimized parameters. We can then recheck constraint satisfaction to confirm that the new structure is indeed geometrically plausible.

4 Results

We implemented a prototype of the algorithm presented above using the Python programming language. In this section, we present results obtained from testing its capabilities on several example systems. Code for our prototype is available online under an open source license from <https://github.com/matthewlakin/LocalizedEnumeratorOptimization>. In these tests, we used the following biophysical parameters: the length per nucleotide in double-stranded regions was 0.34 nm and the length per nucleotide in single-stranded regions was 0.68 nm . These constant values have been used by other similar studies in the literature [12, 19].

4.1 Simple tethered hybridization example

We began by testing our system on a simple example comprising two tethered strands with complementary domains that may or may not hybridize, depending on the relative positions of the two tethers. This example is illustrated in Figure 2(a), including tethered process calculus-style code and a description of the domain lengths (5nt for the spacer domain `spcr` and 30nt for the `x` binding domain). In this example, a single reaction may be possible, namely, the irreversible binding reaction of the complementary domains, `x` and `x*`. This makes it a straightforward initial test of our new constraint solving approach.

As a baseline, we benchmarked our enhanced system against the system from our previous work [25], which relies solely on structure sampling to try to find satisfying structures for a candidate reaction product. In that approach, structure sampling is carried out as outlined above, except that a sampled structure that fails to satisfy the constraints is immediately discarded and another sampling attempt is made, until the maximum number of allowed attempts is reached. Here, we aimed to contrast the sampling-only approach with our optimization-based approach, so we deliberately used smaller numbers of sampling attempts (10, 25, or 50) than in our previous work. In all cases, we tested inter-tether distances

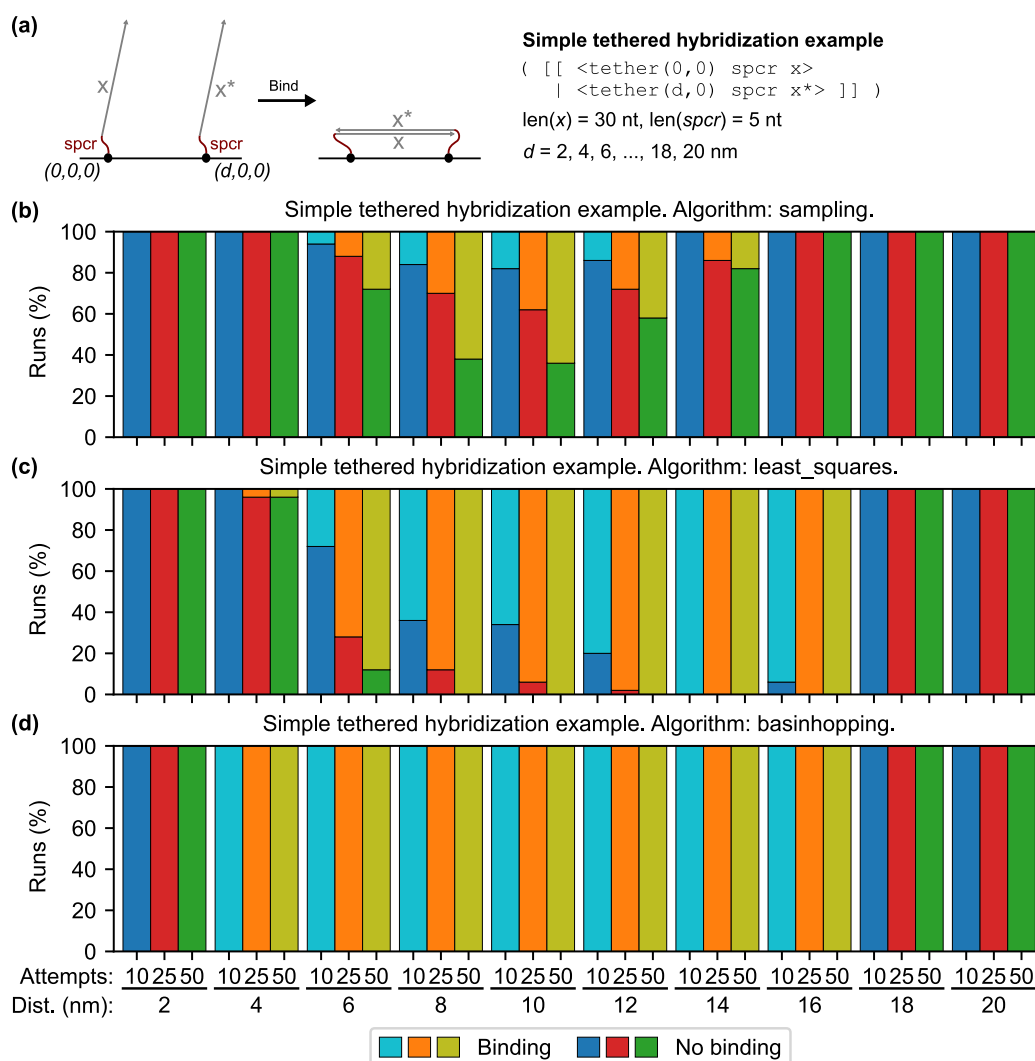


Figure 2 Testing geometric constraint optimizers on a simple tethered hybridization example. (a) System outline, with process calculus-style code and other parameters such as domain lengths in nucleotides. (b) Summary of enumeration results using the sampling-only approach. For each inter-tether distance and number of attempts, 50 runs were carried out. In each case, the percentages of runs in which the binding reaction was or was not enumerated are plotted as stacked bars. (c) Similar summary of enumeration results using structure optimization using the `least_squares` method. (d) Similar summary of enumeration results using structure optimization using the `basinhopping` method. This approach proved the most reliable, with a 100% success rate.

between 2 nm and 20 nm in increments of 2 nm. We generated 100 distinct pseudo-random number generator seeds for each test and computed statistics from these runs to study the reliability of the results.

Perhaps unsurprisingly, with so few sampling attempts permitted the sampling-only system failed to reliably enumerate the binding reaction: only toward the middle of the tested region (8 and 10 nm inter-tether distances) did it achieve greater than 50% success (Figure 2(b)). Given the maximum lengths of the single-stranded spacers, satisfying structures should exist for inter-tether distances between 3.4 and 17 nm.

We next compared this to an optimization-based reaction enumerator using the approach outlined above. In our first tests, we used an optimization algorithm based on a non-linear least-squares minimizer: the `least_squares` function from the `scipy.optimize` Python library [45], using the “Trust Region Reflective” algorithm. This approach can directly handle problems with constraints on the variables, and we used this capability to constrain the vectors and single-stranded region lengths as outlined above. The results using this approach (Figure 2(c)) were much more reliable than those from the sampling-only approach, showing 100% (or close to 100%) enumeration of the binding reaction across much of the parameter space where we would expect solutions to exist. However, particularly for lower numbers of attempts and shorter inter-tether distances, the algorithm often failed to find a satisfying structure. We attribute this to the fact that the solutions for shorter inter-tether distances involve the long double-stranded region and the two single-stranded spacers pointing in opposite directions, so that the spacers can accommodate the fixed-length duplex. This, coupled with the fact that the `least_squares` approach takes only single steps along optimal gradients, makes it prone to getting stuck in local minima for such problems.

Therefore, we tested a second optimization approach, using a global minimizer: the `basinhopping` function from `scipy.optimize`. This algorithm uses larger steps with a Metropolis acceptance criterion to escape local minima, coupled with subsequent local minimization using the BFGS algorithm [46]. We implemented a callback function that immediately exits the solver if a solution is found with energy zero, rather than trying to optimize further. This algorithm does not handle variable constraints directly, so we added these as additional energies: if a variable satisfies the bounds outlined above, its energy is zero, if not, its energy is 10^{100} . This approach was able to find a satisfying structure in 100% of the runs where solutions exist, even with only 10 sampling attempts permitted. We note that here, as in the case of the other algorithms tested, no satisfying structures were found where they were not expected. This shows that our system is better at finding solutions that exist, and does not incorrectly identify any non-solutions as satisfying the constraints.

In summary, these results demonstrate that an optimization-based approach to constraint solving, with a judicious choice of optimization algorithm, can greatly improve the reliability of enumeration results from sampling-based geometric constraint solving.

4.2 Tethered signal transmission line example

We next tested our approach on the tethered signal transmission line system reported by Chatterjee et al. [8]. In this design, a sequence of hairpins are tethered to a DNA origami tile, with a second set of fuel hairpins freely diffusing in solution. That paper reported several transmission lines as well as multi-hairpin logic gates. The two-stator variant of the transmission line system is illustrated in Figure 3, along with the expected reactions. When the first stator is opened via strand displacement by an input strand, it can in turn open a fuel hairpin, which may then bind and open the second stator in the transmission line, and so on. In this system, we consider the two inter-stator distances studied in the original paper: 10.88 nm for the single-spaced variant (also referred to as “1x” spacing below) and 21.76 nm for the double-spaced (“2x”) variant. The domain lengths outlined in Figure 3 match those used in the paper, and the system is designed such that an opened stator with a bound fuel hairpin can interact with the next stator in the single-spaced case but *not* in the double-spaced case. Thus, in the double-spaced case, only the first eight reversible reactions in Figure 3 (those above the broken red line) should be possible. The three-stator system behaves similarly, except that the second stator has an f^* looped toehold rather than a y^* toehold, which means it can only interact with another fuel hairpin. The diffusing, double-stranded reporter complex can only interact with the third stator via *its* y^* toehold.

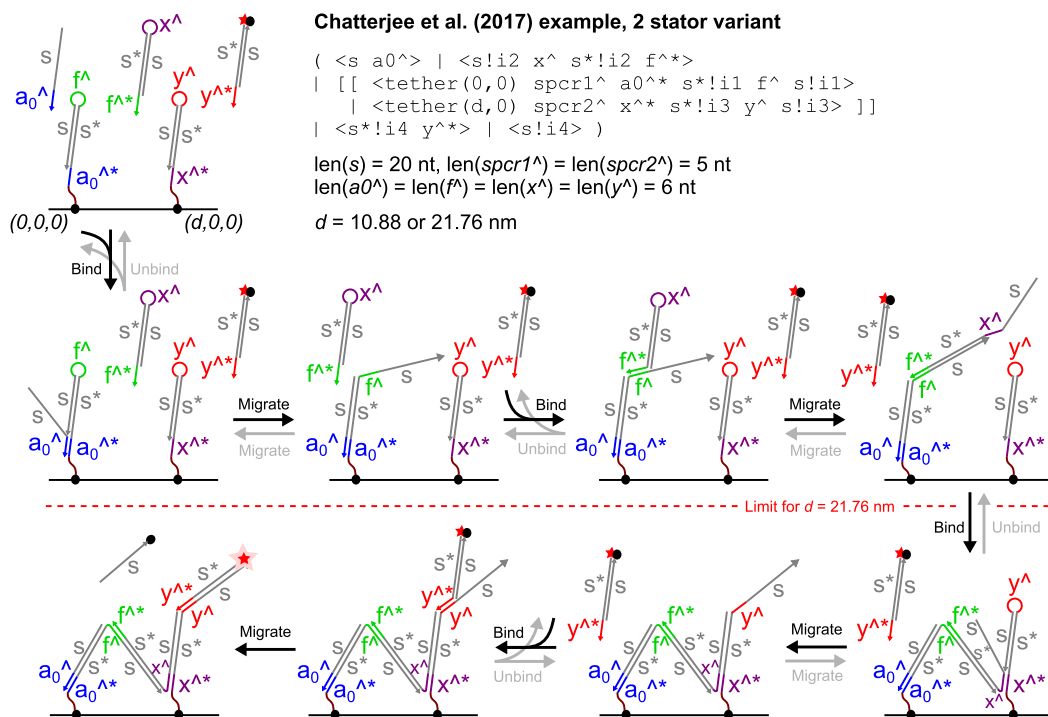


Figure 3 System outline of tethered signal transmission line of Chatterjee et al. [8], including a summary of expected reactions for single- and double-spaced variants of the two-stator system. Domain lengths used here match those used in that paper’s experimental work.

We used the same three approaches to constraint solving described in Section 4.1 above. We tested each 100 times on both the two- and three-stator systems, for 10, 25, and 50 attempts. We also compared the single-spaced (“1x”) and double-spaced (“2x”) variants of each system. In this example there is another parameter to consider: the maximum allowed angle of deviation between two double-stranded regions that meet at a nick. In previous work [25], we introduced this as a parameter that can be used to tune the number of “unexpected” species uncovered by a geometric reaction enumerator such as ours. Here, in all tests we used the value of 105° for the maximum nick angle. In general, larger nick angle parameters are more permissive and lead to more species being enumerated.

We first analyzed the likelihood of each constraint solver to fully enumerate the complete cascade in each case. We expect that the cascade should never complete when the tethers are double-spaced, but for single-spaced tethers both the two- and three-stator systems can theoretically complete. We determined cascade completion by searching the enumerated CRN for an irreversible reaction that displaces the product strand $\langle s \rangle$: the only way this can happen is if the final stator is opened and then activates the reporter complex. These results are presented in Figure 4, which shows that the sampling-only approach does not successfully enumerate the entire cascade in any case. The optimizer using the `least_squares` algorithm did better on the two-stator system, with over 80% success when it was permitted 50 attempts. However, it failed entirely on the three-stator system, which we attribute to the same problem with local minima that we noted above. This is exacerbated in the three-tether transmission line system because there are more ways for the double-stranded regions to fold back on themselves at nicks and create local minima. By contrast, the `basinhopping` algorithm

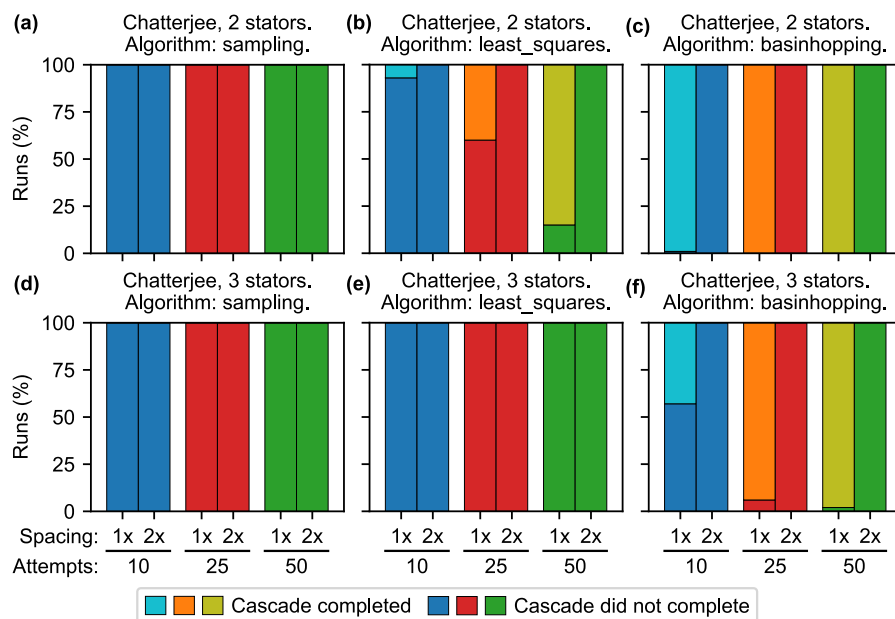


Figure 4 Statistics on cascade completion in 100 attempts to enumerate reaction networks for the Chatterjee et al. tethered transmission line system, outlined in Figure 3. Here and below, results are shown for systems comprising two ((a), (b), (c)) or three ((d), (e), (f)) stators. As in Figure 2, three approaches are compared: sampling only ((a), (d)) and structure optimization using the `least_squares` algorithm ((b), (e)) or the `basinhopping` algorithm ((c), (f)).

correctly enumerated the entire cascade in virtually all single-spaced cases of the two-stator system. Similar success was seen for the three-stator system when 25 or 50 attempts were allowed. Again, no spurious cascade completions were observed for double-spaced stators.

We next determined how often the expected number of reactions was enumerated; these results are presented in Figure C.3 in the Appendix. In almost all cases, the correct number of reactions was observed in the double-spaced case. For the single-spaced two-stator system, the `least_squares` system outperformed sampling only, and `basinhopping` did better again, with almost 100% success. However, in only around 5% of `basinhopping` tests was the expected number of reactions enumerated for the single-spaced three-stator system.

We sought to explain this result by plotting the distributions of the numbers of reactions enumerated in each case. That data is presented in Figure 5 and shows that the sampling-only and `least_squares` optimizers erred on the side of too few reactions in both the two- and three-stator cases. The `basinhopping` optimizer obtained the expected result in virtually all two-stator tests but found more reactions than expected in most cases for the single-spaced three-stator system. Upon inspection, we found that the third stator enables a number of unexpected reaction products to form, which do not feature in the canonical CRN.

A reaction producing such structures is illustrated in Figure 6(a,b). Even with the maximum nick angle limited to 105° , the three-stator system is able to fold back on itself to carry out such reactions. We note, however, that the fact that many of these species are not always enumerated even by the `basinhopping` algorithm indicates that they may be close to the limit of geometric plausibility, which would in turn suggest that their formation may be low probability events in experiments. Regardless, these results illustrate the potential of our approach for discovering potential unanticipated side-products, so that additional design work to eliminate them may be carried out if needed, prior to experimental testing.

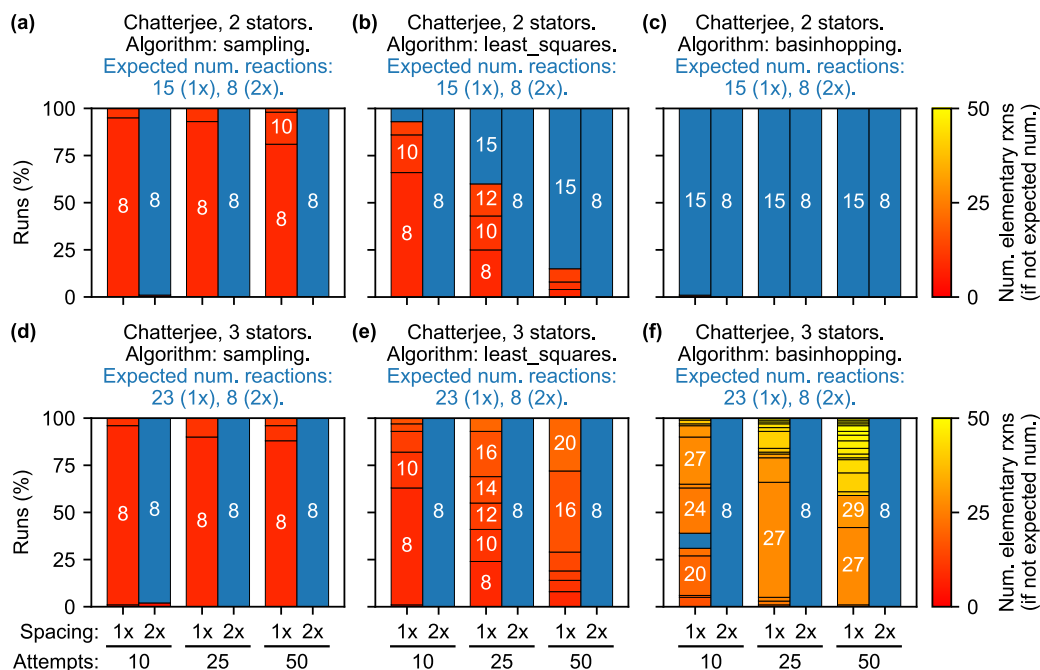


Figure 5 Distributions of numbers of reactions found in 100 attempts to enumerate reaction networks for the Chatterjee et al. tethered transmission line system, outlined in Figure 3. In each case, the numbers of enumeration runs resulting in each number of reactions are plotted as stacked bars, with those producing the smallest number of reactions at the bottom. Bars are color-coded according to the colorbars except for the bar representing the expected number of reactions, which is colored blue. Bars of sufficient size are also labeled with the corresponding number of reactions.

5 Discussion

In summary, we have developed an optimization algorithm that attempts to convert a sampled structure that initially fails to satisfy one or more constraints into one that satisfies them all. Our results show that this can enhance the reliability of geometric reaction enumerators for tethered DNA reaction networks while requiring fewer sampling attempts. This improves upon our previous work in this area [25] and may serve as the basis for future design tools for localized DNA systems, including molecular computers, walkers, and robots.

5.1 Performance Considerations

Here we focused on improving the *reliability* of geometric reaction enumeration. Indeed, our results showed that we can achieve more consistent results with fewer structure sampling attempts than in previous work [22, 25], in particular if the `basinhopping` optimization algorithm is employed. However, the total time taken for enumeration is typically longer, because far more work is carried out in an attempt to salvage each structure sample before it is discarded. We believe that this is a worthwhile tradeoff to obtain a more consistent output from the constraint solver. In this paper we present the statistics from 100 attempts to enumerate each variant of our example systems, though a user might in practice employ some kind of majority voting to settle on a consensus model. Alternatively, a “just-in-time” compilation algorithm such as that employed in our previous work for efficient enumeration and stochastic simulation of very large CRNs [26] or polymerizing CRNs with a potentially infinite number of species and reactions [28] could defer the costs of reaction enumeration.

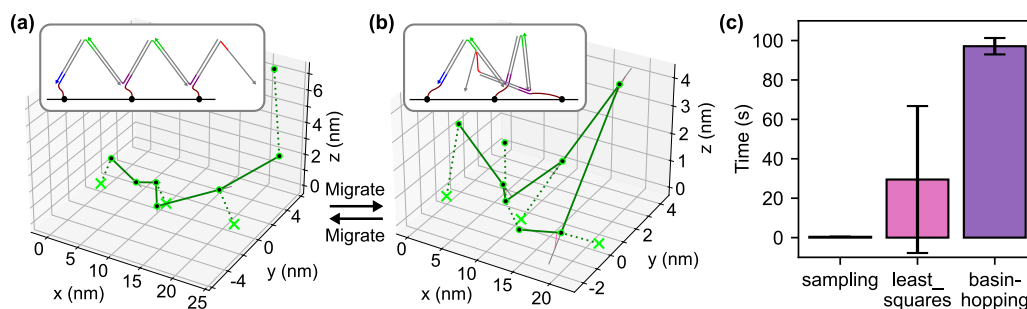


Figure 6 Further analysis of results and performance for the transmission line example of Chatterjee et al. [8]. (a),(b) Example of an unintended interaction in the single-spaced, three-stator system. When the third stator hairpin is opened, as in (a), and has a fuel hairpin attached, the s domain of the fuel hairpin is able to displace the corresponding domain from the first stator, as in (b). Insets show secondary structure cartoons in a two-dimensional projection; main plots show satisfying 3D geometric structures found by our constraint solver using the **basinhopping** optimization algorithm. In the 3D plots, solid lines are double-stranded regions and broken lines are single-stranded. Crosses are tethers; circles are non-tether vertices. Nick angles are annotated in pink, though the projection means that they are not all visible. (c) Average runtimes for all three algorithms applied to the single-spaced, two-stator variant of the tethered transmission line example. In each case, 10 sampling attempts were permitted per species and, as above, the maximum nick angle was 105° . Bars are average runtimes for 100 distinct seeds; errorbars show the standard deviation. Tests were carried out on an Apple MacBook Pro with M1 Pro CPU and 32GB RAM.

Figure 6(c) shows the average time per enumeration for the single-spaced, two-stator transmission line example [8]. The time taken by the sampling-only approach is significantly lower, not just because it does less work but also because it fails to enumerate most of the reactions (Figure 5(a)). The tradeoff for longer runtime is the significant reliability gains from the **basinhopping** approach. While, in the two-stator case, increasing the number of attempts can produce more reliable results from a sampling-only system [25], the same is not true for the three-stator case. Thus, our enhanced approach has significant advantages when we attempt to scale to larger systems involving more tethers, more loops, and more ways to violate constraints. The time taken for the **least_squares** algorithm falls in between as it does less work than the **basinhopping** algorithm; the high variance arises from the fact that for some seeds it fails to enumerate the interaction with the second stator (Figure 5(c)).

Other approaches to improving efficiency could include better structure sampling approaches that look for subgraph isomorphisms between a candidate structure and existing species known to be geometrically plausible, so that the corresponding parts of the new species can begin with their constraints already satisfied. Subdividing structures appropriately could be challenging in general, but splitting based on routes to different tethers might provide a general-purpose approach. We note, however, that even if subgraph constraints can be solved independently, there would still need to be a final integration and verification step to confirm that the structure as a whole is still geometrically plausible. This could yield some gains, though much of the effort would still be spent failing to find solutions for structures that are not geometrically plausible. Therefore, hard-coding short-hand checks for common failure cases, such as looped double-stranded domains, might yield greater improvements.

Because we use the domain-level modeling abstraction common in many design tools [1,32], there is the possibility that an intermediate reaction state, e.g., partially displaced along a duplex domain, might be implausible while the initial and final states are both plausible. Such a reaction would be enumerated by our system even though it could not occur in reality. While theoretically possible, we believe this to be highly unlikely in practice.

5.2 Related and Future Work

The approach developed here draws from a long history of related work on design tools for solution-phase DNA computing systems. A range of such tools exist, including Peppercorn [1], Piperine [42], the Nuskell system [2], KinDA [3], the DyNAMiC Workbench [20], and the Visual DSD system [32, 41] offer a number of options for specifying a system design as a structural description and compiling that description into a kinetic model by enumerating all possible reactions that could occur given those starting species. Considering tethered molecular systems more generally, the work of Dalchau et al. [12] and Dannenberg et al. [13] analyzed specific circuit motifs; the goal of the work outlined here is to generalize and automate such analyses so that they may be applied to arbitrary strand graph reaction systems without specific foreknowledge. We draw inspiration from *distance matrix* approaches [44], though our problem is more complex because single-stranded domains are of variable length.

Our tools could be integrated with DNA sequence designers such as NUPACK [47, 48] and Piperine [42], and with DNA origami design tools such as *cadNAno* [16] and *scadnano* [15]. This could enable staple and oligonucleotide sequences for specific designs to be derived, easing the transition to experimental validation of systems that integrate molecular computing and self-assembly, such as signal-passing [35] and reconfigurable DNA structures [18, 49].

Although our results do agree qualitatively with experimental work on the tethered transmission line system [8], further work could involve comparing the predictions from our system with those from other experiments and from coarse-grained models. Integration with other systems, such as *oxDNA* [4, 17, 34, 37, 43], could inform improvements to the biophysical model that underlies our constraint solving system. We aimed to keep the constraint problems as simple as possible by abstracting away much of the molecular detail: reducing double-strands to rigid rods and neglecting the width of a duplex. However, we could also preserve the rigid rod abstraction while reintroducing some more realistic aspects, e.g., explicitly modeling the width and double-helical structure of duplexes.

We note that our approach does not *guarantee* that all plausible species will be enumerated, but our results support the claim that it greatly increases the probability that if a plausible structure exists, it will be found. Nevertheless, further work will be needed to fully understand the system’s sensitivity to settings such as the chosen optimization algorithm and other parameters, such as the nick flexibility limits, and the number of tethers in localized species.

Finally, our definition of a “nick” may need to be refined: if more than two regions intersect, e.g., in a four-way junction, it might prove needlessly restrictive. Furthermore, nicks are in fact extremely flexible: our restrictions on nick angles are thus not necessarily physical but rather a useful modeling tool, enabling the user to focus on core interactions and only dig into more esoteric behaviors when appropriate. Additionally, many of these reactions could also be ruled out via a toehold adjacency criterion [36], though we permit them here so as to not rule out other similar “remote toehold” interactions [19]. This enables a more nuanced analysis of structures that may form via localized strand displacement reactions.

References

- 1 Stefan Badelt, Casey Grun, Karthik V. Sarma, Brian Wolfe, Seung Woo Shin, and Erik Winfree. A domain-level DNA strand displacement reaction enumerator allowing arbitrary non-pseudoknotted secondary structures. *Journal of the Royal Society Interface*, 17(167):20190866, 2020. doi:10.1098/rsif.2019.0866.
- 2 Stefan Badelt, Seung Woo Shin, Robert F. Johnson, Qing Dong, Chris Thachuk, and Erik Winfree. A general-purpose CRN-to-DSD compiler with formal verification, optimization, and simulation capabilities. In R. Brijder and L. Qian, editors, *Proceedings of the 23rd International Conference on DNA Computing and Molecular Programming*, volume 10467 of *Lecture Notes in Computer Science*, pages 232–248, 2017. doi:10.1007/978-3-319-66799-7_15.

- 3 Joseph Berleant, Christopher Berlind, Stefan Badelt, Frits Dannenberg, Joseph Schaeffer, and Erik Winfree. Automated sequence-level analysis of kinetics and thermodynamics for domain-level DNA strand-displacement systems. *Journal of the Royal Society Interface*, 15:20180107, 2018. doi:10.1098/rsif.2018.0107.
- 4 Joakim Bohlin, Michael Matthies, Erik Poppleton, Jonah Procyk, Aatmik Mallya, Hao Yan, and Petr Šulc. Design and simulation of DNA, RNA and hybrid protein–nucleic acid nanostructures with oxView. *Nature Protocols*, 17:1762–1788, 2022. doi:10.1038/s41596-022-00688-5.
- 5 Hieu Bui, Sudhanshu Garg, Vincent Miao, Tianqi Song, Reem Mokhtar, and John Reif. Design and analysis of linear cascade DNA hybridization chain reactions using DNA hairpins. *New Journal of Physics*, 19:015006, 2017. doi:10.1088/1367-2630/aa53d0.
- 6 Hieu Bui, Vincent Miao, Sudhanshu Garg, Reem Mokhtar, Tianqi Song, and John Reif. Design and analysis of localized DNA hybridization chain reactions. *Small*, 13(12):1602983, 2017. doi:10.1002/sml.201602983.
- 7 Hieu Bui, Shalin Shah, Reem Mokhtar, Tianqi Song, Sudhanshu Garg, and John Reif. Localized DNA hybridization chain reactions on DNA origami. *ACS Nano*, 12(2):1146–1155, 2018. doi:10.1021/acsnano.7b06699.
- 8 Gourab Chatterjee, Neil Dalchau, Richard A. Muscat, Andrew Phillips, and Georg Seelig. A spatially localized architecture for fast and modular DNA computing. *Nature Nanotechnology*, 12:920–927, 2017. doi:10.1038/nnano.2017.127.
- 9 Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik, and Georg Seelig. Programmable chemical controllers made from DNA. *Nature Nanotechnology*, 8:755–762, 2013. doi:10.1038/nnano.2013.189.
- 10 Kevin M. Cherry and Lulu Qian. Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. *Nature*, 559:370–376, 2018. doi:10.1038/s41586-018-0289-6.
- 11 Kevin M. Cherry and Lulu Qian. Supervised learning in DNA neural networks. *Nature*, 645:639–647, 2025. doi:10.1038/s41586-025-09479-w.
- 12 Neil Dalchau, Harish Chandran, Nikhil Gopalkrishnan, Andrew Phillips, and John Reif. Probabilistic analysis of localized DNA hybridization circuits. *ACS Synthetic Biology*, 4(8):898–913, 2015. doi:10.1021/acssynbio.5b00044.
- 13 Frits Dannenberg, Marta Kwiatkowska, Chris Thachuk, and Andrew J. Turberfield. DNA walker circuits: computational potential, design, and verification. *Natural Computing*, 14:195–211, 2015. doi:10.1007/s11047-014-9426-9.
- 14 Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Proceedings of TACAS 2008*, volume 4963 of *LNCS*, pages 337–340. Springer, 2008. doi:10.1007/978-3-540-78800-3_24.
- 15 David Doty, Benjamin L. Lee, and Tristan Stérin. scadnano: A browser-based, scriptable tool for designing DNA nanostructures. In Cody Geary and Matthew J. Patitz, editors, *26th International Conference on DNA Computing and Molecular Programming (DNA 26)*, volume 174 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:17, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.2020.9.
- 16 Shawn M. Douglas, Adam H. Marblestone, Surat Teerapittayanon, Alejandro Vazquez, George M. Church, and William M. Shih. Rapid prototyping of 3D DNA-origami shapes with caDNAno. *Nucleic Acids Research*, 37:5001–5006, 2009. doi:10.1093/nar/gkp436.
- 17 Jonathan P. K. Doye, Thomas E. Ouldridge, Ard A. Louis, Flavio Romano, Petr Šulc, Christian Matek, Benedict E. K. Snodin, Lorenzo Rovigatti, John S. Schreck, Ryan M. Harrison, and William P. J. Smith. Coarse-graining DNA for simulations of DNA nanotechnology. *Physical Chemistry Chemical Physics*, 15:20395–20414, 2013. doi:10.1039/C3CP53545B.
- 18 Sisi Fan, Jin Cheng, Yan Liu, Dongfang Wang, Tao Luo, Bin Dai, Chuan Zhang, Daxiang Cui, Yonggang Ke, and Jie Song. Proximity-induced pattern operations in reconfigurable DNA origami domino array. *Journal of the American Chemical Society*, 142(34):14566–14573, 2020. doi:10.1021/jacs.0c06061.

- 19 Anthony J. Genot, David Yu Zhang, Jonathan Bath, and Andrew J. Turberfield. Remote toehold: A mechanism for flexible control of DNA hybridization kinetics. *Journal of the American Chemical Society*, 133(7):2177–2182, 2011. doi:10.1021/ja1073239.
- 20 Casey Grun, Justin Werfel, David Yu Zhang, and Peng Yin. DyNAMiC Workbench: an integrated development environment for dynamic DNA nanotechnology. *Journal of the Royal Society Interface*, 12:20150580, 2015. doi:10.1098/rsif.2015.0580.
- 21 Dejan Jovanović and Leonardo de Moura. Solving non-linear arithmetic. In *Proceedings of IJCAR 2012*, volume 7364 of *LNCs*, pages 339–354. Springer, 2012. doi:10.1007/978-3-642-31365-3_27.
- 22 Sarika Kumar and Matthew R. Lakin. A geometric framework for reaction enumeration in computational nucleic acid devices. *Journal of the Royal Society Interface*, 20(208):20230259, 2023. doi:10.1098/rsif.2023.0259.
- 23 Sarika Kumar, Julian M. Weisburd, and Matthew R. Lakin. Structure sampling for computational estimation of localized DNA interaction rates. *Scientific Reports*, 11:12730, 2021. doi:10.1038/s41598-021-92145-8.
- 24 Matthew R. Lakin. LocalizedEnumeratorOptimization. Software, swbId: swb:1:dir:095f551c8228cf74f1548237675d8391b3963452 (visited on 2026-07-15). URL: <https://github.com/matthewlakin/LocalizedEnumeratorOptimization>.
- 25 Matthew R. Lakin and Sarika Kumar. Geometric enumeration of localized DNA strand displacement reaction networks. In Shinnosuke Seki and Jaimie Marie Stewart, editors, *30th International Conference on DNA Computing and Molecular Programming (DNA 30)*, volume 314 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:24, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.30.1.
- 26 Matthew R. Lakin, Loïc Paulevé, and Andrew Phillips. Stochastic simulation of multiple process calculi for biology. *Theoretical Computer Science*, 431:181–206, 2012. doi:10.1016/j.tcs.2011.12.057.
- 27 Matthew R. Lakin, Rasmus Petersen, Kathryn E. Gray, and Andrew Phillips. Abstract modelling of tethered DNA circuits. In Satoshi Murata and Satoshi Kobayashi, editors, *Proceedings of the 20th International Conference on DNA Computing and Molecular Programming*, volume 8727 of *Lecture Notes in Computer Science*, pages 132–147. Springer International Publishing, 2014. doi:10.1007/978-3-319-11295-4_9.
- 28 Matthew R. Lakin and Andrew Phillips. Modelling, simulating and verifying Turing-powerful strand displacement systems. In L. Cardelli and W. Shih, editors, *Proceedings of the 17th International Conference on DNA Computing and Molecular Programming*, volume 6937 of *Lecture Notes in Computer Science*, pages 130–144. Springer-Verlag, 2011. doi:10.1007/978-3-642-23638-9_12.
- 29 Matthew R. Lakin and Andrew Phillips. Automated, constraint-based analysis of tethered DNA nanostructures. In R. Brijder and L. Qian, editors, *Proceedings of the 23rd International Conference on DNA Computing and Molecular Programming*, volume 10467 of *Lecture Notes in Computer Science*, pages 1–16. Springer, Cham, 2017. doi:10.1007/978-3-319-66799-7_1.
- 30 Matthew R. Lakin and Andrew Phillips. Automated analysis of tethered DNA nanostructures using constraint solving. *Natural Computing*, 17(4):709–722, 2018. doi:10.1007/s11047-018-9693-y.
- 31 Matthew R. Lakin, Simon Youssef, Luca Cardelli, and Andrew Phillips. Abstractions for DNA circuit design. *Journal of the Royal Society Interface*, 9(68):460–486, 2012. doi:10.1098/rsif.2011.0343.
- 32 Matthew R. Lakin, Simon Youssef, Filippo Polo, Stephen Emmott, and Andrew Phillips. Visual DSD: a design and analysis tool for DNA strand displacement systems. *Bioinformatics*, 27(22):3211–3213, 2011. doi:10.1093/bioinformatics/btr543.
- 33 Richard A. Muscat, Karin Strauss, Luis Ceze, and Georg Seelig. DNA-based molecular architecture with spatially localized components. In *ISCA '13: Proceedings of the 40th Annual International Symposium on Computer Architecture*, pages 177–188, 2013. doi:10.1145/2485922.2485938.

- 34 Thomas E. Ouldrige, Rollo L. Hoare, Ard A. Louis, Jonathan P. K. Doye, Jonathan Bath, and Andrew J. Turberfield. Optimizing DNA nanotechnology through coarse-grained modeling: A two-footed DNA walker. *ACS Nano*, 7:2479–2490, 2013. doi:10.1021/nm3058483.
- 35 Jennifer E. Padilla, Ruojie Sha, Martin Kristiansen, Junghuei Chen, Natasha Jonoska, and Nadrian C. Seeman. A signal-passing DNA-strand-exchange mechanism for active self-assembly of DNA nanostructures. *Angewandte Chemie International Edition*, 54(20):5939–5942, 2015. doi:10.1002/anie.201500252.
- 36 Rasmus L. Petersen, Matthew R. Lakin, and Andrew Phillips. A strand graph semantics for DNA-based computation. *Theoretical Computer Science*, 632:43–73, 2016. doi:10.1016/j.tcs.2015.07.041.
- 37 Erik Poppleton, Joakim Bohlin, Michael Matthies, Shuchi Sharma, Fei Zhang, and Petr Šulc. Design, optimization and analysis of large DNA and RNA nanostructures through interactive visualization, editing and molecular simulation. *Nucleic Acids Research*, 48(12):e72, 2020. doi:10.1093/nar/gkaa417.
- 38 Lulu Qian and Erik Winfree. Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332:1196–1201, 2011. doi:10.1126/science.1200520.
- 39 Lulu Qian, Erik Winfree, and Jehoshua Bruck. Neural network computation with DNA strand displacement cascades. *Nature*, 475:368–372, 2011. doi:10.1038/nature10262.
- 40 Paul W. K. Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440:297–302, 2006. doi:10.1038/nature04586.
- 41 Carlo Spaccasassi, Matthew R. Lakin, and Andrew Phillips. A logic programming language for computational nucleic acid devices. *ACS Synthetic Biology*, 8(7):1530–1547, 2019. doi:10.1021/acssynbio.8b00229.
- 42 Niranjana Srinivas, James Parkin, Georg Seelig, Erik Winfree, and David Soloveichik. Enzyme-free nucleic acid dynamical systems. *Science*, 358(6369):eaal2052, 2017. doi:10.1126/science.aal2052.
- 43 Antonio Suma, Erik Poppleton, Michael Matthies, Petr Šulc, Flavio Romano, Ard A. Louis, Jonathan P. K. Doye, Cristian Micheletti, and Lorenzo Rovigatti. TacoxDNA: A user-friendly web server for simulations of complex DNA structures, from single strands to origami. *Journal of Computational Chemistry*, 40(29):2586–2595, 2019. doi:10.1002/jcc.26029.
- 44 Michael W. Trosset. Distance matrix completion by numerical optimization. *Computational Optimization and Applications*, 17:11–22, 2000. doi:10.1023/A:1008722907820.
- 45 Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17:261–272, 2020. doi:10.1038/s41592-019-0686-2.
- 46 David J. Wales and Jonathan P. K. Doye. Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *Journal of Physical Chemistry A*, 101(28):5111–5116, 1997. doi:10.1021/jp970984n.
- 47 Brian R. Wolfe and Niles A. Pierce. Sequence design for a test tube of interacting nucleic acid strands. *ACS Synthetic Biology*, 4(10):1086–1100, 2015. doi:10.1021/sb5002196.
- 48 Brian R. Wolfe, Nicholas J. Porubsky, Joseph N. Zadeh, Robert M. Dirks, and Niles A. Pierce. Constrained multistate sequence design for nucleic acid reaction pathway engineering. *Journal of the American Chemical Society*, 139(8):3134–3144, 2017. doi:10.1021/jacs.6b12693.
- 49 Fei Zhang, Jeanette Nangreave, Yan Liu, and Hao Yan. Reconfigurable DNA origami to generate quasifractal patterns. *Nano Letters*, 12:3290–3295, 2012. doi:10.1021/nl301399z.

A Tethered reaction enumeration definitions

$$\begin{array}{c}
\frac{
\begin{array}{l}
G = (V, \text{length}, \text{color}, A, \text{toehold}, E) \quad a \in A \setminus E \quad a \cap \text{sites}(E) = \emptyset \\
G' = (V, \text{length}, \text{color}, A, \text{toehold}, (E \cup \{a\})) \quad \text{plausible}(G')
\end{array}
}{G \rightarrow G'} \text{ (BIND)} \\
\\
\frac{
\begin{array}{l}
G = (V, \text{length}, \text{color}, A, \text{toehold}, E) \\
a = \{s_1, s_2\} \in E \quad \text{toehold}(a) \quad \neg \text{sameSpecies}(s_1, s_2, G') \\
G' = (V, \text{length}, \text{color}, A, \text{toehold}, (E \setminus \{a\})) \quad \text{plausible}(G')
\end{array}
}{G \rightarrow G'} \text{ (UNBIND)} \\
\\
\frac{
\begin{array}{l}
G = (V, \text{length}, \text{color}, A, \text{toehold}, E) \quad e = \{s, s'\} \in E \\
a = \{s, s''\} \in A \setminus E \quad s'' \notin \text{sites}(E) \quad \text{sameSpecies}(s, s'', G) \\
G' = (V, \text{length}, \text{color}, A, \text{toehold}, ((E \setminus \{e\}) \cup \{a\})) \quad \text{plausible}(G')
\end{array}
}{G \rightarrow G'} \text{ (MIGRATE3)} \\
\\
\frac{
\begin{array}{l}
G = (V, \text{length}, \text{color}, A, \text{toehold}, E) \quad e_1 = \{\text{3pr}(s_4), s_1\} \in E \\
e_2 = \{\text{3pr}(s_1), s_2\} \in E \quad e_3 = \{\text{3pr}(s_2), s_3\} \in E \quad e_4 = \{\text{3pr}(s_3), s_4\} \in E \\
a_1 = \{\text{3pr}(s_1), s_4\} \in A \quad a_2 = \{\text{3pr}(s_3), s_2\} \in A \\
G' = (V, \text{length}, \text{color}, A, \text{toehold}, ((E \setminus \{e_2, e_4\}) \cup \{a_1, a_2\})) \quad \text{plausible}(G')
\end{array}
}{G \rightarrow G'} \text{ (MIGRATE4)}
\end{array}$$

■ **Figure A.1** Inference rules defining a geometrically-aware transition relation ($\rightarrow$) between strand graphs; reproduced from previous work [22] for convenience. The *plausible* predicate defines the interface to a geometric constraint solver that which determines if the reaction is permitted by the geometry of the candidate product structure.

B Tethered region graph definitions

Here we reproduce the definition of region graphs from our previous work [22]. We do not modify this to incorporate tether information as it is not necessary to do so; tether information is added back when the constraints to be solved for a given tethered strand graph are defined.

B.1 Defining regions

In previous work, we defined a notion of “condensing” a structure representation to collapse adjacent single-stranded or double-stranded domains in a process representation of a structure, so that the domain boundaries became the only important positions represented in the constraint problem. Here we use a similar idea by assigning domains to “regions.” Each region is assigned a label, which we represent using natural numbers.

$$\begin{array}{c}
\frac{S_i \in \bar{S} \quad S_i \rightarrow G' \quad \bar{S}' = \bar{S} \setminus \{S_i\} \cup \text{tethered}(G')}{[[\bar{S}]] \Rightarrow [[\bar{S}']] + \text{free}(G')} \text{ (TILEONE)} \\
\\
\frac{S_i, S_j \in \bar{S} \quad i \neq j \quad (S_i \parallel S_j) \rightarrow G' \quad \bar{S}' = \bar{S} \setminus \{S_i, S_j\} \cup \text{tethered}(G')}{[[\bar{S}]] \Rightarrow [[\bar{S}']] + \text{free}(G')} \text{ (TILETWO)} \\
\\
\frac{S_i \in \bar{S} \quad (S_i \parallel S) \rightarrow G' \quad \bar{S}' = \bar{S} \setminus \{S_i\} \cup \text{tethered}(G')}{[[\bar{S}]] + S \Rightarrow [[\bar{S}']] + \text{free}(G')} \text{ (TILEFREE)} \\
\\
\frac{(S \parallel S') \rightarrow G' \quad \text{tethered}(G') = \emptyset}{S + S' \Rightarrow \text{free}(G')} \text{ (FREEFREE)}
\end{array}$$

■ **Figure A.2** Inference rules defining a geometrically-aware transition relation ($\Rightarrow$) between collections of strand graph species, including tile species with tethered components; these are reproduced from previous work [25] for convenience. These rules build on our previously reported $\rightarrow$ rules for geometrically aware reaction enumeration on strand graphs of free species [22], shown here in Figure A.1. The rules presented here implement a CRN semantics [36] for systems that may contain both free and tethered species, where we write $\bar{S}$ for finite collections of fully connected strand graphs, each of which represents a single molecular species. The syntax $[[\bar{S}]]$ represents a tile species containing species $\bar{S}$, each of which must be tethered; we assume that other species are freely diffusing. We write $S \parallel S'$ for the strand graph resulting from the parallel composition of S and S' . We write $\text{tethered}(G)$ and $\text{free}(G)$ for the connected components of a strand graph G that represent tethered and free species, that is, those species which do or do not include a tether specification, respectively. Rule (TILEONE) identifies a single species tethered to the same tile that may react. Similarly, rule (TILETWO) encodes a reaction between two species tethered to the same tile. Rule (TILEFREE) identifies a free species that may react with a species tethered to a tile, and rule (FREEFREE) lifts the standard rules for reactions involving two free species into the $\Rightarrow$ relation. All reactions involving at least one tile species may produce new tethered species, and possibly some free species also. In the case of rule (FREEFREE), we note that the $\text{tethered}(G') = \emptyset$ premise is vacuously true since a tethered species cannot spontaneously emerge from a reaction between two free species. These rules form the basis of our localized geometric reaction enumerator.

► **Definition B.1** (Region mappings). *Assuming the definition of strand graphs in the main text, a region mapping for a given strand graph is a function $\text{region} : \text{legal}_S \rightarrow \mathbb{N}$ that assigns a region label to every legal site in the strand graph G , and which has the following properties:*

- if $\text{region}((v, n)) = r$ and $(v, n) \notin \text{sites}(E)$, then:
 - if $(v, n - 1) \in \text{legal}_S(V, \text{length})$ then $\text{region}((v, n - 1)) = r$ iff $(v, n - 1) \notin \text{sites}(E)$;
 - if $(v, n + 1) \in \text{legal}_S(V, \text{length})$ then $\text{region}((v, n + 1)) = r$ iff $(v, n + 1) \notin \text{sites}(E)$.
- if $\text{region}((v, n)) = r$ and $\{(v, n), (v', n')\} \in E$, then:
 - if $(v, n - 1) \in \text{legal}_S(V, \text{length})$ then $\text{region}((v, n - 1)) = r$ iff $\{(v, n - 1), (v', n' + 1)\} \in E$;
 - if $(v, n + 1) \in \text{legal}_S(V, \text{length})$ then $\text{region}((v, n + 1)) = r$ iff $\{(v, n + 1), (v', n' - 1)\} \in E$.
- if we define $\text{allregionlabels}(G) = \{r \mid \exists s \in \text{legal}_S. \text{region}(s) = r\}$, that is, the set of region labels assigned for strand graph G , and there are n such distinct region labels, then $\text{allregionlabels}(G) = \{0, \dots, n - 1\}$.
- if $s < s'$ in the ordering of sites on the canonically relabeled strand graph and $\text{region}(s) \neq \text{region}(s')$ then $\text{region}(s) < \text{region}(s')$.

In other words, a valid region mapping assigns the same region label to all sites representing domains in a continuous run of single-stranded domains or double-stranded domains that form a continuous double helix (i.e., with no nicks or overhangs). Furthermore, we note that the sites in a given region are either all bound or all unbound. This allows us to define a predicate *isBoundRegion* on region labels such that *isBoundRegion*(r) returns true if all sites in region r are bound and false otherwise (i.e., if all sites are unbound). This will be helpful in defining the constraints associated with a region, below. The last two conditions do ensure that the region mapping for a given strand graph is unique; the final condition essentially says that the ordering on sites induces an ordering on regions.

We also define *regionsites*(r), the inverse of the *region* function, which returns the set of sites contained in a particular region, as follows:

$$\text{regionsites}(r) = \{s \mid \text{region}(s) = r\}.$$

We define a *position* p as a pair (s, w) where $s \in \text{legal}_G$ is a legal site from the original strand graph G and $w \in \{5', 3'\}$ is a tag which determines whether we are referring to the 5' or 3' end of the domain represented by that site. This will be useful below, to define which parts of the structure we are referring to. We will write *5prPosns*(r) for the set of positions located at the end of region r on the 5' end of their site, and likewise *3prPosns*(r) for the set of positions located at the end of that region but on the 3' end of their site, which can be defined as follows:

$$\begin{aligned} 5prPosns(r) &= \{((v, n), 5') \mid (v, n) \in \text{regionsites}(r) \wedge (v, n-1) \notin \text{regionsites}(r)\} \\ 3prPosns(r) &= \{((v, n), 3') \mid (v, n) \in \text{regionsites}(r) \wedge (v, n+1) \notin \text{regionsites}(r)\}. \end{aligned}$$

Furthermore, it will also be necessary to uniquely identify the two “ends” of a region r , and compute which positions are located there (there will be one position at each end for a single-stranded region but two at each end for a double-stranded region). For clarity we will refer to these as the A and B ends of the region. These will be defined by exploiting the underlying ordering on sites that arises from the canonical representations defined for strand graphs; in the case of a double-stranded region, this allows us to pick one strand whose directionality then uniquely induces the choice of A and B ends. We will write *APosns*(r) and *BPosns*(r) for the sets of positions located at the A and B ends of region r , which can be defined as follows:

- $APosns(r) = \{((v, n), 5')\} \cup \{((v', n'), 3') \mid \{(v, n), (v', n')\} \in E\}$, where (v, n) is the **least** site in the set *5prPosns*(r).
- $BPosns(r) = \{((v, n), 3')\} \cup \{((v', n'), 5') \mid \{(v, n), (v', n')\} \in E\}$, where (v, n) is the **least** site in the set *3prPosns*(r).

Finally, we define the set of all positions that occur at either end of some region in G :

$$\text{allPosns}(G) = \{p \mid \exists r \in \text{allregionlabels}(G). p \in (5prPosns(r) \cup 3prPosns(r))\}.$$

B.2 Defining region graphs

► **Definition B.2** (Region graphs). *Given a (canonically labeled) strand graph G , we are now in a position to define the corresponding region graph, $\text{regiongraph}(G)$. The region graph will be an undirected graph (RGV, RGE) whose set of vertices $RGV = \{P_1, \dots, P_k\}$, where $\{P_1, \dots, P_k\}$ is a partition of $\text{allPosns}(G)$ such that p and p' occur in the same subset P_i iff any of the following statements is true of p and p' :*

- $p \in (APosns(r) \cap 5prPosns(r))$ and $p' \in (APosns(r) \cap 3prPosns(r))$, for some r .
- $p \in (BPosns(r) \cap 5prPosns(r))$ and $p' \in (BPosns(r) \cap 3prPosns(r))$, for some r .

- $p = ((v, n), 5')$ and $p' = ((v, n - 1), 3')$.
- $p = ((v, n), 3')$ and $p' = ((v, n + 1), 5')$.

Note that the first two points imply that for all $r \in \text{allregionlabels}(G)$, $A\text{Posns}(r) \subseteq P_i$ for some P_i and $B\text{Posns}(r) \subseteq P_j$ for some P_j . Note that it is possible for i and j to be equal in the previous statement: this would correspond to a loop comprising a single region. Note also that the latter two points only apply where both p and p' are elements of $\text{allPosns}(G)$. We can therefore define the set of edges RGE of the region graph, as follows:

$$RGE = \{\{P_i, P_j\} \mid \exists r \in \text{allregionlabels}(G). A\text{Posns}(r) \subseteq P_i \wedge B\text{Posns}(r) \subseteq P_j\}.$$

We also assume that each edge is labeled with the corresponding region label r from above; this allows properties of the region such as the length or type (single-stranded vs double-stranded) to be looked up or stored.

Note that, since the region mapping is unique for a given strand graph, the partition used to label the vertices of the region graph will also be unique. Thus, it follows that the region graph $\text{regiongraph}(G)$ for a given strand graph G will be unique.

C Additional data

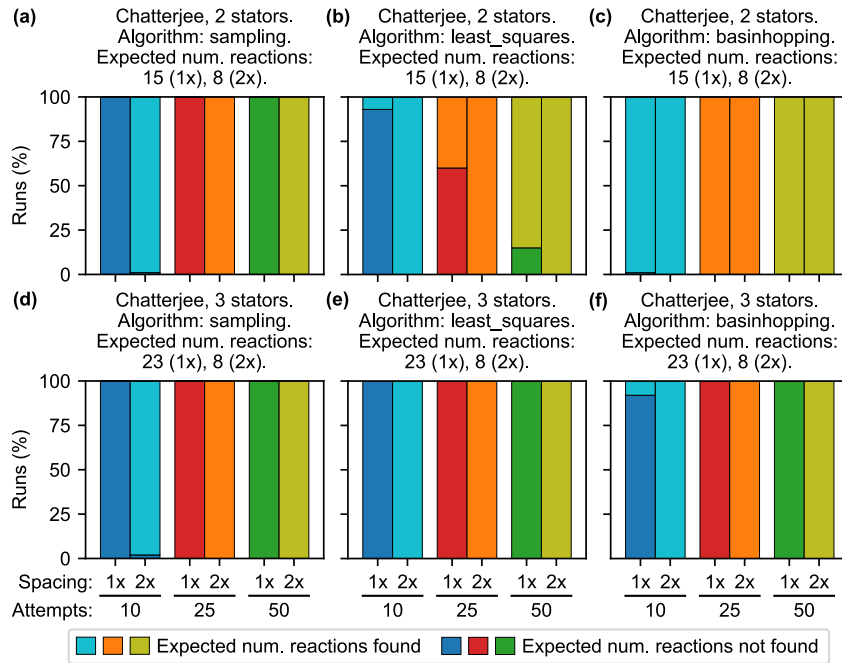


Figure C.3 Statistics on how often we observed the expected number of reactions in 100 attempts to enumerate reaction networks for the Chatterjee et al. tethered transmission line system. The expected numbers of reactions are shown in each subfigure title and were determined via manual analysis of the systems. The analysis for the two-stator case is shown in Figure 3. The single-spaced three-stator case is similar, except that more reactions are required to open the second stator and second fuel hairpin. The expected number of reactions for the double-spaced case is the same for the two- and three-stator systems, as in both cases only the reversible reactions that precede the interaction with the second stator are permitted by the tether geometry.

Reverse-Robust Computation with Chemical Reaction Networks

Ravi Kini ✉ 

Computer Science, University of California, Davis, CA, USA

David Doty ✉ 

Computer Science, University of California, Davis, CA, USA

Abstract

Chemical reaction networks, or CRNs, are known to stably compute semilinear Boolean-valued predicates and functions, provided that all reactions are irreversible. However, this property does not hold for wet-lab implementations, as all chemical reactions are reversible, even at very slow rates. We study the computational power of CRNs under the *reverse-robust* computation model, where reactions are permitted to occur either in forward or in reverse up to a cutoff point, after which they may only occur in forward. Our main results show that all semilinear predicates and all semilinear functions can be computed reverse-robustly, and in fact, that existing constructions continue to hold under the reverse-robust computational model. A key tool used to prove correctness under the reverse-robust computation model is *invariants*: linear (or linear modulo some m) combinations of the counts of the species that are preserved by all reactions.

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases chemical reaction networks, reverse-robust computation, semilinear

Digital Object Identifier 10.4230/LIPIcs.DNA.32.6

Funding *David Doty*: funded by NSF awards 2211793 and 2329909.

1 Introduction

Chemical reaction networks (CRNs) are a fundamental tool for understanding and designing molecular systems. By abstracting chemical reactions into a set of finite, rule-based transformations, CRNs allow us to model the behavior of complex chemical systems. For example, a CRN with the single reaction $X_1 + X_2 \rightarrow Y$ produces one molecule of Y whenever an X_1 reacts with an X_2 . Interpreting the counts of X_1, X_2 as the vector-valued input and the count of Y as the output, the CRN effectively calculates the function $f(x_1, x_2) = \min(x_1, x_2)$. Suppose the single reaction is instead $X_1 + X_2 \rightarrow 2X_2$, where X_2 acts as a catalyst, producing another molecule of X_2 when it reacts with a X_1 . Interpreting the presence of X_1 as “no” and the presence of X_2 as “yes”, the CRN effectively calculates the predicate $\phi(x_1, x_2) = \text{yes} \Leftrightarrow x_2 > 0$. It is known that precisely the *semilinear* predicates $\phi : \mathbb{N}^k \rightarrow \{0, 1\}$ [1] and functions $f : \mathbb{N}^k \rightarrow \mathbb{N}$ [2] can be computed *stably*, which roughly means that we can reach a correct state regardless of the reactions that occur or the order in which they occur. More precisely, we require that for every state $\mathbf{c}$ reachable from the initial state encoding the input, there is a correct state $\mathbf{y}$ reachable from $\mathbf{c}$, where $\mathbf{y}$ is also *stable*, meaning that every state reachable from $\mathbf{y}$ has the same output as $\mathbf{y}$. Under the reasonable assumption that only a finite number of states are reachable from the initial state, this simple combinatorial definition based on reachability is equivalent to requiring that the CRN *will* reach a correct, stable state with probability 1.

Existing constructions [1, 2, 3] assume that reactions may not occur in reverse. In reality, it is known that any chemical reaction that can happen in the forward direction, turning reactants into products, can also happen in reverse, turning the products back into the



© Ravi Kini and David Doty;

licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 6; pp. 6:1–6:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

reactants, even if at a much slower rate.¹ To address this incompatibility, we introduce the notion of *reverse-robust* computation with CRNs, in which reactions are allowed to occur in reverse, but we must always be able to reach a correct state using forward reactions. We formalize this by generalizing the previous definition of stable computation to require that from any state $\mathbf{c}$ reachable from the initial state by any combination of forward and reverse reactions, there is a correct, stable state reachable using only forward reactions. In other words, although the reverse reactions can take the CRN to states potentially unreachable through forward reactions alone, such reactions cannot prevent the CRN from reaching the correct answer. We note that this definition allows reverse reactions only transiently; if they eventually stop, then forward reactions will always take the CRN to a correct, stable state.²

To understand the distinction between this model and the stable computation model, consider the CRN $2X \rightarrow Y$, which stably computes the function $f(x) = \lfloor x/2 \rfloor$. We then add two reactions involving a third species, Z :



Once again interpreting X as the input and Y as the output, under the stable computation model, since there are no copies of Z present initially and no reaction produces Z , there is no way for reactions (2) and (3) to occur. Consequently, the CRN behaves identically to having only reaction (1), and computes $f(x) = \lfloor x/2 \rfloor$. Consider, however, what happens when reactions are allowed to occur in reverse, under the reverse-robust computation model: when reaction (1) produces a Y , reaction (2) in reverse may convert the Y to a Z , and reaction (3) (forward) may then annihilate that Z . It is then possible to reach a state where no molecules remain, and from which no correct state can be reached by only forward reactions.³ Reactions (2) and (3) behave like a “trap”, only disrupting the computation when reverse reactions are allowed.

We show that it is precisely the semilinear predicates and functions that can be computed reverse-robustly; that is, CRNs are equally as powerful under the reverse-robust computation model as under the stable computation model. Note that the capabilities of reverse-robustly computing CRNs are clearly limited by the established results about the capabilities of stably computing CRNs. We may think of stable computation as requiring that the CRN work against an adversary that may choose which reaction occurs at each intermediate state, only allowing reactions to occur in the forward direction. We can then think of reverse-robust computation as requiring that the CRN work against an adversary that may choose which reaction occurs at each intermediate state, allowing reactions to occur in both the forward and reverse directions. The adversary of the reverse-robust computation model is evidently more powerful than that of the stable computation model; the main challenge is then to

¹ The fully general model of CRNs associates to a reaction such as $A \xrightleftharpoons[r]{f} B$ a forward *rate constant* f and reverse rate constant r . Without getting into details of the precise energy model, if $f > r$, this means that B is more energetically stable than A , but to have $r = 0$ (i.e., the reaction is completely irreversible) requires that A be have energy *infinitely* larger than B .

² Note that if the definition permitted reverse reactions to occur indefinitely, this would trivially prevent any state from being stable, since one could reverse out of any state possibly all the way back to the initial state. See Section 5 for more discussion.

³ Note that any state can be reached by a combination of forward and reverse reactions, reversing reaction (3) to produce arbitrary amounts of Z , and the other two reactions to produce target amounts of X and Y from that.

show that against this more powerful adversary, we can compute anything that we could compute against the weaker adversary. In fact, we show that the existing constructions for CRNs that worked against the weaker adversary to compute semilinear predicates and functions continue to work against the stronger adversary. To prove the correctness of those constructions under the reverse-robust computation model, we use “invariants”, quantities that are preserved by the reactions of a CRN (and thus, by the reverse of those reactions as well).

The paper is organized as follows. Section 2 formally defines what it means for a CRN to compute reverse-robustly (Definitions 1 and 2) and introduces the concept of an invariant. Sections 3 and 4 contain the main positive results of the paper, and prove that the existing constructions used to decide semilinear sets and compute semilinear functions stably also work under the reverse-robust computation model with minimal or no modification. Appendix A contains the proofs of two technical lemmas used in proving the main results.

2 Preliminaries

2.1 Notation

Let $\mathbb{N}$ denote the nonnegative integers. For any finite set Λ , $\mathbb{N}^\Lambda$ denotes the set of functions $f : \Lambda \rightarrow \mathbb{N}$. Equivalently, $\mathbf{c} \in \mathbb{N}^\Lambda$ can be interpreted as a vector of $|\Lambda|$ nonnegative integers, where each element specifies the nonnegative integer count of an element of Λ . $\mathbf{c}(i)$ denotes the i -th coordinate of $\mathbf{c}$, and if $\mathbf{c}$ is indexed by elements of Λ , then $\mathbf{c}(Y)$ denotes the count of species $Y \in \Lambda$. We sometimes use multiset notation for such vectors, e.g., $\{A, 3C\}$ for the vector $\mathbf{c} = (1, 0, 3)$, assuming there are three species A, B, C , indexed in that order. For $\Sigma \subseteq \Lambda$, $\mathbf{i}|_\Sigma$ denotes the restriction of $\mathbf{i}$ to Σ .

For two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^k$, we write $\mathbf{x} \geq \mathbf{y}$ to indicate that $\mathbf{x}(i) \geq \mathbf{y}(i)$ for all $1 \leq i \leq n$, $\mathbf{x} \geq \mathbf{y}$ to indicate that $\mathbf{x} \geq \mathbf{y}$ but $\mathbf{x} \neq \mathbf{y}$, and $\mathbf{x} > \mathbf{y}$ to indicate that $\mathbf{x}(i) > \mathbf{y}(i)$ for all $1 \leq i \leq n$. When $\mathbf{y} = \mathbf{0}$, we refer to $\mathbf{x}$ as nonnegative, semipositive, and positive, respectively. $\leq, \leq, <$ are defined analogously.

2.2 Chemical reaction networks

A *chemical reaction network* (CRN) is a pair $\mathcal{C} = (\Lambda, R)$ where Λ is a finite set of *chemical species* and R is a finite set of (forward) reactions over Λ , where a (*forward*) *reaction* is a pair $r = (\mathbf{a}, \mathbf{p}) \in \mathbb{N}^\Lambda \times \mathbb{N}^\Lambda$ where $\mathbf{a}$ indicates the *reactants* and $\mathbf{p}$ the *products*. For a reaction $r = (\mathbf{a}, \mathbf{p})$, we use $r^\leftarrow = (\mathbf{p}, \mathbf{a})$ to denote the corresponding *reverse reaction*.

A *configuration* $\mathbf{c} \in \mathbb{N}^\Lambda$ assigns nonnegative integer counts to every species $\lambda \in \Lambda$. A reaction $r = (\mathbf{a}, \mathbf{p})$ is *applicable* to a configuration $\mathbf{x}$ if $\mathbf{x} \geq \mathbf{a}$; applying r to $\mathbf{x}$ then results in the configuration $\mathbf{y} = \mathbf{x} - \mathbf{a} + \mathbf{p}$. We say r is *reverse-applicable* to $\mathbf{x}$ if $\mathbf{x} \geq \mathbf{p}$. Equivalently, we may say that $r^\leftarrow$ is applicable to $\mathbf{x}$. We say r is *bi-applicable* to $\mathbf{x}$ if it is either applicable or reverse applicable to $\mathbf{x}$. If some reaction r is applicable or reverse-applicable to $\mathbf{x}$ and (reverse-)applying it results in the configuration $\mathbf{y}$, we write $\mathbf{x} \mapsto^1 \mathbf{y}$. If it is known that r was applicable, we write $\mathbf{x} \rightarrow^1 \mathbf{y}$. When the specific reaction is relevant, we may also write $\mathbf{x} \rightarrow^r \mathbf{y}$ if it is the forward reaction, and $\mathbf{x} \mapsto^{r^\leftarrow} \mathbf{y}$ if it is the reverse. A *bi-execution* $\mathcal{E}$ is a finite or infinite sequence of configurations $(\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \dots)$ such that for all $j \in \{1, \dots, |\mathcal{E}| - 1\}$, $\mathbf{c}_{j-1} \mapsto^1 \mathbf{c}_j$. A (*forward*-) *execution* is defined analogously, instead requiring that $\mathbf{c}_{j-1} \rightarrow \mathbf{c}_j$. A configuration $\mathbf{y}$ is *bireachable* from another configuration $\mathbf{x}$ (written as $\mathbf{x} \mapsto \mathbf{y}$) if there exists some finite bi-execution that starts at $\mathbf{x}$ and ends at $\mathbf{y}$, and *reachable* (written as $\mathbf{x} \rightarrow \mathbf{y}$) if there exists such a forward-execution. Note that both bireachability and reachability are

additive: if $\mathbf{x} \rightsquigarrow \mathbf{y}$, for all $\mathbf{z} \in \mathbb{N}^\Lambda$, $\mathbf{x} + \mathbf{z} \rightsquigarrow \mathbf{y} + \mathbf{z}$, and analogously for when $\mathbf{x} \rightarrow \mathbf{y}$. It also follows trivially that both bireachability and reachability are transitive and symmetric; however, bireachability is clearly reflexive, as we can simply reverse-apply the reactions moving backwards through the sequence of configurations, while the same is not true for reachability.

2.3 Stable and reverse-robust computation with CRNs

To perform computation using CRNs, we define a way to interpret the final configuration after letting forward reactions until the result can no longer change by applying forward reactions (characterized below as *stable computation*). We define *reverse-robust computation* analogously, letting both forward and reverse reactions occur until the result can no longer change by applying forward reactions. Computation using CRNs primarily involves the evaluation of *predicates* $\phi : \mathbb{N}^k \rightarrow \{0, 1\}$ and *functions* $f : \mathbb{N}^k \rightarrow \mathbb{N}$.

The definitions below reference *input species* $\Sigma \subseteq \Lambda$ and an *initial context* $\mathbf{s} \in \mathbb{N}^{\Lambda \setminus \Sigma}$. If $\mathbf{s} = \mathbf{0}$, we say that the CRN is *leaderless*. We may assume without loss of generality that if the initial context is nonzero, it contains a single copy of some *leader species* L . In both cases, we say that $\mathbf{i} \in \mathbb{N}^\Lambda$ is a *valid initial configuration* if $\mathbf{i} = \mathbf{s} + \mathbf{x}$ where $\mathbf{x}(\lambda_j) = 0$ for all $\lambda_j \in \Lambda \setminus \Sigma$; i.e., $\mathbf{i}$ contains only input species in addition to the initial context.

A *chemical reaction decider* (CRD) is a tuple $\mathcal{D} = (\Lambda, R, \Sigma, \Upsilon_1, \Upsilon_0, \mathbf{s})$ where (Λ, R) is a CRN, $\Sigma \subseteq \Lambda$ is the set of input species, Υ_1 is the set of *yes voters*, Υ_0 is the set of *no voters*, and $\mathbf{s} \in \mathbb{N}^{\Lambda \setminus \Sigma}$ is the initial context. In the context of CRDs, we use *output species* to refer to $\Gamma = \Upsilon_0 \cup \Upsilon_1$, i.e. the set of voters. We define a global output partial function $\Phi : \mathbb{N}^\Lambda \rightarrow \{0, 1\}$ as follows: $\Phi(\mathbf{c}) = 0$ if $\mathbf{c}(\lambda_j) > 0$ for some $\lambda_j \in \Upsilon_0$ and $\mathbf{c}(\lambda'_j) = 0$ for all $\lambda'_j \in \Upsilon_1$, $\Phi(\mathbf{c}) = 1$ if $\mathbf{c}(\lambda_j) > 0$ for some $\lambda_j \in \Upsilon_1$ and $\mathbf{c}(\lambda'_j) = 0$ for all $\lambda'_j \in \Upsilon_0$, and is undefined otherwise. In other words, a unanimous vote is required for an output. We say a configuration $\mathbf{c}$ is *stable* if for all $\mathbf{c}'$ such that $\mathbf{c} \rightarrow \mathbf{c}'$, $\Phi(\mathbf{c}) = \Phi(\mathbf{c}')$. A CRD $\mathcal{D}$ is said to *stably decide* the predicate $\phi : \mathbb{N}^\Sigma \rightarrow \{0, 1\}$ if for any valid initial configuration $\mathbf{i} \in \mathbb{N}^\Lambda$, letting $\mathbf{i}_0 = \mathbf{i}|_\Sigma$, for all $\mathbf{c} \in \mathbb{N}^\Lambda$ such that $\mathbf{i} \rightarrow \mathbf{c}$, there exists some $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$ where $\mathbf{o}$ is stable and $\Phi(\mathbf{o}) = \phi(\mathbf{i}_0)$.

► **Definition 1.** A CRD $\mathcal{D}$ is said to reverse-robustly decide the predicate $\phi : \mathbb{N}^\Sigma \rightarrow \{0, 1\}$ if for any valid initial configuration $\mathbf{i} \in \mathbb{N}^\Lambda$, letting $\mathbf{i}_0 = \mathbf{i}|_\Sigma$, for all $\mathbf{c} \in \mathbb{N}^\Lambda$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$, there exists some $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$ where $\mathbf{o}$ is stable and $\Phi(\mathbf{o}) = \phi(\mathbf{i}_0)$.

A *chemical reaction computer* (CRC) is a tuple $\mathcal{C} = (\Lambda, R, \Sigma, Y, \mathbf{s})$ where (Λ, R) is a CRN, $\Sigma \subseteq \Lambda$ is the set of input species, $Y \in \Lambda$ is the *output species*, and $\mathbf{s} \in \mathbb{N}^{\Lambda \setminus \Sigma}$ is the initial context. We say a configuration $\mathbf{c}$ is *stable* if for all $\mathbf{c}'$ such that $\mathbf{c} \rightarrow \mathbf{c}'$, $\mathbf{c}(Y) = \mathbf{c}'(Y)$. A CRC $\mathcal{C}$ is said to *stably compute* the function $f : \mathbb{N}^\Sigma \rightarrow \mathbb{N}$ if for any valid initial configuration $\mathbf{i} \in \mathbb{N}^\Lambda$, letting $\mathbf{i}_0 = \mathbf{i}|_\Sigma$, for all $\mathbf{c} \in \mathbb{N}^\Lambda$ such that $\mathbf{i} \rightarrow \mathbf{c}$, there exists some $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$ where $\mathbf{o}$ is stable and $\mathbf{o}(Y) = f(\mathbf{i}_0)$.

► **Definition 2.** A CRC $\mathcal{C}$ is said to reverse-robustly compute the function $f : \mathbb{N}^\Sigma \rightarrow \mathbb{N}$ if for any valid initial configuration $\mathbf{i} \in \mathbb{N}^\Lambda$, letting $\mathbf{i}_0 = \mathbf{i}|_\Sigma$, for all $\mathbf{c} \in \mathbb{N}^\Lambda$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$, there exists some $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$ where $\mathbf{o}$ is stable and $\mathbf{o}(Y) = f(\mathbf{i}_0)$.

Henceforth, we use CRN when a statement is applicable to either CRCs or CRDs. We can then say that a CRN computes stably if for all valid input configurations $\mathbf{i}$, for all configurations $\mathbf{c}$ such that $\mathbf{i} \rightarrow \mathbf{c}$, there exists some stable correct configuration $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$. Analogously, a CRN computes reverse-robustly if for all valid input configurations $\mathbf{i}$, for all configurations $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$, there exists some stable correct configuration $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$.

It is known that it is exactly the semilinear predicates that can be stably decided by a CRD, and exactly the semilinear functions that can be stably computed by a CRC.

► **Definition 3.** A set $L \subseteq \mathbb{N}^k$ is linear if there are vectors $\mathbf{b}, \mathbf{p}_1, \dots, \mathbf{p}_s$ such that $L = \{\mathbf{b} + n_1\mathbf{p}_1 + \dots + n_s\mathbf{p}_s \mid n_1, \dots, n_s \in \mathbb{N}\}$. A set is semilinear if it is a finite union of linear sets. A predicate $\phi : \mathbb{N}^d \rightarrow \{0, 1\}$ is semilinear if the set $\phi^{-1}(1)$ is semilinear. A function $f : \mathbb{N}^k \rightarrow \mathbb{N}$ is semilinear if its graph $\{(x, y) \in \mathbb{N}^{k+1} \mid f(x) = y\}$ is semilinear.

2.4 “Shuffling” executions

The ability to parallelize computation is a key technique used in designing CRNs that compute stably [2]. By this we mean the common trick of splitting input species X_i into two variants via a reaction $X_i \rightarrow X_{i,1} + X_{i,2}$, where $X_{i,j}$ is the i 'th input species for CRN C_j , and C_1 and C_2 then run in parallel on their respective copies of the input species. We show that this technique continues to work under the reverse-robust computation model.

We first state two technical lemmas about reachability that will help in showing this. Their proofs are contained in Appendix A. If $\mathbf{c}_0 \xrightarrow{r_1} \mathbf{c}_1 \xrightarrow{r_2} \mathbf{c}_2$, then we cannot in general commute the reactions since r_2 may not be applicable to $\mathbf{c}_0$. Furthermore, even if r_2 is applicable to $\mathbf{c}$ and $\mathbf{c}_0 \xrightarrow{r_2} \mathbf{c}'_1$, then r_1 may not be applicable to $\mathbf{c}'_1$. However, if none of the products of r_1 are reactants of r_2 , then in fact the reactions can be commuted.

► **Lemma 4.** Let $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2$ such that $\mathbf{c}_0 \xrightarrow{r_1} \mathbf{c}_1 \xrightarrow{r_2} \mathbf{c}_2$, where $r_1 = (\mathbf{a}_1, \mathbf{p}_1)$ and $r_2 = (\mathbf{a}_2, \mathbf{p}_2)$. If $\mathbf{p}_1 \cap \mathbf{a}_2 = \emptyset$ (none of the products of r_1 are reactants of r_2), then there is some $\mathbf{c}'_1$ such that $\mathbf{c}_0 \xrightarrow{r_2} \mathbf{c}'_1 \xrightarrow{r_1} \mathbf{c}_2$.

► **Lemma 5.** Let $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2$ such that $\mathbf{c}_0 \xrightarrow{r_1 \leftarrow} \mathbf{c}_1 \xrightarrow{r_2} \mathbf{c}_2$, where $r_1 = (\mathbf{a}_1, \mathbf{p}_1)$ and $r_2 = (\mathbf{a}_2, \mathbf{p}_2)$. If $\mathbf{a}_1 \cap \mathbf{a}_2 = \emptyset$ (none of the products of $r_1 \leftarrow$ are reactants of r_2), then there is some $\mathbf{c}'_1$ such that $\mathbf{c}_0 \xrightarrow{r_2} \mathbf{c}'_1 \xrightarrow{r_1 \leftarrow} \mathbf{c}_2$.

Further, when a forward reaction occurs consecutively with its reverse reaction, the pair may intuitively be canceled out, producing a bi-execution where those reactions do not occur. We now show that duplicating the input species allows for the parallel reverse-robust computation of multiple predicates or functions.

► **Lemma 6.** Let C_1, C_2 be some reverse-robustly computing CRNs with input species $\{X_1, \dots, X_k\}$. For clarity, rename X_i to $X_{i,1}$ in C_1 and $X_{i,2}$ in C_2 . Let C be the CRN formed by combining the reactions of C_1 and C_2 and adding the reaction $r_{i,0}$, for each $i \in \{1, \dots, k\}$:



Then for all valid input configurations $\mathbf{i}$, for all configurations $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$, there exists some stable configuration $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$ and the individual outputs of both C_1 and C_2 are correct.

Proof. Let $\mathbf{i}$ be an arbitrary initial configuration, where $\mathbf{i}(X_i) = x_i$. It is easy to see that if we start with only x_i each of $X_{i,1}$ and $X_{i,2}$ in configuration $\mathbf{i}^*$, the CRNs can be run independently, and are able to reverse-robustly compute their respective functions/predicates in parallel. Then for any $\mathbf{c}^*$ such that $\mathbf{i}^* \rightsquigarrow \mathbf{c}^*$, there exists some stable $\mathbf{o}^*$ such that $\mathbf{c}^* \rightarrow \mathbf{o}^*$ and the individual outputs are correct.

Now let $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$. For each i , note that by repeatedly applying $r_{i,0}$, we can reach some $\mathbf{c}'$ such that $\mathbf{c} \rightarrow \mathbf{c}'$ and $r_{i,0}$ is no longer applicable; we may then assume that in $\mathbf{c}$, $r_{i,0}$ is no longer applicable, and so X_i is not present.

We first argue that $\mathbf{i} \rightsquigarrow \mathbf{c}$ by a bi-execution where $r_{i,0}$ is never applied in reverse for all i . Let $\mathcal{E} = (\mathbf{c}_0, \mathbf{c}_1, \dots, \mathbf{c}_n)$ be a bi-execution where $r_{i,0}$ is reverse-applied (with $\mathbf{c}_0 = \mathbf{i}$ and $\mathbf{c}_n = \mathbf{c}$) for some i . Note that the only reaction where the products of $r_{i,0}^\leftarrow$ are reactants is $r_{i,0}$, and that those are the only reactions in which X_i is involved; this affords us a great deal of flexibility in shuffling the order of reactions.

Note that we can always select j and k where $j < k$ such that $\mathbf{c}_j \rightsquigarrow^{r_{i,0}^\leftarrow} \mathbf{c}_{j+1}$ and $\mathbf{c}_k \xrightarrow{r_{i,0}} \mathbf{c}_{k+1}$ and $\mathbf{c}_{j+1} \rightsquigarrow \mathbf{c}_k$ by a bi-execution where $r_{i,0}$ does not occur in forward or in reverse. This follows from the fact that X_i is not present in $\mathbf{c}$; if $r_{i,0}^\leftarrow$ occurred, $r_{i,0}$ must have occurred afterwards, as it is the only way X_i can be consumed.

We then have that:

$$\mathbf{c}_j \rightsquigarrow^{r_{i,0}^\leftarrow} \mathbf{c}_{j+1} \rightsquigarrow \mathbf{c}_k \xrightarrow{r_{i,0}} \mathbf{c}_{k+1}$$

where $\mathbf{c}_{j+1} \rightsquigarrow \mathbf{c}_k$ by a bi-execution where $r_{i,0}$ does not occur in forward or in reverse. Since none of the products of $r_{i,0}^\leftarrow$ (X_i) are reactants of any of the reactions that occur between $\mathbf{c}_{j+1}$ and $\mathbf{c}_k$, we can “shuffle” $r_{i,0}^\leftarrow$ forward using Lemma 5 such that:

$$\mathbf{c}_j \rightsquigarrow \mathbf{c}'_{k-1} \rightsquigarrow^{r_{i,0}^\leftarrow} \mathbf{c}_k \xrightarrow{r_{i,0}} \mathbf{c}_{k+1}$$

where $\mathbf{c}_j \rightsquigarrow \mathbf{c}'_{k-1}$ by a bi-execution where $r_{i,0}$ does not occur in forward or in reverse. We may then cancel out $r_{i,0}^\leftarrow$ and $r_{i,0}$, and so $\mathbf{c}_j \rightsquigarrow \mathbf{c}_{k+1}$ by a bi-execution where $r_{i,0}$ does not occur in forward or in reverse.

By repeatedly canceling out any reverse-applications of $r_{i,0}$ in this manner, we see that $\mathbf{i} \rightsquigarrow \mathbf{c}$ by an execution where $r_{i,0}^\leftarrow$ never occurs for all i .

We now further argue that $\mathbf{i} \rightsquigarrow \mathbf{c}$ by an execution where $r_{i,0}$ is never reverse-applied for all i and $\mathbf{i}^*$ is reached at some point. Since we have an execution that starts at $\mathbf{i}$ and ends at $\mathbf{c}$ where $r_{i,0}^\leftarrow$ never occurs for all i , no reactions that have the reactants of $r_{i,0}$ (X_i) as products occur for all i . This allows us to shuffle the applications of $r_{i,0}$ to the beginning of the execution using Lemma 4 and Lemma 5:

$$\mathbf{i} \rightarrow \mathbf{i}' \rightsquigarrow \mathbf{c}$$

where $\mathbf{i} \rightarrow \mathbf{i}'$ by a forward-execution where only $r_{i,0}$ are applied and $\mathbf{i}' \rightsquigarrow \mathbf{c}$ by a bi-execution where $r_{i,0}$ does not occur in forward or in reverse for all i . It is easy to see that $\mathbf{i}' = \mathbf{i}^*$, and so it follows that there exists some stable $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$ and the individual outputs are correct. It follows that $\mathbf{c} \rightarrow \mathbf{o}$. Consequently, for all valid input configurations $\mathbf{i}$, for all configurations $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$, there exists some stable configuration $\mathbf{o}$ such that $\mathbf{c} \rightarrow \mathbf{o}$ and the individual outputs of both $\mathcal{C}_1$ and $\mathcal{C}_2$ are correct. $\blacktriangleleft$

2.5 Invariants

For a given CRN, we define an *invariant* to be a function $I : \mathbb{N}^\Lambda \rightarrow \mathbb{N}$ such that for any valid configurations $\mathbf{c}$ and $\mathbf{c}'$, if $\mathbf{c} \rightsquigarrow \mathbf{c}'$, then $I(\mathbf{c}) = I(\mathbf{c}')$. Note that if I is an invariant, so is kI for all $k \in \mathbb{N}$ and if I_1, I_2 are invariants, so is $I_1 + I_2$. This paper uses what we refer to as linear and modular invariants:

- An invariant is *linear* if it is of the form $I_{\text{linear}}(\mathbf{c}) = \sum_{j=1}^k w_j \mathbf{c}(\lambda_j)$ for $w_1, \dots, w_k \in \mathbb{Z}$.
- An invariant is *modular* if it is of the form $I_{\text{modular}}(-\mathbf{a} + \mathbf{p}) = \left(\sum_{j=1}^k w_j \mathbf{c}(\lambda_j) \right) \bmod m$ for $w_1, \dots, w_k \in \mathbb{Z}$ and $m \in \mathbb{N}$. We refer to m as the *modulus* of the invariant.

Further, we refer to a linear function as the *linearization* of a modular invariant if taking the linear function modulo m yields the modular invariant. For example, in the following CRN:



Examining reactions (5)-(7), note that when A_i reacts with A_j , the product is $A_{i+j \bmod 3}$. Similarly, examining reaction (8), note that B acts as a catalyst to turn copies of B into copies of C . We then see that $I_1(\mathbf{c}) = \left(\sum_{i=1}^2 i\mathbf{c}(A_i) \right) \bmod 3$ is a modular invariant and that $I_2(\mathbf{c}) = \mathbf{c}(B) + \mathbf{c}(C)$ is a linear invariant. The linearization of I_1 is $I_{1,\text{lin}} = \sum_{i=1}^2 i(\mathbf{c}(A_i))$.

► **Observation 7.** *To check whether a quantity I is a linear or modular invariant, it is sufficient to check that for all reactions $r = (\mathbf{a}, \mathbf{p})$, $I(\mathbf{a}) - I(\mathbf{p}) = 0$.*

To see why this is true, let $\mathbf{c}$ and $\mathbf{c}'$ be arbitrary configurations such that $\mathbf{c} \rightarrow \mathbf{c}'$ by a single (forward or reverse) reaction. Then let $r = (\mathbf{a}, \mathbf{p})$ such that $\mathbf{c} - \mathbf{a} + \mathbf{p} = \mathbf{c}'$. If I is a linear invariant:

$$\begin{aligned} I(\mathbf{a}) - I(\mathbf{p}) &= I(\mathbf{a}) - I(\mathbf{p}) + I(\mathbf{c}') - I(\mathbf{c}) \\ &= I(\mathbf{a}) - I(\mathbf{p}) + I(\mathbf{c} - \mathbf{a} + \mathbf{p}) - I(\mathbf{c}) \\ &= I(\mathbf{a}) - I(\mathbf{p}) + I(\mathbf{c}) - I(\mathbf{a}) + I(\mathbf{p}) - I(\mathbf{c}) = 0 \end{aligned}$$

If I is a modular invariant with modulus m :

$$\begin{aligned} I(\mathbf{a}) - I(\mathbf{p}) &= I(\mathbf{a}) - I(\mathbf{p}) + I(\mathbf{c}') - I(\mathbf{c}) \\ &= (I_{\text{lin}}(\mathbf{a}) - I_{\text{lin}}(\mathbf{p}) + I_{\text{lin}}(\mathbf{c}') - I_{\text{lin}}(\mathbf{c})) \bmod m \\ &= (I_{\text{lin}}(\mathbf{a}) - I_{\text{lin}}(\mathbf{p}) + I_{\text{lin}}(\mathbf{c} - \mathbf{a} + \mathbf{p}) - I_{\text{lin}}(\mathbf{c})) \bmod m \\ &= (I_{\text{lin}}(\mathbf{a}) - I_{\text{lin}}(\mathbf{p}) + I_{\text{lin}}(\mathbf{c}) - I_{\text{lin}}(\mathbf{a}) + I_{\text{lin}}(\mathbf{p})) - I_{\text{lin}}(\mathbf{c})) \bmod m = 0 \end{aligned}$$

3 Exactly the semilinear sets are reverse-robustly decidable

In this section, we will show that under the reverse-robust computation model, CRDs have the same computational power as under the stable computation model. The following is the main result of the section:

► **Theorem 8.** *Exactly the semilinear sets can be reverse-robustly decided by a CRD.*

Since semilinear sets are Boolean combinations of mod and threshold predicates, we prove this theorem by showing that both mod and threshold sets can be reverse-robustly decided, as well as any Boolean combination of such.

3.1 All mod sets can be reverse-robustly decided

► **Lemma 9.** *Every mod set $M = \{(x_1, \dots, x_k) \mid \sum_{i=1}^n w_i x_i \equiv c \bmod m\}$ (where $w_i, m, c \in \mathbb{N}$ with $0 \leq w_i, c < m$) is reverse-robustly decidable by a CRD.*

We employ a modified version of the construction from [1]. Let the set of input species be $\Sigma = \{X_1, \dots, X_k\}$. We then add the following reaction, for each $i \in \{1, \dots, k\}$:



Then, for each $p, q \in \{0, \dots, m-1\}$ add the reactions:



Let the set of yes voters be $\Gamma_1 = \{Y_c\}$ and the set of no voters be $\Gamma_0 = \{Y_p \mid p \neq c\}$. Intuitively, the CRN does leader election, with the leader accumulating the weighted sum $\bmod m$.

Proof. By inspecting the reactions, we see that the CRN has the modular invariant:

$$I_M(\mathbf{c}) = \left(\sum_{i=1}^k w_i \mathbf{c}(X_i) + \sum_{p=0}^{m-1} p \mathbf{c}(Y_p) \right) \bmod m$$

For an initial configuration $\mathbf{i}$ where $\mathbf{i}(X_i) = x_i$, the invariant has value:

$$I_M(\mathbf{i}) = \sum_{i=1}^k w_i \mathbf{c}(X_i) \bmod m = \sum_{i=1}^d w_i x_i \bmod m$$

We now prove that this construction reverse-robustly decides M . Let $\mathbf{i}$ be an arbitrary initial configuration, where $\mathbf{i}(X_i) = x_i$, and let $\mathbf{c}$ such that $\mathbf{i} \rightarrow^* \mathbf{c}$. If any X_i are present in $\mathbf{c}$, we first apply reaction (9) repeatedly until X_i is no longer present. We then apply reaction (10) repeatedly until only one Y_{p^*} is present and no more reactions can occur; the resulting configuration $\mathbf{o}$ must then be stable. To see that $\mathbf{o}$ also has the correct output, note that the invariant has value:

$$I_M(\mathbf{o}) = p^* \mathbf{o}(Y_{p^*}) \bmod m = p^*$$

The remaining Y_{p^*} is then a yes voter if and only if $\sum_{i=1}^d w_i x_i \equiv c \bmod m$. Since $\mathbf{o}$ is stable with the correct output and $\mathbf{c} \rightarrow \mathbf{o}$, the CRD reverse-robustly decides M . $\blacktriangleleft$

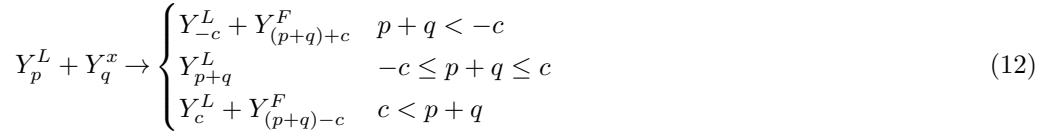
3.2 All threshold sets can be reverse-robustly decided

► **Lemma 10.** *Every threshold set $T = \{(x_1, \dots, x_k) \mid \sum_{i=1}^k w_i x_i \geq t\}$ (where $w_i, t \in \mathbb{Z}$) is reverse-robustly decidable by a CRD.*

We employ a modified version of the construction from [1]. Let the set of input species be $\Sigma = \{X_1, \dots, X_k\}$ and define c such that $c = \max\{|w_1|, \dots, |w_k|, |t|\} + 1$. We then add the following reaction, for each $i \in \{1, \dots, k\}$:



Then, for each $p, q \in \{-c, \dots, c\}$ and $x \in \{L, F\}$ add the reactions:



Let the set of yes voters be $\Gamma_1 = \{Y_p^L \mid p \geq t\}$ and the set of no voters be $\Gamma_0 = \{Y_p^L \mid p < t\}$. Intuitively, the CRN does leader election, with the leader accumulating the weighted sum, clipped to the range $[-c, c]$ so that the set of species is finite. Any excess is accumulated by one or more of the followers.

Proof. By inspecting the reactions, we see that the CRN has the linear invariant:

$$I_T(\mathbf{c}) = \sum_{i=0}^k w_i \mathbf{c}(X_i) + \sum_{p=-c}^c p \mathbf{c}(Y_p^L) + \sum_{q=-c}^c q \mathbf{c}(Y_q^F)$$

For an initial configuration $\mathbf{i}$ where $\mathbf{i}(X_i) = x_i$, the invariant has value:

$$I_T(\mathbf{i}) = w_i \mathbf{c}(X_i) = \sum_{i=1}^k w_i x_i$$

We now prove that this construction reverse-robustly decides T . Let $\mathbf{i}$ be an arbitrary initial configuration, where $\mathbf{i}(X_i) = x_i$, and let $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$. If any X_i are present in $\mathbf{c}$, we first apply reaction (11) repeatedly until X_i is no longer present. We then apply reaction (12) repeatedly until only one $Y_{p^*}^L$ is present (possibly in addition to some Y_q^F) and no more reactions can occur; the resulting configuration $\mathbf{o}$ must then be stable. To see that $\mathbf{o}$ also has the correct output, we consider two cases.

In the case that $Y_{p^*}^L$ is the only remaining molecule, the invariant has value:

$$I_T(\mathbf{o}) = p^* \mathbf{o}(Y_{p^*}^L) = p^*$$

The remaining $Y_{p^*}^L$ is then a yes voter if and only if $\sum_{i=1}^k w_i x_i \geq t$.

Alternatively, if some Y_q^F are present, we first note that either $p^* = -c$ or $p^* = c$ and further, p^* and all q must have the same sign; otherwise reaction (12) would be applicable. If $p^* = c > 0$, the invariant has value:

$$I_T(\mathbf{o}) = c \mathbf{o}(Y_c^L) + \sum_{q=1}^c q \mathbf{o}(Y_q^F) = c + \sum_{q=1}^c q \mathbf{o}(Y_q^F)$$

The remaining Y_c^L is a yes voter, which is correct since $\sum_{i=1}^k w_i x_i \geq t = c + \sum_{q=1}^c q \mathbf{o}(Y_q^F) > c$. By a similar argument, if $p^* = -c$, the remaining Y_{-c}^L is correctly a no voter.

Since $\mathbf{o}$ is stable with the correct output and $\mathbf{c} \rightarrow \mathbf{o}$ in all cases, the CRD reverse-robustly decides T . $\blacktriangleleft$

3.3 Reverse-robustly decidable sets are closed under Boolean operations

► **Lemma 11.** *If sets $X_1, X_2 \subseteq \mathbb{N}^d$ are reverse-robustly decided by some CRDs $\mathcal{D}_1$ and $\mathcal{D}_2$, then so are $X_1 \cup X_2$, $X_1 \cap X_2$, and $\overline{X_1}$.*

Proof. To reverse-robustly decide $\overline{X_1}$, swap the yes and no voters. As the reactions themselves remain unchanged, it is trivial to show that this construction is reverse-robust.

To show that $X_1 \cup X_2$ and $X_1 \cap X_2$ are reverse-robustly decidable, consider a construction where both sets are decided in parallel and their votes are recorded in a new voter species. We first add reactions that duplicate the input and produce one of the four new voter species $V_{NN}, V_{NY}, V_{YN}, V_{YY}$:



Note that V_{NN} is chosen arbitrarily. $X_{i,1}$ and $X_{i,2}$ then replace X_i in any reactions involving X_i in $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. We then add reactions that store the separate votes in the new voter species. The votes of the $\mathcal{D}_1$ and $\mathcal{D}_2$ are stored in the first and second subscript of the new voter species, respectively. For each $b \in \{Y, N\}$ and for every S_b and T_b that vote in $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively, and for each $x \in \{Y, N\}$, we add the reactions:



To reverse-robustly decide $X_1 \cup X_2$, we let the yes voters be $\Gamma = \{V_{NY}, V_{YN}, V_{YY}\}$ and to reverse-robustly decide $X_1 \cap X_2$, we let the yes voters be $\Gamma = \{V_{YY}\}$.

We now prove that this construction reverse-robustly decides $X_1 \cup X_2$ or $X_1 \cap X_2$. Let $\mathbf{i}$ be an arbitrary initial configuration, where $\mathbf{i}(X_i) = x_i$, and let $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$. In the configuration $\mathbf{c}$, the voter species may or may not have the correct vote—depending on the reactions applied they may end up as any of V_{NN} , V_{NY} , V_{YN} , or V_{YY} . If any X_i are present in $\mathbf{c}$, we first apply reactions (13) repeatedly until X_i is no longer present. We then run $\mathcal{D}_1$ and $\mathcal{D}_2$ until no more of their reactions can occur, then finally do the same for reactions (14) and (15); this resulting configuration $\mathbf{o}$ must then be stable. Since $\mathcal{D}_1$ and $\mathcal{D}_2$ reverse-robustly decide X_1 and X_2 and their voters do not have their counts changed by reactions (14) and (15), by Lemma 6, their individual votes are stable and correct in $\mathbf{o}$. If any of the new voter molecules with the wrong overall vote are present, either (14) and (15) would be applicable; all of the new voter molecules must then have the correct overall vote. Since $\mathbf{o}$ is stable with the correct output, the CRD reverse-robustly decides $X_1 \cup X_2$ or $X_1 \cap X_2$, depending on how the yes and no voters are selected. ◀

4 Exactly the semilinear functions are reverse-robustly computable

In this section, we will show that under the reverse-robust computation model, CRCs have the same computational power as under the stable computation model. The following is the main result of the section:

► **Theorem 12.** *Exactly the semilinear functions can be reverse-robustly computed by a CRC.*

4.1 Diff-representations of all partial affine functions can be reverse-robustly computed

We say that a partial function $f : \mathbb{N}^k \rightarrow \mathbb{N}$ is *affine* if there exist vectors $a_1, \dots, a_k \in \mathbb{Q}$, $c_1, \dots, c_k \in \mathbb{N}$, and $b \in \mathbb{N}$ such that when $\mathbf{x}(i) - c_i \geq 0$:

$$f(\mathbf{x}) = b + \sum_{i=1}^k a_i(\mathbf{x}(i) - c_i)$$

Note that f is not defined for all $\mathbf{x} \in \mathbb{N}^k$; not only must $\mathbf{x}(i) \geq c_i$ for all i , but $b + \sum_{i=1}^k a_i(\mathbf{x}(i) - c_i)$ must be a nonnegative integer. For a partial function f we write $\text{dom} f$ for the domain of f , the set of inputs for which f is defined. For convenience, we can work with integer valued molecule counts by factoring out the reduced denominator d the dot product, where d may be taken to be the least common multiple of the denominators of the rational coefficients in the original definition such that $n_i = d \cdot a_i$:

$$f(\mathbf{x}) = b + \sum_{i=1}^k a_i(\mathbf{x}(i) - c_i) \iff f(\mathbf{x}) = b + \frac{1}{d} \sum_{i=1}^k n_i(\mathbf{x}(i) - c_i)$$

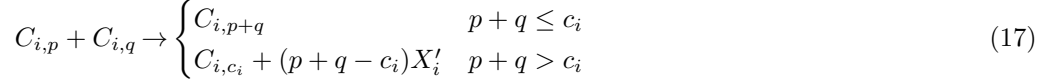
We say that a partial function $\hat{f} : \mathbb{N}^k \rightarrow \mathbb{N}^2$ is a *diff-representation* of f if $\text{dom} f = \text{dom} \hat{f}$ and for all $x \in \text{dom} f$, if $(y_P, y_C) = \hat{f}(\mathbf{x})$, then $f(\mathbf{x}) = y_P - y_C$, and $y_P = O(f(x))$. In other words, $\hat{f}$ represents f as the difference of its two outputs y_P and y_C , the larger output y_P possibly being larger than the output of the original function, but at most a multiplicative constant larger.

► **Lemma 13.** *Let $f : \mathbb{N}^k \rightarrow \mathbb{N}$ be an affine partial function. Then there is a diff-representation $\hat{f} : \mathbb{N}^k \rightarrow \mathbb{N}^2$ and a CRC that reverse-robustly computes $\hat{f}$.*

We employ the construction from [3]. Let the set of input species be $\Sigma = \{X_1, \dots, X_k\}$ and the set of output species be $\Gamma = \{Y^P, Y^C\}$. Then, for each $i \in \{1, \dots, k\}$, we add the following reaction:



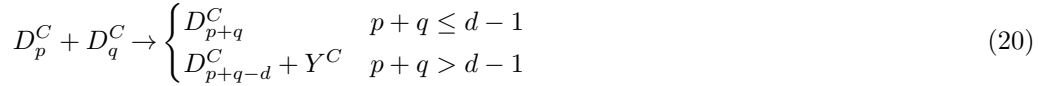
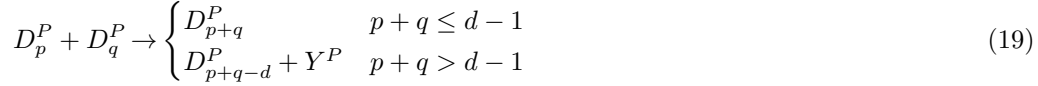
Then, for each $i \in \{1, \dots, k\}$ and for each $p, q \in \{1, \dots, c_i\}$, add the reactions:



Then, for each $i \in \{1, \dots, k\}$, add the reaction:



Then, for each $p, q \in \{1, \dots, d-1\}$, add the reactions:



Then, add the reaction:



Proof. By inspecting the reactions, we see that the CRN has the linear invariant:

$$\begin{aligned} I_{\hat{f}}(\mathbf{c}) = & \sum_{i=1}^k n_i \left(\mathbf{c}(X_i) + \sum_{p=1}^{c_i} p \mathbf{c}(C_{i,p}) + \sum_{i=1}^k \mathbf{c}(X'_i) \right) \\ & + \sum_{p=1}^{d-1} p \mathbf{c}(D_p^P) - \sum_{q=1}^{d-1} q \mathbf{c}(D_q^C) + d(\mathbf{c}(Y^P) - \mathbf{c}(Y^C)) - b d \mathbf{c}(B) \end{aligned}$$

in addition to the modular invariants:

$$\begin{aligned} I_P(\mathbf{c}) &= \left(\sum_{n_i \geq 0} n_i \left(\mathbf{c}(X_i) + \sum_{p=1}^{c_i} p \mathbf{c}(C_{i,p}) + \sum_{i=1}^k \mathbf{c}(X'_i) \right) + \sum_{p=1}^{d-1} p \mathbf{c}(D_p^P) \right) \bmod d \\ I_C(\mathbf{c}) &= \left(\sum_{n_i < 0} n_i \left(\mathbf{c}(X_i) + \sum_{p=1}^{c_i} p \mathbf{c}(C_{i,p}) + \sum_{i=1}^k \mathbf{c}(X'_i) \right) - \sum_{q=1}^{d-1} q \mathbf{c}(D_q^C) \right) \bmod d \end{aligned}$$

For an initial configuration $\mathbf{i}$ where $\mathbf{i}(X_i) = x_i$, the invariants have values:

$$\begin{aligned} I_{\hat{f}}(\mathbf{i}) &= \sum_{i=1}^k n_i \mathbf{c}(X_i) = \sum_{i=1}^k n_i x_i \\ I_P(\mathbf{i}) &= \left(\sum_{n_i \geq 0} n_i \mathbf{i}(X_i) \right) \bmod d = \left(\sum_{n_i \geq 0} n_i x_i \right) \bmod d \\ I_C(\mathbf{i}) &= \left(\sum_{n_i < 0} n_i \mathbf{i}(X_i) \right) \bmod d = \left(\sum_{n_i < 0} n_i x_i \right) \bmod d \end{aligned}$$

We now prove that this construction reverse-robustly calculates $\hat{f}$. Let $\mathbf{i}$ be an arbitrary initial configuration, where $\mathbf{i}(X_i) = x_i$ and $x_i - c_i \geq 0$, and let $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$. If any X_i are present in $\mathbf{c}$, we first apply reactions (16) repeatedly. We then apply reaction (17) repeatedly until only one C_{i,s_i^*} is present for each i and no more of that reaction can occur. We then do the same for reaction (18) until no more X_i' are present, and then reactions (19) and (20) until only one $D_{p^*}^P$ and $D_{q^*}^C$ are present. Finally, we do the same for reaction (21) until only one B is present; this resulting configuration $\mathbf{o}$ must then be stable. To see that $\mathbf{o}$ also has the correct output, we first consider the following quantity:

$$I_{c,i}(\mathbf{c}) = \mathbf{c}(X_i) + \sum_{p=1}^{c_i} p\mathbf{c}(C_{i,p})$$

In the initial configuration, the quantity has value $I_{c,i}(\mathbf{i}) = \mathbf{i}(X_i) = x_i \geq c_i$. By inspecting reaction (16), as well as reactions (18) through (21), we see that those reactions preserve the quantity. For reaction (17), however, while the quantity is preserved whenever $p + q \leq c_i$, this is not true whenever $p + q > c_i$, as $\sum_{p=1}^{c_i} p\mathbf{c}(C_{i,p})$ changes. Specifically, it decreases when the reaction is applied in forward and increases when applied in reverse. Even so, since C_{i,c_i} is one of the products of the forward reaction, we see that the quantity remains at least c_i for any $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$. Since only one C_{i,s_i^*} is present for each i and no X_i are present, the quantity has the value $I_{c,i}(\mathbf{o}) = s_i^* \mathbf{o}(C_{i,s_i^*}) = s_i^*$, and it must be then be the case that $s_i^* = c_i$.

We next consider the modular variants I_P and I_C . In $\mathbf{o}$, I_P has the value:

$$I_P(\mathbf{o}) = \left(\sum_{n_i \geq 0} n_i c_i \mathbf{c}(C_{i,c_i}) + p^* \mathbf{c}(D_{p^*}^P) \right) \bmod d = \left(\sum_{n_i \geq 0} n_i c_i + p^* \right) \bmod d$$

For this to be true, $p^* = \sum_{n_i \geq 0} n_i (x_i - c_i) \bmod d$. By a similar argument, $q^* = \sum_{n_i < 0} (-n_i) \cdot (x_i - c_i) \bmod d$. Note that for $f(\mathbf{x})$ to be an integer, $\sum_{i=1}^k n_i (\mathbf{x}(i) - c_i) \bmod d = (p^* - q^*) \bmod d = 0$. Since $0 \leq p^*, q^* < d$, it must be the case that $p^* - q^* = 0$. We then see that $I_{\hat{f}}$ has value:

$$\begin{aligned} I_{\hat{f}}(\mathbf{o}) &= \sum_{i=1}^k n_i c_i \mathbf{o}(C_{i,c_i}) + p^* \mathbf{o}(D_{p^*}^P) - q^* \mathbf{o}(D_{q^*}^C) + d(\mathbf{o}(Y^P) - \mathbf{o}(Y^C)) - bd\mathbf{o}(B) \\ &= \sum_{i=1}^k n_i c_i + p^* - q^* + d(\mathbf{o}(Y^P) - \mathbf{o}(Y^C)) - bd \\ &= \sum_{i=1}^k n_i c_i + d(\mathbf{o}(Y^P) - \mathbf{o}(Y^C)) - bd \end{aligned}$$

For this to be true, it must be the case that:

$$\mathbf{o}(Y^P) - \mathbf{o}(Y^C) = b + \frac{1}{d} \sum_{i=1}^k n_i (x_i - c_i) = f(\mathbf{x})$$

Since $\mathbf{o}(Y^P) - \mathbf{o}(Y^C) = f(\mathbf{x})$, the CRC reverse-robustly computes $\hat{f}$. ◀

4.2 All semilinear functions can be reverse-robustly computed

We require the following result from [4], which guarantees that any semilinear function can be decomposed into affine partial functions with disjoint domains.

► **Lemma 14.** *Let $f : \mathbb{N}^k \rightarrow \mathbb{N}$ be a semilinear function. Then there is a finite set $\{f_1 : \mathbb{N}^k \rightarrow \mathbb{N}, \dots, f_m : \mathbb{N}^d \rightarrow \mathbb{N}\}$ of affine partial functions, where each $\text{dom}f_i$ is a linear set, and $\text{dom}f_i \cap \text{dom}f_j = \emptyset$ for all $i \neq j$, such that, for each $x \in \mathbb{N}^k$, if $\mathbf{x} \in \text{dom}f_i$, then $f(\mathbf{x}) = f_i(\mathbf{x})$, and $\bigcup_{i=1}^m \text{dom}f_i = \mathbb{N}^k$.*

The next theorem shows that semilinear functions can be reverse-robustly computed.

► **Theorem 15.** *Let $f : \mathbb{N}^k \rightarrow \mathbb{N}$ be a semilinear function. Then there is a CRC that reverse-robustly computes f .*

We employ the construction from [3]. Let the set of input species be $\Sigma = \{X_1, \dots, X_k\}$ and the output species be Y . By Lemma 14, f can be decomposed into partial affine functions $f_1, \dots, f_m$ with disjoint linear domains, and by Theorem 8 and Lemma 13, each of the partial affine functions and their domains can be computed reverse-robustly. By duplicating X_i , we can reverse-robustly decide the predicate $\mathbf{x} \in \text{dom}f_j$ and compute $\hat{f}_j(\mathbf{x})$ in parallel for each f_j . Let L_j^Y and L_j^N be the yes and no voters, respectively, for the CRD $\mathcal{D}_j$ reverse-robustly deciding if $\mathbf{x} \in \text{dom}f_j$ and let $\hat{Y}_j^P$ and $\hat{Y}_j^C$ be the output species for the CRC $\mathcal{C}_j$ reverse-robustly computing $\hat{f}_j$, using the constructions from the previous sections. We interpret each $\hat{Y}_j^P$ and $\hat{Y}_j^C$ as “inactive” versions of the “active” output species Y_j^P and Y_j^C . Now, we convert the function result of the applicable partial affine function to the global output by adding the following reactions for each $j \in \{1, \dots, m\}$ and for each L_j^Y that votes yes and L_j^N that votes no in $\mathcal{D}_j$:



Reaction (22) produces an output copy of species Y and (23) undoes the first reaction. Both are catalyzed by the vote of the i -th predicate result. We also add the reactions:



and



Proof. We now prove that this construction reverse-robustly decides f . Let $\mathbf{i}$ be an arbitrary initial configuration, where $\mathbf{i}(X_i) = x_i$, and let $\mathbf{c}$ such that $\mathbf{i} \rightsquigarrow \mathbf{c}$. We first run each $\mathcal{D}_j$ and $\mathcal{C}_j$ in parallel until no more of the inactive output species can be produced. We then run reactions (22) through (26) until no more reactions can occur; this resulting configuration $\mathbf{o}$ must be stable.

To see that $\mathbf{o}$ also has the correct output, since each $\mathcal{D}_j$ reverse-robustly decides $\mathbf{x} \in \text{dom}f_j$ and the counts of their voters are not changed by reactions (22) through (26), by Lemma 6, their individual votes are stable and correct in $\mathbf{o}$. Although each $\mathcal{C}_j$ reverse-robustly computes $\hat{f}_j$, since their output is consumed, we cannot apply Lemma 6. However, let $I_{\hat{f}_j}$ be the linear invariant derived for each $\mathcal{C}_j$. We see that for each $j \in \{1, \dots, m\}$, appropriately renaming species, constant, and indices, the CRN has the linear invariant:

$$\begin{aligned}
I_j(\mathbf{c}) &= \sum_{i=1}^k n_{i,j} \mathbf{c}(X_i) + I_{f_j}(\mathbf{c}) + d_j(\mathbf{c}(Y_j^P) - \mathbf{c}(Y_j^C)) \\
&= \sum_{i=1}^k n_{i,j} \left(\mathbf{c}(X_i) + \mathbf{c}(X_{i,j}) + \sum_{p=1}^{c_{i,j}} p \mathbf{c}(C_{i,p,j}) + \sum_{i=1}^k \mathbf{c}(X'_{i,j}) \right) \\
&\quad + \sum_{p=1}^{d_j-1} p \mathbf{c}(D_{p,j}^P) - \sum_{q=1}^{d_j-1} q \mathbf{c}(D_{q,j}^C) + d_j(\mathbf{c}(\hat{Y}_j^P) + \mathbf{c}(Y_j^P) - \mathbf{c}(\hat{Y}_j^C) - \mathbf{c}(Y_j^C)) - b_j d_j \mathbf{c}(B_j)
\end{aligned}$$

By a similar argument to the partial affine case, it follows that:

$$\mathbf{o}(\hat{Y}_j^P) + \mathbf{o}(Y_j^P) - \mathbf{o}(\hat{Y}_j^C) - \mathbf{o}(Y_j^C) = b_j + \frac{1}{d_j} \sum_{i=1}^k n_{i,j} (x_i - c_{i,j}) = f_j(\mathbf{x})$$

We further see by inspecting the reactions that the CRN has the additional linear invariant:

$$I_0(\mathbf{c}) = \sum_{j=1}^m \mathbf{c}(Y_j^P) - \mathbf{c}(Y)$$

For an initial configuration $\mathbf{i}$ where $\mathbf{i}(X_i) = x_i$, the invariant has value:

$$I_0(\mathbf{i}) = 0$$

We argue that in $\mathbf{o}$, there is exactly one j^* such that $L_{j^*}^Y$ is present, and that $\mathbf{x} \in \text{dom } f_{j^*}$; this follows from each $\mathcal{D}_j$ reverse-robustly deciding their predicate, and their voters deactivating the incorrect output species. It must then be the case that there is exactly one j^* (the same j^*) such that $Y_{j^*}^P$ and $Y_{j^*}^C$ can be present and further, that $\hat{Y}_{j^*}^P$ and $\hat{Y}_{j^*}^C$ are absent, as otherwise they could be activated by the voters of $\mathcal{D}_{j^*}$. It then follows that:

$$\mathbf{o}(Y_{j^*}^P) - \mathbf{o}(Y_{j^*}^C) = f_{j^*}(\mathbf{x})$$

Additionally, the invariant I_0 has the value:

$$I_0(\mathbf{o}) = \mathbf{o}(Y_{j^*}^P) - \mathbf{o}(Y)$$

For this to be true, $\mathbf{o}(Y_{j^*}^P) = \mathbf{o}(Y)$. Note that it is impossible for $Y_{j^*}^P$, $Y_{j^*}^C$, and Y to be simultaneously present, as otherwise reaction (26) could occur. If $\mathbf{o}(Y_{j^*}^P) = \mathbf{o}(Y) > 0$, $\mathbf{o}(Y_{j^*}^C) = 0$. On the other hand, if $\mathbf{o}(Y_{j^*}^P) = \mathbf{o}(Y) = 0$, since $f_{j^*}(\mathbf{x}) \geq 0$, $\mathbf{o}(Y_{j^*}^C) = 0$. In either case, $\mathbf{o}(Y_{j^*}^C) = 0$. Then:

$$\mathbf{o}(Y) = \mathbf{o}(Y_{j^*}^P) = f_{j^*}(\mathbf{x})$$

Since $\mathbf{x} \in \text{dom } f_{j^*}$, $\mathbf{o}(Y) = f(\mathbf{x})$, and the CRC reverse-robustly computes f . ◀

5 Conclusion

We explored the capabilities of chemical reaction networks (CRNs) under the reverse-robust computation model, where reactions are allowed to occur in reverse while maintaining the existence of a reachable correct stable configuration. This model aligns with thermodynamic theory assuming all species have a finite free energy, implying all chemical reactions are reversible, albeit at some possibly very slow rate. We showed that CRNs have the same

computational power under the reverse-robust computation model as they do under the stable computation model. Both semilinear predicates can be decided and semilinear functions can be computed reverse-robustly without an initial leader.

Some key questions remain open: first, are there a set of conditions that are necessary and/or sufficient to show that a CRN is reverse-robust? This paper was able to use existing constructions designed for the stable computation model to prove results for the reverse-robust computation model. In fact, we were unable to design a CRN that works under the former model but not the latter without the presence of a “trap” reaction that cannot occur when only forward reactions are allowed. We thus conjecture that the absence of such reactions is sufficient to prove reverse-robustness (and so, that a stably computing CRN can be made reverse-robust by removing such reactions without affecting its correctness). Second, our model only allows reverse-reactions to occur transiently, as we require that only forward reactions occur in the final path to reach a stable state. Is it possible to formally define some reasonable model of computation in which reverse-reactions are always allowed to occur? Note that such a model cannot require stability, as it is always possible to reverse the path of reactions that led to a state. Instead, we may require “favorability”; although temporary reversal to incorrect states can occur, the correct answer is present “most of the time”.

References

- 1 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Comput.*, 18(4):235–253, 2006. doi:10.1007/S00446-005-0138-3.
- 2 Ho-Lin Chen, David Doty, and David Soloveichik. Deterministic function computation with chemical reaction networks. *Nat. Comput.*, 13(4):517–534, 2014. doi:10.1007/S11047-013-9393-6.
- 3 David Doty and Monir Hajiaghayi. Leaderless deterministic chemical reaction networks. *Nat. Comput.*, 14(2):213–223, 2015. doi:10.1007/S11047-014-9435-8.
- 4 David Doty and Ben Heckmann. The computational power of discrete chemical reaction networks with bounded executions. In Dan Alistarh, editor, *38th International Symposium on Distributed Computing, DISC 2024, Madrid, Spain, October 28 – November 1, 2024*, LIPIcs, pages 20:1–20:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.DISC.2024.20.

A Commuting reactions

We restate Lemmas 4 and 5 below for convenience and provide their proofs.

► **Lemma 4.** *Let $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2$ such that $\mathbf{c}_0 \xrightarrow{r_1} \mathbf{c}_1 \xrightarrow{r_2} \mathbf{c}_2$, where $r_1 = (\mathbf{a}_1, \mathbf{p}_1)$ and $r_2 = (\mathbf{a}_2, \mathbf{p}_2)$. If $\mathbf{p}_1 \cap \mathbf{a}_2 = \emptyset$ (none of the products of r_1 are reactants of r_2), then there is some $\mathbf{c}'_1$ such that $\mathbf{c}_0 \xrightarrow{r_2} \mathbf{c}'_1 \xrightarrow{r_1} \mathbf{c}_2$.*

Proof. Since $\mathbf{c}_0 \xrightarrow{r_1} \mathbf{c}_1$ and $\mathbf{c}_1 \xrightarrow{r_2} \mathbf{c}_2$, $\mathbf{c}_0 \geq \mathbf{a}_1$ and $\mathbf{c}_1 \geq \mathbf{a}_2$. If $\mathbf{p}_1(\lambda) > 0$ (λ is a product of r_1), since we have assumed by hypothesis that it is not a reactant of r_2 , then $\mathbf{a}_2(\lambda) = 0$. Consequently, $\mathbf{a}_1(\lambda) + \mathbf{a}_2(\lambda) = \mathbf{a}_1(\lambda)$, and so $\mathbf{c}_0(\lambda) \geq \mathbf{a}_1(\lambda) + \mathbf{a}_2(\lambda)$. On the other hand, if λ is not a product of r_1 ($\mathbf{p}_1(\lambda) = 0$)

$$\begin{aligned}
 \mathbf{c}_0(\lambda) &= \mathbf{c}_1(\lambda) + \mathbf{a}_1(\lambda) - \mathbf{p}_1(\lambda) \\
 &= \mathbf{c}_1(\lambda) + \mathbf{a}_1(\lambda) && \text{since } \mathbf{p}_1(\lambda) = 0 \\
 &\geq \mathbf{a}_2(\lambda) + \mathbf{a}_1(\lambda). && \text{since } \mathbf{c}_1(\lambda) \geq \mathbf{a}_2(\lambda)
 \end{aligned}$$

It then follows that $\mathbf{c}_0 \geq \mathbf{a}_1 + \mathbf{a}_2$ implying $\mathbf{c}_0 \geq \mathbf{a}_2$ since $\mathbf{a}_1 \geq \mathbf{0}$. Consequently, r_2 is applicable to $\mathbf{c}_0$; let $\mathbf{c}'_1 = \mathbf{c}_0 - \mathbf{a}_2 + \mathbf{p}_2$ be the resulting configuration. It remains to show r_1 is applicable to $\mathbf{c}'_1$, i.e., $\mathbf{c}'_1 \geq \mathbf{a}_1$. We have

$$\begin{aligned} \mathbf{c}'_1 &= \mathbf{c}_0 - \mathbf{a}_2 + \mathbf{p}_2 \\ &\geq (\mathbf{a}_1 + \mathbf{a}_2) - \mathbf{a}_2 + \mathbf{p}_2 && \text{since } \mathbf{c}_0 \geq \mathbf{a}_1 + \mathbf{a}_2 \\ &= \mathbf{a}_1 + \mathbf{p}_2 \geq \mathbf{a}_1. && \text{since } \mathbf{p}_2 \geq \mathbf{0} \end{aligned} \quad \blacktriangleleft$$

► **Lemma 5.** *Let $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2$ such that $\mathbf{c}_0 \rightsquigarrow^{r_1 \leftarrow} \mathbf{c}_1 \xrightarrow{r_2} \mathbf{c}_2$, where $r_1 = (\mathbf{a}_1, \mathbf{p}_1)$ and $r_2 = (\mathbf{a}_2, \mathbf{p}_2)$. If $\mathbf{a}_1 \cap \mathbf{a}_2 = \emptyset$ (none of the products of $r_1 \leftarrow$ are reactants of r_2), then there is some $\mathbf{c}'_1$ such that $\mathbf{c}_0 \xrightarrow{r_2} \mathbf{c}'_1 \rightsquigarrow^{r_1 \leftarrow} \mathbf{c}_2$.*

Proof. Since $\mathbf{c}_0 \rightsquigarrow^{r_1 \leftarrow} \mathbf{c}_1$ and $\mathbf{c}_1 \xrightarrow{r_2} \mathbf{c}_2$, $\mathbf{c}_0 \geq \mathbf{p}_1$ and $\mathbf{c}_1 \geq \mathbf{a}_2$. If $\mathbf{a}_1(\lambda) > 0$ (λ is a product of $r_1 \leftarrow$), since we have assumed by hypothesis that it is not a reactant of r_2 , then $\mathbf{a}_2(\lambda) = 0$. Consequently, $\mathbf{p}_1(\lambda) + \mathbf{a}_2(\lambda) = \mathbf{p}_1(\lambda)$, and so $\mathbf{c}_0(\lambda) \geq \mathbf{p}_1(\lambda) + \mathbf{a}_2(\lambda)$. On the other hand, if λ is not a product of $r_1 \leftarrow$ ($\mathbf{a}_1(\lambda) = 0$)

$$\begin{aligned} \mathbf{c}_0(\lambda) &= \mathbf{c}_1(\lambda) + \mathbf{p}_1(\lambda) - \mathbf{a}_1(\lambda) \\ &= \mathbf{c}_1(\lambda) + \mathbf{p}_1(\lambda) && \text{since } \mathbf{a}_1(\lambda) = 0 \\ &\geq \mathbf{a}_2(\lambda) + \mathbf{p}_1(\lambda). && \text{since } \mathbf{c}_1(\lambda) \geq \mathbf{a}_2(\lambda) \end{aligned}$$

It then follows that $\mathbf{c}_0 \geq \mathbf{p}_1 + \mathbf{a}_2$ implying $\mathbf{c}_0 \geq \mathbf{a}_2$ since $\mathbf{a}_1 \geq \mathbf{0}$. Consequently, r_2 is applicable to $\mathbf{c}_0$; let $\mathbf{c}'_1 = \mathbf{c}_0 - \mathbf{a}_2 + \mathbf{p}_2$ be the resulting configuration. It remains to show $r_1 \leftarrow$ is applicable to $\mathbf{c}'_1$, i.e., $\mathbf{c}'_1 \geq \mathbf{p}_1$. We have

$$\begin{aligned} \mathbf{c}'_1 &= \mathbf{c}_0 - \mathbf{a}_2 + \mathbf{p}_2 \\ &\geq (\mathbf{p}_1 + \mathbf{a}_2) - \mathbf{a}_2 + \mathbf{p}_2 && \text{since } \mathbf{c}_0 \geq \mathbf{p}_1 + \mathbf{a}_2 \\ &= \mathbf{p}_1 + \mathbf{p}_2 \geq \mathbf{p}_1. && \text{since } \mathbf{p}_2 \geq \mathbf{0} \end{aligned} \quad \blacktriangleleft$$

We note that for the lemma above, stating that “none of the products of $r_1 \leftarrow$ are reactants of r_2 ” is equivalent to stating that none of the reactants of r_1 are reactants of r_2 .

Scaling up Thermodynamically Favoured Scaffolded DNA Computing by Sculpting the Energy Landscape

Joshua Petrack¹ ✉ 

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Constantine G. Evans¹ ✉ 

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Angel Cervera Roldan ✉ 

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Mahboobeh Enayati ✉ 

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Damien Woods ✉ 

Hamilton Institute and Department of Computer Science, Maynooth University, Ireland

Abstract

Thermodynamically favoured molecular computation offers advantages over more typical out-of-equilibrium computing, including simpler experimental protocols and automatic error correction. But as systems scale to large sizes the number of states increases dramatically, increasing the need for efficiently navigable energy landscapes. We give results in two theoretical models of Scaffolded DNA Computing (SDC), a recently implemented form of thermodynamically favoured DNA computing [Stérin, Eshra et al, bioRxiv 2025]. We show their computational power is characterised by logarithmic space complexity classes, meaning they are expressive at scale. Our first energy landscape result is an exact relation between the probability of target configurations (outputs), temperature and DNA sequence domain length, showing that domain length merely logarithmic in scaffold length is sufficient for the probability of the target configuration to outcompete all off-target structures. We show that even in the presence of imperfect/unequal binding strength scaffold domains we still achieve good kinetics: $O(N^2)$ or $O(N^3)$ expected completion time, depending on model assumptions, and there are even narrow conditions that yield fast $O(N)$ -time kinetics. Finally, we address a thorny scaling problem: SDC outputs often have repeated compute domains and any binding energy variances get exaggerated under repetition making errors favourable, but we give a construction that reprograms the energy landscape to convert such a non-isoenergetic system into one with almost perfectly isoenergetic energy plateaus. We also show that systems maintain good (polynomial-time) kinetics, even in the face of a poor (uphill) scaffold energy landscape. These results give a roadmap for scaling up the SDC while highlighting the role kinetics and energy landscape programming could play in thermodynamically-favoured computing more generally.

2012 ACM Subject Classification Applied computing → Chemistry; Applied computing → Physics; Theory of computation → Models of computation

Keywords and phrases Thermodynamically favoured computing, molecular programming, DNA computing, kinetics, computational power, Scaffolded DNA Computer

Digital Object Identifier 10.4230/LIPIcs.DNA.32.7

Supplementary Material *Collection (Source code and data for paper & figures):*
<https://doi.org/10.5281/zenodo.21441279>

¹ Equal contribution



Funding Supported by the European Union’s (EU) European Innovation Council (EIC) and SMEs Executive Agency (EISMEA), No 101115422, DISCO project, Research Ireland (RI) under grant number 20/FFP-P/8843, and Maynooth University. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the EU, EISMEA or RI. Neither the EU nor the granting authority can be held responsible for them.

Acknowledgements We thank Abeer Eshra, Tristan Stérin & Janet Adio for discussions on SDC scaling and data in Figure 4, and John Reif for conversations on complexity results.

1 Introduction

The idea behind thermodynamically-favoured molecular computing is to mix a soup of molecules, let them find their chemical equilibrium – *et voilà!* – we have the answer to a computation. The notion has roots in the work of Bennett [2, 3], who reasoned about reversible computation at equilibrium, in order to understand the ultimate energy costs in computing [6]. In the DNA computing and molecular programming field, one useful model has been Thermodynamic Binding Networks [7], which were recently implemented [21]. DNA nanostructures have had a thermodynamically-favoured method for some decades: DNA origami [14] works so well precisely because the intended structure is usually favoured and the folding landscape likely has efficient kinetic pathways – an ongoing topic of research [4, 8, 10, 16, 18]. Three key rules underlie DNA origami’s energetic favourability: (1) staples in high concentration excess bind to a long low-excess scaffold strand; (2) staples do not bind to each other; and (3) good 3D geometric design. The Scaffolded DNA Computer (SDC) [20] obeys the first rule, but carefully breaks the second, by having short strands bind both to a scaffold and to each other, crucially exchanging information and logic by doing so, see Figure 1. We desire thermodynamically favoured computers be easy to prepare (just mix the molecules) and easy to run (heat up, cool down). The scaffolded principle allows for straightforward implementation of thermodynamically-favoured computing, with multiple tunable system parameters to optimise the energy landscape. Here, we explore several.

The SDC was demonstrated on 10 programs, was shown to be reusable multiple (25) times, and even ran super-fast 16-bit computations in under a minute – some of the fastest nontrivial DNA computations to date [20]. For scaling up, binary ADDITION of pairs of 25-bit binary numbers was experimentally demonstrated (a 100-bit computation: 50 input bits, 25 carry bits, 25 output bits). However, for a very small fraction of inputs (around 2^{-50}) the system was stuck in an incorrect energy minimum. One problem identified was non-isoenergetic compute and scaffold domains (Figure 1a) conspiring to mess up the scaled up energy landscape in two ways: creation of local minima that mess up the kinetics, and even having erroneous structures unintentionally be the most energetically favoured.

1.1 Sculpting the energy landscape

Here we take the approach of using both program logic, and characterisation of system parameters, to sculpt the energy landscape to have both a high probability of the target output, as well as efficient, mostly downhill, pathways to that target.

For instance, the target output of the BITCOPY program (Figure 1a) on a length N scaffold, repeats the same compute domain (either bit 0 or 1) $N - 1$ times, but if the binding strength of 0 is slightly weaker than 1 by some ϵ , then the two target configurations for input 0 and 1 will differ by $\epsilon(N - 1)$, a large amplification. In that case, for some large enough N , an incorrect configuration will become the most favourable structure meaning that the energy

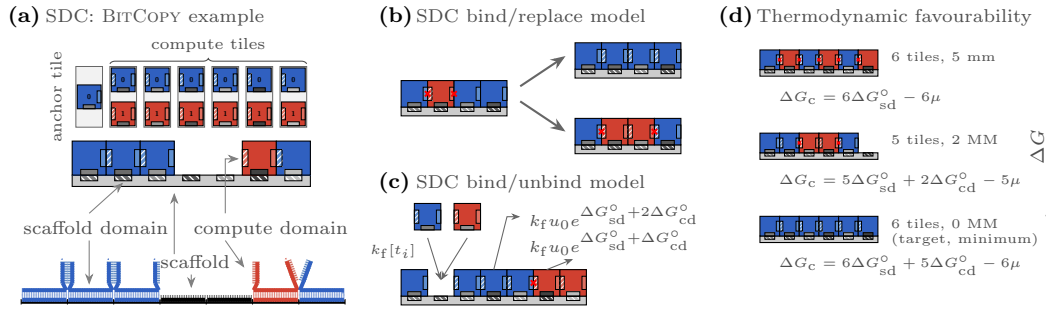


Figure 1 (a) In the Scaffolded DNA Computer (SDC) tiles bind to a 1D scaffold. In the BITCOPY program, an input bit is encoded in an anchor tile placed at the first scaffold position, then 0/1 tiles compete at each position to faithfully transmit the bit to the right. (b) The bind/replace SDC model considers configurations with a tile at every position; a tile is replaced if the replacing tile matches the same number of, or more, compute domains. (c) In the bind/unbind SDC model, tiles attach at concentration-dependent rates, and detach at energy-dependent rates. (d) The configuration energy depends on the number of tiles and bonds (or mismatches), with the full, no-mismatch target configuration having minimum energy configuration, assuming its bonds are sufficiently strong. See [20] for 10 more programs.

landscape has the “wrong” global minimum as well as being polluted by local minima that slow the system down.² The program ISOENERGETICBITCOPY [20] remedied the situation by reprogramming the energy landscape to be composed of a sequence of energy plateaus, one for each number of compute domain mismatches m , guaranteed to have both the correct global minimum and good kinetic pathways to get there. In Section 6 we prove that many SDC programs can be made isoenergetic in this way.

A second scaling problem comes from having non-isoenergetic scaffold domains: a strong scaffold domain can lock-in an incorrect tile during an anneal, or a weak scaffold domain can mean that two halves of the computation become isolated from each other, slowing the system’s natural error-correction processes. Since synthesis of custom-designed long DNA scaffolds is expensive and challenging, and tricks like DNA-base mismatches on biological (M13) scaffolds to even out domain energies have shown limited benefit [20], here, we instead turn to mathematics by proving an asymptotic result showing that good, polynomial time, kinetics can be maintained even in the face of poor scaffold energetics.

Finally, another scaling issue is whether for large enough scaffold length N there are accessible and wide enough regions of experimental assembly conditions (temperature, domain strength, concentrations, etc.) to allow for easy running of the system. We want to retain the in-principle advantage of just heating the system and cooling it down to find the intended correct output – precipitating a careful analysis of the system’s phase space.

1.2 Results and paper structure

In Section 2 we define two models: The SDC bind/replace model (Figure 1b) is intended to be simple to mathematically analyse, but ignores physically important properties like domain strength, tile concentration and temperature, key features of the SDC bind/unbind (Figure 1c) model. Sections 3–6 each give results concerned with scaling to larger scaffolds.

² Such as 011111: input is bit 0, the structure gains favourability by repeating lots of 1 bonds, and suffers only the cost of a single compute domain mismatch; see Figure 5 for more details.

In Section 3 we find that computational power of SDC systems is exactly characterised by the complexity class NL, or nondeterministic logspace [11] – assuming we allow polynomially many tile types per scaffold position and arbitrary amounts of nondeterminism. However, most SDC programs implemented to date, have a more restrictive left-to-right dynamics which turns out to characterise the complexity class L (deterministic logspace). (Previous work proved that SDCs simulate finite state machines [20], hence we generalise that result.)

Section 4 (Corollary 16) shows that the SDC bind/replace model has expected time for programs like BITCOPY of $\leq N^2$ (a small improvement on a previous $O(N^2)$ upperbound [20]), but mainly serves to set the scene for a more dramatic result in Section 6 (Theorem 29) where despite adversarially left-to-right increasing scaffold domain strengths on the scaffold we retain $O(N^2)$ completion time if the strength increase is a constant amount over the entire scaffold length, or polynomial completion time if it is $O(\log N)$ over the entire scaffold.

In Section 5, for the more physically accurate SDC bind/unbind model, this quadratic scaling result becomes $O(N^3)$ due to the wait time for tiles to fall off. This in turn comes from a positive scaling result (Corollary 18) showing that compute domain length of merely $O(\log N)$ DNA bases (or strength) is sufficient to have the correct target (output) with constant probability, greater than the total probability of exponentially many other structures. Thus, for a mere factor- N increase in running time, we gain high-probability computing. More generally, we give an exact expression for the probability of the target structure, over all filled structures, as a function of scaffold length and compute domain strength. We also find certain regimes (temperature, domain strength/length) with super-fast, mostly left-to-right, $O(N)$ -time kinetics. These theoretical results align with simulation, and in some cases experiment, despite us not yet fully understanding the energetics of our systems.

Finally, in Section 6.1 we prove a generalisation of the ISOENERGETICBITCOPY result described above: we show that a wide class of SDC programs can be simulated by systems that have almost perfectly isoenergetic configurations; in other words these simulator SDCs have an energy landscape that is composed of a sequence of plateaus that increase in height with the number of mismatches, and that lack kinetic traps. Traversing such a landscape amounts to downhill ratcheting steps, interspersed with flat plateau steps, giving fast dynamics.

1.3 Future work

We leave a number of open questions and directions for future work.

Many of our theoretical kinetics results are shown for BITCOPY, an SDC program that is straightforward to analyze while demonstrating much of the system’s dynamics. The analysis holds for similar programs with $O(1)$ tiles competing at each scaffold position, or where following the leftmost mismatch resolves errors. Can these results be generalized to programs that have a variable number of tiles per scaffold position, or where pushing the leftmost mismatch to the right results in unresolvable kinetic traps, or other complex pathways through the energy landscape?

In the SDC bind/replace model with scaffold length N , Corollary 16 gives a running time of $\leq N^2$ (for BITCOPY and similar programs). We conjecture that the running time from configurations chosen uniformly at random is $\Theta(N^2)$, a proof of which would show that mismatches cancelling each other out does not asymptotically speed up completion time.

Isoenergetic programs: In Section 6.1 we show that with a multiplicative scale factor equal to the number of distinct compute domains, we can compile any program into one with isoenergetic configurations that have a fixed number of mismatches. Can we reduce the scale factor? Can isoenergetic programs be achieved with scale factor 1? Can we further reduce energy differences between configurations with the same number of mismatches?

Temperature anneals: An advantage of thermodynamically-favoured DNA computing, and SDC in particular, is the ability to run the system via a simple thermal anneal: heat it up to melt everything, then cool down at some rate to compute. Previous experimental work showed 4-bit ADDITION in under a minute, but requiring several hours for 25-bit ADDITION [20]. What is the theoretically optimal annealing schedule for high-probability computing in terms of scaffold length (or number of input bits)?

Going beyond the SDC-specific work here, is there a more general theory of thermodynamically favoured computation, that embodies both equilibrium and kinetic pathways through a state space defined by the Boltzmann distribution over valid molecular complexes?

2 Definitions

The *domains* are a finite set D , partitioned into scaffold domains D_{sd} and compute domains D_{cd} . A domain $d \in D$ has a unique complement $d^* \in D$ with $(d^*)^* = d$. Complements respect the partition: either $d, d^* \in D_{\text{sd}}$, or $d, d^* \in D_{\text{cd}}$.

► **Definition 1 (tile).** A *tile* is a triple $t = (l, p, r)$ where $p \in D_{\text{sd}}$ is a scaffold domain and $l, r \in D_{\text{cd}}$ are the left and right compute domains, respectively.

► **Definition 2 (SDC).** An *SDC* S is a sequence of N nonempty tilesets $S = (T_1, T_2, \dots, T_N)$ where for each i , all tiles in T_i have the same scaffold domain p_i . The SDC is said to have scaffold length N with scaffold domains $p_1^*, p_2^*, \dots, p_N^*$ at scaffold positions $1, 2, \dots, N$. If a tileset T_i has size 1, we say that T_i (or position i , or the tile in T_i) is an *anchor*.

Although the previous definition allows for scaffold domains to be repeated, in this paper, and previous work [20], all N scaffold domains are distinct, i.e. $p_i \neq p_j$ when $i \neq j$.

The *configuration space* of S is $S^* = (T_1 \cup \{\emptyset\}) \times (T_2 \cup \{\emptyset\}) \times \dots \times (T_N \cup \{\emptyset\})$. A *configuration* is an N -tuple $C \in S^*$, indicating a tile $C[i]$, or no tile ($C[i] = \emptyset$), at each position i . In C , if adjacent tiles $t_{i,j} = (l, p_i, r) \in T_i$ and $t_{i+1,j'} = (l', p_{i+1}, r') \in T_{i+1}$ have adjacent complementary compute domains (meaning $r = l'^*$), we say the tiles, and their compute domains, *match*. Otherwise, they *mismatch*. A *full configuration* has no empty positions, a *partial configuration* has at least one. Absent tiles are treated as having domains which are not complementary to anything.

One key example of an SDC is BITCOPY (Figure 1a), having compute domains 0 and 1, and their complements $0^*, 1^*$. The tileset at each position i has two tiles $T_i = \{(0^*, p_i, 0), (1^*, p_i, 1)\}$, except for the first, which is an anchor (one tile). The only full mismatch-free configuration has all tiles matching the anchor, meaning it acts as a wire.

► **Definition 3.** An *SDC model* is a tuple $\mathcal{S} = (S, \xrightarrow{1}, R)$, where S is an SDC, $\xrightarrow{1}$ is a relation on $S^* \times S^*$ defining *one-step reachability* and denoted as $C \xrightarrow{1} C'$, and $R : S^* \times S^* \rightarrow \mathbb{R}^{\geq 0}$ is a *transition rate matrix* where $R(C, C') > 0 \iff C \xrightarrow{1} C'$. These rates define a continuous-time Markov chain over S^* .

This definition is intentionally abstract; two concrete models will be instantiated in Definitions 5 and 7. In a model, C' is reachable from C if there is a sequence of steps $C \xrightarrow{1} \dots \xrightarrow{1} C'$. A *terminal configuration* is one to which no step applies. For computation, target (output) configuration(s) should be reachable from any configuration and terminal; a computation is a sequence of steps from an initial to a terminal configuration. An SDC model is *directed* if it has a unique terminal configuration,³ and *undirected* otherwise (it reaches multiple terminal

³ This notion of determinism (directness) comes from work on the abstract tile assembly model [19].

configurations, or none). The *expected time to completion* (respectively, expected steps) from configuration C of a directed SDC model is the expected time (steps) for the Markov chain to reach a terminal configuration from C . When $\xrightarrow{1}$ and R are clear from context, we use S and $\mathcal{S}$ interchangeably. $C \xrightarrow{i} C'$ means $C \xrightarrow{1} C'$ and C, C' differ by a single tile $C[i] \neq C'[i]$.

2.1 SDC bind/replace model

► **Definition 4** (SDC bind/replace: mismatch energy). The *mismatch energy* of a configuration C is $G_{\text{MM}}(C) = |\{i \mid 1 \leq i < N \text{ and } t_i, t_{i+1} \text{ mismatch}\}|$; that is, the number of compute domain mismatches in C .⁴

► **Definition 5** (SDC bind/replace model). Given an SDC S , the *SDC bind/replace* model $(S, \xrightarrow{1}, R)$ is defined by $C \xrightarrow{1} C'$ if C and C' differ in exactly one position i , $C'[i] \neq \emptyset$, and either $C[i] = \emptyset$ or $G_{\text{MM}}(C') \leq G_{\text{MM}}(C)$. That is, in one step, a tile may bind to an empty position, and a tile may replace another if the replacement binds with equal or fewer mismatches. The transition rates are given by $R(C, C') = 1$ if $C \xrightarrow{1} C'$ and 0 otherwise.

The SDC bind/replace model has a first step where tiles bind to the scaffold, nondeterministically chosen at each position, yielding a full configuration. We ignore partial configurations and instead consider arbitrary full configurations, as this first step is generally fast.

2.2 SDC bind/unbind model

The SDC bind/unbind model includes key physical/chemical details. Throughout the paper, T denotes temperature in Kelvin, and R is the gas constant – a molar equivalent of the Boltzmann constant, in energy/temperature/amount ($R = 1.987 \text{ cal K}^{-1} \text{ mol}^{-1}$). A tile t has a positive real-valued concentration denoted $[t]$, in units of amount/volume (Molar, or M).

► **Definition 6** (SDC bind/unbind: standard binding free energy). Each pair of domains has a *standard free energy of binding* $\Delta G^\circ : \mathbb{R}^+ \times D \times D \rightarrow \mathbb{R}$ mapping a temperature and domain pair to a real-valued free energy in energy/amount units (kcal/mol [15]), where more negative is more favourable. We assume $\Delta G^\circ(T, d, d') = 0$ if $d' \neq d^*$, unless otherwise stated. This energy is calculated with respect to reactants at a reference concentration $u_0 = 1 \text{ M}$; the reference is denoted by the $^\circ$ sign.⁵

► **Definition 7** (SDC bind/unbind model). Given an SDC S , the *SDC bind/unbind* model $(S, \xrightarrow{1}, R)$ is defined by $C \xrightarrow{1} C'$ if C and C' differ in exactly one position i , where exactly one of $C[i]$ and $C'[i]$ is $\emptyset$. The transition rates $R(C, C')$ for attachment (when $C[i] = \emptyset$) or detachment (when $C'[i] = \emptyset$) of a tile $t_i = (l_i, p_i, r_i)$ are given by

$$\blacksquare \quad r_{\text{att}} = k_f [t],$$

$$\blacksquare \quad r_{\text{det}} = k_f u_0 e^{(\Delta G^\circ(T, p_i, p_i^*) + \Delta G^\circ(T, r_{i-1}, l_i) + \Delta G^\circ(T, r_i, l_{i+1}))/RT}$$

where k_f is an assumed or experimentally-derived rate constant (positive, real-valued, 1/concentration/time, and consistently $10^6 \text{ M}^{-1} \text{ s}^{-1}$ in this work).

For notational convenience, we define the *chemical potential* of tile t as $\mu(t) = RT \ln([t]u_0^{-1})$, implying a binding rate of the form $r_{\text{att}} = k_f u_0 \exp(\mu(t)/RT)$.

⁴ This notion shows up in the Ising model, where a mismatch is called a domain. Here, the word domain has a completely different meaning coming from chemistry.

⁵ While a reference concentration is the convention in many works [15], another convention uses amount fractions rather than concentrations, and results in different standard free energy values; this convention is notably used by NUPACK [5].

Equilibrium and configuration energy

In the SDC bind/unbind model, rather than defining the free energy of configurations by fiat, we can derive it from the previous definitions (see Appendix A.1 for the proof):

► **Lemma 8.** *Given any configuration C in the SDC bind/unbind model with reaction rates from Definition 7, the equilibrium probability of C is $\text{prob}[C] = e^{-\Delta G_c(T,C)/RT} \cdot Z^{-1}$, where*

$$\Delta G_c(T, C) = \sum_{i \in \{1, \dots, N\}} [\Delta G^\circ(T, p_i, p_i^*) - \mu(t_i)] + \sum_{i \in \{1, \dots, N-1\}} \Delta G^\circ(T, r_i, l_{i+1}) \quad (1)$$

is called the configuration energy of C at temperature T , R is the gas constant, $t_i = (l_i, p_i, r_i)$ is the tile at position i , $[t_i]$ is its concentration (if no tile is present $\mu(t_i) = 0$), and $Z = \sum_{C' \in \mathcal{C}} e^{-\Delta G_c(T, C')/RT}$ is the partition function over the set of all configurations $\mathcal{C}$.

2.3 Programs, inputs and outputs

In computer science, a problem is an infinite set of input-output pairs. To solve problems with SDCs we need to encode inputs and outputs, and a notion of an infinite set of SDCs to handle an infinite problem; we take inspiration from uniform Boolean circuit families [1, 12].

► **Definition 9** (SDC program). Let Σ be a finite alphabet, and let $\mathcal{P}(\mathfrak{T})$ be the power set of all tiles $\mathfrak{T}$. An SDC program is a tuple $(f_{\text{tiles}}, f_{\text{in}}, f_{\text{out}})$, where the function $f_{\text{tiles}} : \mathbb{N} \rightarrow \mathcal{P}(\mathfrak{T}) \times \mathcal{P}(\mathfrak{T}) \times \dots \times \mathcal{P}(\mathfrak{T})$ on input $x \in \Sigma^*$ maps its length $|x|$ to an N -tuple of tilesets $f_{\text{tiles}}(|x|) = \tilde{T} = (\tilde{T}_1, \dots, \tilde{T}_N)$ called the *program*. The function f_{in} encodes an input word by removing some tiles from the program, thus defining the SDC instance $S \stackrel{\text{def}}{=} f_{\text{in}}(x, f_{\text{tiles}}(|x|)) = (T_1, \dots, T_N)$ where $T_i \subseteq \tilde{T}_i$. Finally, the function f_{out} maps the input length to a tuple of *interpretation functions*: $f_{\text{out}}(|x|) = (f_1, f_2, \dots, f_N)$, where $f_i : T_i \rightarrow \Sigma \cup \{\epsilon\}$ interprets the tile at index i in a full mismatch-free configuration C of S as a, possibly empty, symbol. C is interpreted as a whole by concatenating of these symbols.

An SDC's full mismatch-free configuration(s) are called *target* configuration(s). Note that this definition is chosen for simplicity: it requires the SDC to have at least one target configuration, and focuses on what these configuration(s) are, rather than how we get there kinetically. A language $L \subseteq \Sigma^*$ is deterministically (respectively, nondeterministically) SDC-decided if for all $x \in \Sigma^*$, $x \in L$ if and only if the (respectively, at least one) target configuration(s) of the SDC $S = f_{\text{in}}(x, f_{\text{tiles}}(|x|))$ are interpreted as 1 (the string in Σ^*) by $f_{\text{out}}(|x|)$. The functions $f_{\text{tiles}}, f_{\text{in}}, f_{\text{out}}$ should be “easily computable” otherwise the encoding/decoding scheme could cheat by doing all the computation; here, we assume they are DLOGTIME computable [1], an incredibly weak notion of computability commonly used in uniform circuit complexity [12].⁶

► **Definition 10.** An SDC bind/unbind S is (a) *perfectly compute-domain isoenergetic* if for all temperatures T and all compute domains $d, d' \in D_{\text{cd}}$ in S , $\Delta G^\circ(T, d, d^*) = \Delta G^\circ(T, d', d^*)$, and is (b) *perfectly scaffold-domain isoenergetic* if for all scaffold domains $d, d' \in D_{\text{sd}}$ in S , $\Delta G^\circ(T, d, d^*) = \Delta G^\circ(T, d', d^*)$, and (c) is *perfectly isoenergetic* if both are true.

⁶ This encoding assumes that the entire system is described as a binary string. For f_{tiles} there needs to be a DLOGTIME-computable predicate verifying $t \in \tilde{T}_i$, given input $|x| = n$, index i and tile t . We note that these inputs are describable in $O(\log n)$ bits. Likewise, for f_{in} , the predicate takes as input an index $0 \leq i < n$ and $O(1)$ bits in defined indices $x_i, \dots, x_{i+O(1)}$ (the extra $O(1)$ bits are to accommodate putting multiple input bits on a single tile). f_{out} acts on index i , a tile t and bit b . These predicates would be applied a polynomial number of times, in parallel, to verify the uniformity of the entire polynomial size SDC system.

3 Computational power of SDC

Previous work showed that SDC systems simulate finite state machines [20] and that MFE prediction is in logspace (L) [17] for $O(1)$ tiles per scaffold position. Here we show that, computationally, SDC systems are characterised exactly by the complexity class NL: they can solve any problem in that class and no more. The result requires both nondeterminism and a large number of tiles, polynomial in scaffold length N . We also get an L -completeness result for polynomially many tiles, but with a determinism restriction that seems syntactically strong, but many implemented programs satisfy it.

3.1 SDCs are simulated by logarithmic space machines

A Turing machine has a read-only input tape, write-once-only output tape and a read/write work tape. The class of languages decided by deterministic (resp., nondeterministic) Turing machines using workspace $O(\log n)$ on input length n is called deterministic logspace and denoted L, resp. nondeterministic logspace and denoted NL [11, 13].

► **Lemma 11.** *Given an SDC S , there is a nondeterministic logspace Turing machine M , whose accepting computations output the set of full mismatch-free configurations of S .*

Proof. M maintains a scaffold position counter $i \in \{1, \dots, N\}$, on its worktape. Encodings are DLOGTIME computable (Definition 9), hence computable in logspace. First, M places any tile from T_1 on its output tape, and stores the right-side compute domain of that tile on its worktape, which needs only $O(\log |x|)$ bits since there are at most polynomially many domains since DLOGTIME is a subset of P (see Footnote 6). At scaffold position $i \geq 2$, M nondeterministically chooses any tile $t_{i,j}$ from tileset T_i if $t_{i,j}$'s left compute domain matches the right compute domain of the guessed tile at position $i-1$. Then M increments i , iterating the procedure. If $i = N$, and a matching tile was placed at position N , M accepts. Else if M reaches some position $\leq N$ where there is no matching tile to place, it rejects.

Given any mismatch-free configuration C , M outputs C by choosing each of its tiles successively, only ever writing one domain on its worktape; no mismatches implies that each successive tile is always a valid choice. Conversely, by construction, M can never choose a mismatching tile, and so can never output a configuration with a mismatch. ◀

► **Lemma 12.** *Let $L \subseteq \{0, 1\}^*$, be decided by an SDC program (Definition 9). Then $L \in \text{NL}$.*

The proof is in Appendix A.2.

3.2 Logarithmic space machines are simulated by SDCs

► **Lemma 13.** *Let $L \in \text{NL}$. Then there is an SDC program that decides L .*

Proof. We will define the SDC program by describing the behavior of f_{tiles} , f_{in} and f_{out} on an input $x \in \{0, 1\}^*$ to the nondeterministic logspace Turing machine M_L that decides L . Let $n = |x|$ and consider the configuration graph of M_L on inputs of length n (as a reminder [11, 13]: $G = (V, E)$ is a directed graph with one node c for every possible (including unreachable) configuration c of the *worktape* of M_L on inputs of length n , and given a particular input x , there is an edge from node c to c' iff M_L goes from configuration c to configuration c' in a single step when running on x [11]). G has at most $m = n^{O(1)}$ nodes, since that is the number of configurations of M_L on inputs of length n [11, 13].

The program is given by $f_{\text{tiles}}(n) = (V \times \{i\} \times V)^m$: that is, the compute domains of the SDC are nodes of M_L 's configuration graph, and the scaffold length is the number of nodes. On a specific input, $f_{\text{in}}(x)$ restricts these tilesets: for each $i \geq 2$, the set T_i has one

tile (c, i, c') for each directed edge (c, c') in G (which may depend on x). We let T_1 contain a single tile $(c, 1, c)$ where c is the configuration in which M_L is initialised on input x (noting that the left domain does not matter). Finally, f_{out} gives functions $f_i((c, i, c')) = \epsilon$ (the empty symbol) for $i < N$, and $f_N((c, N, c')) = 1$ if c' is an accepting configuration of M_L and ϵ otherwise. It is straightforward, albeit tedious, to verify these encodings are DLOGTIME computable, by iterating over all log-size pieces of the construction. $\blacktriangleleft$

3.3 An L characterisation

Many SDC programs implemented to date are *directed* (with exactly one target configuration), and have the property that their computation is not “too nondeterministic” (BITCOPY, ISOENERGETICBITCOPY, ADDITION, PARITY, MULTIPLYBY3, DIVBY2, COUNTER, RULE110 and BALANCEDBRACKETS [20]), which we define here and which limits their power to be within the class L.

► **Theorem 14.** *Consider the class of SDC programs P where each SDC instance S of P has an anchor in position 1 ($|T_1| = 1$) and is left-to-right deterministic: no tileset T in S has two tiles with the same left compute domain, i.e. given $t_i \in T_i$, there is at most one t_{i+1} in T_{i+1} that matches it. The languages decided by such programs are exactly L.*

Proof sketch. ($\subseteq$ L) We use the proof from Lemma 12, except that there is only one choice at each step when enumerating mismatch-free configurations, so nondeterminism is not needed.

($\supseteq$ L) Again, the exact proof of Lemma 13 works here: the configuration graph G of any deterministic logspace Turing machine M has outdegree ≤ 1 at each node, so no two tiles in any tileset T_i can have the same left compute domain. $\blacktriangleleft$

We note that the previous theorem relates to Theorem 3 from Shalaby et al [17], which showed that computing the MFE of an SDC system, in (essentially) our bind/unbind model, is in L for $O(1)$ size tilesets. Here we generalise to polynomial size tilesets, but in bind/replace.

4 SDC bind/replace: $O(N^2)$ kinetics via the leftmost mismatch

In this section, we analyze how long SDC programs might take to run under the simple SDC bind/replace model. We start with a lemma helping us generalize a result from [20].

► **Lemma 15.** *Let $S = (S, \xrightarrow{1}, R)$ be any SDC model where S is a BITCOPY SDC, $\xrightarrow{1}$ is the same as in bind/replace (see Definition 5), and R has the following properties:*

1. *Transition rates are local: If $C \xrightarrow{1}_i C'$, then $R(C, C')$ does not depend on anything outside the neighborhood $[i - 1, i + 1]$ in either C or C' .*
2. *More mismatches do not slow a transition: if $C_1 \xrightarrow{1}_i C'_1$ and $C_2 \xrightarrow{1}_i C'_2$ make the same replacement (i.e. $C_1[i] = C_2[i]$ and $C'_1[i] = C'_2[i]$), and $C_1[i]$ matches more of its neighbors than $C_2[i]$, then $R(C_1, C'_1) \leq R(C_2, C'_2)$.*

Then the configuration C_0 with a single mismatch in the leftmost position (i.e., a mismatch with the anchor) has the largest expected time to completion among all full configurations.

Proof sketch. Conceptually, BITCOPY cannot finish until its leftmost mismatch disappears. Putting that mismatch far away from the end of the scaffold maximizes the expected time to completion. Having any other mismatches cannot slow down the process because any mismatch that does not disappear before the leftmost mismatch disappears will instead cancel with the leftmost mismatch as it moves to the right, causing the position of the new leftmost mismatch to jump forward. We formalize this in Appendix A.3 using coupling. $\blacktriangleleft$

This framework demonstrates the advantage of the SDC bind/replace model: we need only think about one particular configuration to analyze its kinetics. Later, in Section 6, we will use this to analyze a more complex version of the model. Here, we characterize the simple version.

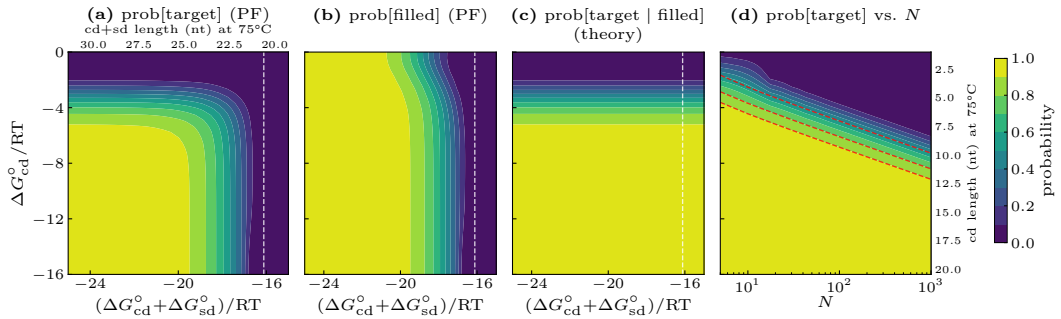
► **Corollary 16.** *In the SDC bind/replace model on a BITCOPY SDC with scaffold length N , the expected completion time from any configuration is less than N^2 .*

Proof. By Lemma 15, we need only consider the configuration with a single mismatch in the leftmost position. By Lemma 28 (a more general result, shown later), this configuration takes $(N - 1)^2$ expected steps, and therefore also expected time bounded by this (it has only 1 mismatch, each transition has rate 1, so each step takes expected time at most 1). ◀

5 The SDC bind/unbind model

5.1 Equilibrium in isoenergetic systems

Simple observation of the configuration energy shows that, in a perfectly isoenergetic (Definition 10) SDC in the SDC bind/unbind model, configurations will melt, that is, the empty configuration will be the most probable, at the critical point of $\mu \leq \Delta G_{sd}^\circ + \Delta G_{cd}^\circ$, when a tile attaching by one compute domain and one scaffold domain is unfavorable. In conditions where $N\Delta G_{sd}^\circ - N\mu + (N - 1)\Delta G_{cd}^\circ < 0$, the target configuration (full, mismatch-free) will be the minimum free energy configuration and thus the most probable. However, being the most probable amongst many configurations does not necessarily mean that the target configuration is probable overall. The overall equilibrium probability of the target configuration depends on the full partition function, which, in general, must be calculated algorithmically [17]. Here, we consider analytic approximations.



■ **Figure 2** Equilibrium probabilities for BITCOPY in the bind/unbind model. (a) Probability of the target configuration for $N = 20$, using the full partition function algorithm. Dashed line shows $\Delta G_{cd}^\circ + \Delta G_{sd}^\circ - \mu = 0$. (b) Probability that all scaffold sites are filled ($N = 20$, using full partition function). (c) $\text{prob}[\text{target} | \text{filled}]$ from Theorem 17. (d) $\text{prob}[\text{target}]$ vs. N at fixed $\Delta G_{sd}^\circ/RT = -19$; dashed red lines show constant $\text{prob}[\text{target} | \text{filled}]$ from Theorem 17.

► **Theorem 17.** *Let S be a bind/unbind BITCOPY SDC system of length N with all tiles at equal concentration $[t]$. Conditioned on all scaffold sites being filled, the equilibrium probability of the target configuration is $p = (1 + e^{\Delta G_{cd}^\circ/RT})^{-(N-1)}$.*

Proof. We take the approach of [20], but seek an exact result for any p . Recall that the first position in S is an anchor, with only one tile, and every other position has two, copying either 0 or 1. Each mismatch causes an increase in free energy of $-\Delta G_{cd}^\circ$. For this analysis,

we choose to set the free energy of the target state, with no mismatches, to be 0, then a state with m mismatches will differ in free energy from that target state by $-m\Delta G_{cd}^\circ$. As there are $(N-1)$ possible mismatches and therefore $\binom{N-1}{m}$ states with m mismatches, the partition function (i.e., the sum of all Boltzmann weights) will be $Z = \sum_{m=0}^{N-1} \binom{N-1}{m} e^{+m\Delta G_{cd}^\circ/RT}$.

In this regime, as the free energy of the target state is simply 0, the equilibrium probability of the target is $p = 1/Z$. The binomial theorem gives: $p = Z^{-1} = (1 + e^{\Delta G_{cd}^\circ/RT})^{-(N-1)}$. ◀

► **Corollary 18.** *Conditioned on all N scaffold positions being filled, the ΔG_{cd}° required to maintain a fixed target state probability p at equilibrium scales as $-O(\ln N)$.*

Proof. Rearranging, $\Delta G_{cd}^\circ/RT = \ln(p^{-1/(N-1)} - 1)$. Expand $p^{-1/(N-1)} \approx 1 - \ln p/(N-1)$ for large N , giving $\Delta G_{cd}^\circ/RT \approx \ln(-\ln p) - \ln(N-1)$ and noting p is constant wrt N . ◀

An implication of these results is that we may choose some desired $p > 0.5$ and the system and input will be easy to prepare experimentally, since there is no need to prepare particular configuration(s). Indeed components can be mixed haphazardly, heated up (to melt everything) and cooled down to give the target high enough probability.

5.2 Kinetics in isoenergetic systems

5.2.1 Simulations and the phases of growth behaviours

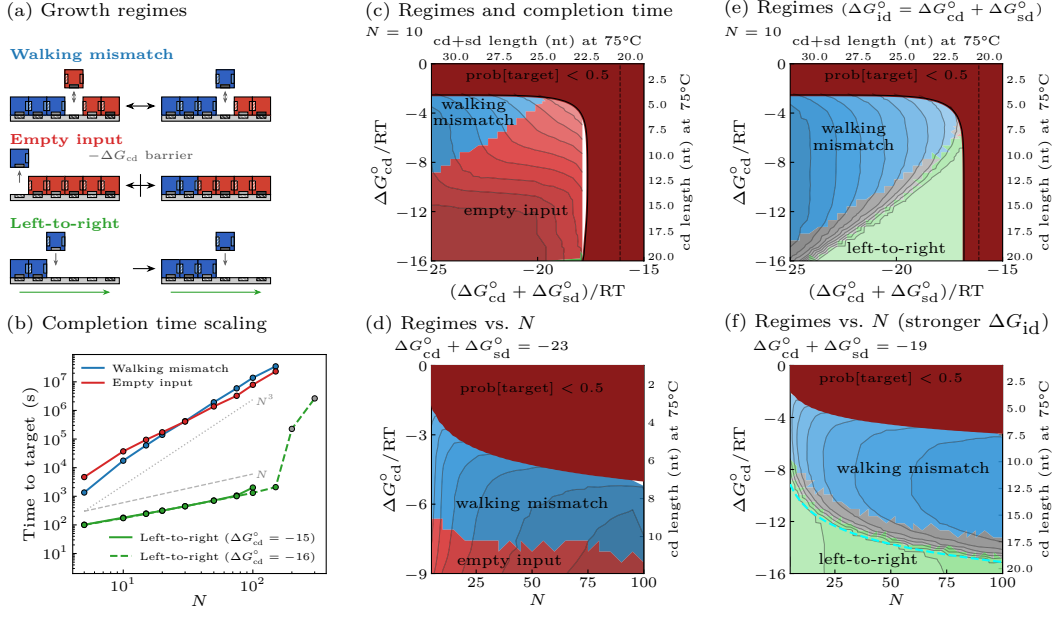
To understand SDC bind/unbind kinetics, we implemented the model in the Rgrow kinetic simulator [9]. We ran simulations of BITCOPY at fixed $[t] = 100$ nM, measuring “completion time” as the time taken to reach within 5% of the equilibrium target probability across many scaffolds. Simulations showed three distinct regimes, summarized in Figure 3b. At low N , with weak CDs and strong SDs, computation took place mainly via a “walking mismatch” phase: scaffolds quickly filled with potentially mismatching tiles, remained at high occupancy, and evolved primarily through single detachments followed by attachment, approximating the behavior of the SDC bind/replace model. We describe this phase in Section 5.2.2.

With stronger CDs and weaker SDs, however, behavior shifted: the number of mismatches per scaffold was often nearly zero even when the number of scaffolds in the target state was low; while mostly filled, many scaffolds became trapped for long periods of time in an all-incorrect state with an empty input site. We briefly discuss this phase in Remark 21.

The empty input behavior suggested a simple change: if instead of using a completely uniform SD strength, we set the SD at the left-most input location to $\Delta G_{id}^\circ = \Delta G_{sd}^\circ + \Delta G_{cd}^\circ$, then attachment of the input tile would be favorable even without a bond with its neighbor. With this change, for sufficiently strong ΔG_{cd}° and weak ΔG_{sd}° , the system exhibits a “left-to-right growth” phase: computation takes place on an initially empty scaffold, with the target state growing via a biased random walk of attachments and detachments at a growth front of a contiguous region of tiles starting at the left-most input. Completion times are much faster than other phases, and scale more slowly with N ; however, as N increases, the energy conditions required for the phase narrow. We investigate this phase in Section 5.2.3.

5.2.2 Walking-mismatch behaviour: $O(N^3)$ for fixed target probability

► **Theorem 19.** *In a perfectly isoenergetic (Definition 10) BITCOPY SDC bind/unbind system, in conditions where growth proceeds via a mostly-filled configuration where single detachments are quickly filled by attachment events, as ΔG_{cd}° increases in strength, the system behavior will approximate the SDC bind/replace model, with the replacement of a tile attached by one compute domain bond taking an expected time $k_f^{-1}u_0^{-1}e^{-(\Delta G_{sd}^\circ + \Delta G_{cd}^\circ)/RT} + (2k_f[t])^{-1}$.*

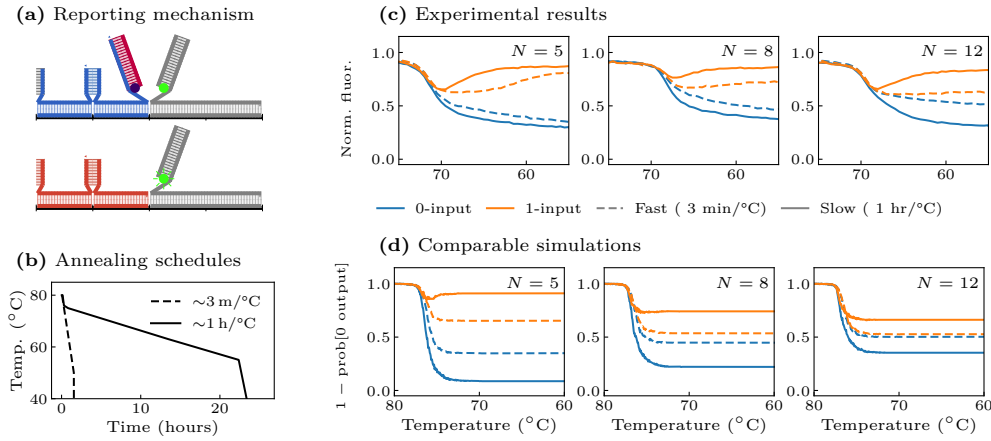


■ **Figure 3** Simulations and theory for kinetic phases of the bind/unbind model. (a) Three phases of behavior with $\text{prob}[\text{target}] > 0.5$. (b) For fixed values of ΔG_{sd}^o , and ΔG_{cd}^o set to obtain a particular target probability, scaling of completion time with N (time to 5% less than equilibrium target probability), for simulations of BITCOPY. (c)–(d) Observed behavior and completion time in simulations. Contour lines represent 5-fold changes in completion time. (e)–(f) Observed behavior and completion time in simulations when the input SD strength is increased to $\Delta G_{id}^o = \Delta G_{sd}^o + \Delta G_{cd}^o$. Grey regions represent ambiguous/mixed behavior; dashed blue line is $p_{\text{tr}} = 0.95$ from Theorem 22.

Proof. Consider a single site on a full scaffold, where the adjoining sites are both filled. The expected time for the tile to detach will be $k_f^{-1} u_0^{-1} e^{-(\Delta G_{sd}^o + b\Delta G_{cd}^o)/RT}$, where b is the number of compute domain bonds the tile makes with adjoining tiles. Once detached, another tile will attach with expected time $(2k_f[t])^{-1}$ (as there are two possible attachments). If we assume ΔG_{cd}^o is strong (very negative), then we may assume that a tile attached by two CD bonds will detach at a negligible rate, and a tile attached by zero will detach immediately. Thus, a tile making two bonds may not be replaced by a detach-attach sequence, a tile attached by zero bonds may be replaced by any other tile, and a tile attached by one bond may be replaced by a tile making one or two bonds. ◀

► **Corollary 20.** *In an isoenergetic (Definition 10) BITCOPY SDC bind/unbind system, in conditions where a mostly-filled configuration evolves through single detachments that are quickly filled by attachment events, for a fixed probability of the target state at equilibrium and fixed ΔG_{sd}^o , with ΔG_{cd}^o set to obtain the target probability, the time for a scaffold to reach the target state scales as $O(N^3)$.*

Proof. In SDC bind/replace, the expected time for a scaffold to reach the target state is $O(N^2)$ (Corollary 16). However, to maintain a fixed equilibrium target state probability, $\Delta G_{cd}^o/RT \leq \ln(-\ln p) - \ln(N-1)$ (Corollary 18); this means that the expected time per replacement event scales as $O(N)$, and thus the overall expected time for a scaffold to reach the target state scales as $O(N^3)$. ◀



■ **Figure 4** BITCOPY simulations and experiments for the bind/unbind model. (a) Experiments use a fluorophore/quencher reporter at the last position, with a fluorophore on one of the output tile types. 0-bit output results in a decrease in bulk fluorescence, while 1-bit output (or an empty output) remains fluorescent. (b) Two annealing protocols used for experiments. (c) Experimental bulk fluorescence during anneals at two rates, for scaffold lengths $N = 5, 8, 12$, of ISOENERGETICBITCOPY. Unpublished data courtesy of authors of [20], using extended-length reporting domains along with protocols in [20]. (d) Corresponding rgrow kinetic simulations with averaged, uniform domain energies, using nearest-neighbor parameters from [15].

► **Remark 21.** With uniform ΔG_{sd}^o , a region of incorrect tiles next to an empty left-most position allows a kinetic trap that cannot arise elsewhere on the scaffold, as illustrated in Figure 3a. The input tile has only one compute domain, and thus, in this case, will attach by only ΔG_{sd}^o , unfavorably. At this point, in order to form a bond with a matching tile in the next location, the incorrect tile, attached by $\Delta G_{sd}^o + \Delta G_{cd}^o$, would need to detach, also an unfavorable step. This results in a two-step kinetic barrier of $\Delta G_{sd}^o - \mu$ for the first configuration and $-\Delta G_{cd}^o$ for the second. However, it becomes less relevant with increasing N since it occurs only in one scaffold location.

5.2.3 Left-to-right growth: $O(N)$ in shrinking window of conditions

Here we consider *left-to-right growth*, that starts at an anchor tile on the left, grows to the right through a (possibly biased) random walk, but finishes on the right hand side of the scaffold before other tiles can nucleate to the right of the growth front. It is essentially a race of intended growth, against adversarial nucleation. The proof is in Appendix A.4.

► **Theorem 22.** In a BITCOPY SDC system in the bind/unbind model with input scaffold domain strength ΔG_{id}^o such that $\Delta G_{id}^o - \mu < 0$ and scaffold domain strength $\Delta G_{sd}^o - \mu \gg 0$, a scaffold will grow via a biased random walk from left to right, with mean growth rate $r_g = k_f([t] - u_0 e^{(\Delta G_{sd}^o + \Delta G_{cd}^o)/RT})$, with probability p_{ltr} such that $\ln p_{ltr} \approx -\frac{1}{2} k_f r_g^{-1} N^2 [t]^2 u_0^{-1} e^{-\Delta G_{sd}^o/RT}$.

► **Corollary 23.** In the left-to-right regime, the mean time for a scaffold to complete is $N r_g^{-1}$.

5.2.4 SDC bind/unbind simulations and experimental results

While we have primarily considered the evolution of systems at constant binding energy conditions, experimental implementations of SDC used temperature ramps of varying speed from a high, above-melting temperature to a low temperature where no further changes took

place [20]. Taking ΔG° to be of the form $\Delta H^\circ - T\Delta S^\circ$, where both terms are temperature-independent, and, while sequence-dependent, strengthen roughly linearly with domain length, a decrease in temperature corresponds with a decrease (strengthening) of both $\Delta G_{\text{sd}}^\circ$ and $\Delta G_{\text{cd}}^\circ$. Below the melting temperature, systems transition through increasing $\text{prob}[\text{target}]$, but decreasing detachment rates, and thus increasing expected completion time (experiments in left-to-right conditions have not yet been attempted). Computation takes place in the temperature window where both $\text{prob}[\text{target}]$ and detachment rates are sufficiently high; below this window, on experiment time scales, scaffolds remain “frozen”. Optimal temperature schedules to move close to equilibrium as quickly as possible remain an open question.

Experimentally, as shown in Figure 4a, observation of system state uses bulk fluorescence measurements, with the last (“output”) tile of one type (0 for BITCOPY) labelled with a long domain complementary to a quencher-modified strand, and an additional tile, binding to a corresponding fluorophore-modified strand and with an extended scaffold domain, attached to the right of the output position. Bulk fluorescence, measured on a qPCR machine, decreases when a high percentage of scaffolds have a 0 output tile attached, and remains high if there are a high percentage of scaffolds where the position is empty or has a 1 output tile attached.

Experiments and simulations in Figure 4c-d, show qualitative similarities. In both, the probability of a 1/0 output tile evolves depending on input only within a certain temperature range. An initial fluorescence decrease with input 1 is seen in some slow anneals in experiments and simulations, and as N increases, the annealing time needed for high yield increases.

However, experiments show changing fluorescence over a wider temperature range, a lower melting temperature, and a variance in the temperature of fluorescence changes depending on specific output position. Unlike simulations, where scaffold domains are assumed to be uniform strength, experimental implementations have so far used M13-based scaffolds with widely varying scaffold domain energies [20]; the energetic effects of junctions between bound compute and scaffold domains have also not yet been modelled.

6 Resilience to variable domain strengths

When designing DNA computers, there can be significant inaccuracy or non-ideality in the binding energies of strands from many sources: inaccuracies in our models, imperfections in DNA sequence design, or even using non-designed biological sequences for cost reasons. Here, we study how the SDC model performs when accounting for these discrepancies in scaffold or compute domain strength, particularly when scaling to large scaffold size N .

6.1 Addressing variable compute domain energies by programming an isoenergetic energy landscape

When scaling to larger scaffold length N , non-isoenergetic compute domains are a particularly prominent source of undesirable behaviour, due to DNA sequence design inaccuracies/non-idealities. The fundamental problem is that compute domains typically [20] appear multiple times along the scaffold. Hence, a small difference in compute domain strength may accumulate up to $N - 1$ times, and cause unintended structures with mismatches to become more thermodynamically favourable than intended target structures, as shown in Figure 5.

One approach would be good DNA sequence design: to scale up we would design more isoenergetic compute domain DNA sequences. The number of sequences increases exponentially with length and by Corollary 18 we expect compute domains to be of length $O(\log N)$ anyway, so the approach seems reasonable at first. However, redesigning sequences is not always desirable: perhaps the lab has a known “good” set of compute domains and

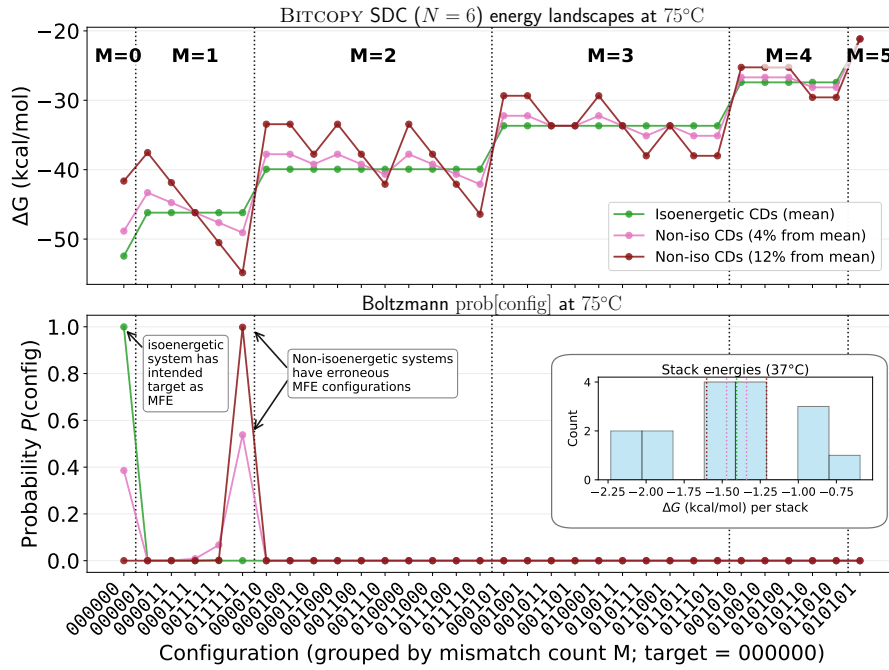


Figure 5 Energy landscapes of BITCOPY SDC systems, with input bit 0 and scaffold length $N = 6$, that are identical except for differences in compute domain energies. Each bit represents a tile, the correct target is 000000. Inset histogram: Distribution of nearest-neighbour stack energies at 37°C, for the 16 stacks with 4 DNA bases [15]. Free energy $\Delta G_c(75^\circ\text{C}, C)$ (top panel) and Boltzmann probability $\text{prob}[C]$ (bottom) at equilibrium, for isoenergetic compute domains (green) and non-isoenergetic domains at 4% and 12% from the mean stack energy (pink and dark red; in each case 0-0 binding domains are weaker (more positive) than 1-1). Arrows mark probability maxima on non-iso curves (erroneous MFE configurations). Compute domains are of length 12, and scaffold domains are isoenergetic and of length 24 bases, as used in previous experimental work [20].

we'd like to repurpose them for a larger system. Also, inaccuracies in the nearest neighbour model of DNA binding and in our ability to assay structures' energies mean that perfectly isoenergetic design is a huge challenge.

In this section, we take a more systematic approach. We show it is possible to sculpt SDC energy landscapes using the logic of SDC tilesets (programs). In previous work [20], a program called ISOENERGETICBITCOPY was experimentally implemented on a length $N = 25$ scaffold. The program is designed so that both of its two target configurations are approximately isoenergetic, as are configurations with the same number of compute domain mismatches, which in turn makes the energy landscape close to the ideal (green plateaus in Figure 5). ISOENERGETICBITCOPY outperformed another non-isoenergetic program, ADDITION, for $N = 25$. That work left open the interesting problem of whether ADDITION, or other SDC programs that repeat the same tile logic at each scaffold position, could be made to be isoenergetic. Theorem 25 gives a positive answer.

An implication of an SDC being perfectly compute-domain isoenergetic (Definition 10) is that for all m , $1 \leq m \leq N - 1$, there is a constant $c_m \in \mathbb{R}$ such that all full configurations C with m mismatches have identical free energy $c_m = \Delta G_c(T, C)$, assuming fixed temperature T and that mismatches have the same, e.g. 0, free energy. Here, we loosen things a little to state our main theorem of this section, which is proved in Appendix A.5.

► **Definition 24.** For bind/unbind SDC S , let $\epsilon = \max_{d,d' \in D_{cd}} \Delta G_{cd}^\circ(T, d, d^*) - \Delta G_{cd}^\circ(T, d', d'^*)$ be the largest difference between compute domain binding energies. S is *almost compute-domain isoenergetic* if there is a constant c independent of N such that all full configurations with m mismatches have free energy within additive $c(m+1)\epsilon$ of each other.

► **Theorem 25.** Given an SDC bind/unbind system S with scaffold length N , let D_R be the set of distinct compute domains appearing on the right sides of tiles in S , i.e., $D_R = \{r \mid \exists l, i : (l, i, r) \in T_i\}$. Assume that this value is the same for the left sides of tiles, i.e. that all compute domains appearing on tiles are capable of binding. Then there is an almost compute-domain isoenergetic SDC $\bar{S}$ that simulates S with scale factor $k = |D_R|$ (Definitions 24 and 30).

6.2 Kinetics under adversarial scaffold domain strengths

Here, we generalise the bind/replace model, and prove formal guarantees about the scaling of SDC kinetics under varying scaffold domain strengths. Proofs are in Appendix A.6.

► **Definition 26** (energetic SDC bind/replace model). Given an SDC S , the *energetic SDC bind/replace* model $(S, \xrightarrow{1}, R)$ is defined with the same reachability $\xrightarrow{1}$ as bind/replace (Definition 5). However, the replacement rate $R(C, C')$ for $C \xrightarrow{1} C'$ where $C'[i]$ replaces $C[i] = (l_i, p_i, r_i) \neq \emptyset$ is similar to the detachment rate from the bind/unbind model: $R(C, C') = \frac{k_{fuo}}{t'} e^{(\Delta G^\circ(T, p_i, p_i^*) + \Delta G^\circ(T, r_{i-1}, l_i) + \Delta G^\circ(T, r_i, l_{i+1}))/RT}$, where t' is the number of tiles that could have replaced $C[i]$, and other parameters are as in Definition 7. When $C[i] = \emptyset$ (for binding), we maintain the bind/replace convention that $R(C, C') = 1$.

This model captures some of the nuance of having binding domains with differing energies. SDC programs typically do computation from left to right, so it is desirable for good kinetics to have stronger scaffold domains closer to the left. Here we study what happens in the opposite, somewhat adversarial, regime, where scaffold domains become stronger to the right. We first show how to translate physical domain energies into mathematical probabilities.

► **Lemma 27.** In the biased SDC bind/replace model, for a configuration with a single mismatch between tiles t_i and t_{i+1} that can be replaced, the probability that the next replacement step replaces t_{i+1} is given by $(1 + e^{\Delta G_l - \Delta G_r})^{-1}$, where $\Delta G_l = (\Delta G^\circ(T, p_i, p_i^*) + \Delta G^\circ(T, r_{i-1}, l_i))/RT$ and $\Delta G_r = (\Delta G^\circ(T, p_{i+1}, p_{i+1}^*) + \Delta G^\circ(T, r_{i+1}, l_{i+2}))/RT$ give the total binding strength (one scaffold domain and possibly one compute domain) of t_i and t_{i+1} .

By focusing on the probability of moving forward in a given step, we can analyze simpler, more purely mathematical processes, and use these results to reason about SDC runtimes.

► **Lemma 28.** Consider the following biased random walk (representing the movement of the last mismatch in BITCOPY): There are $n = N - 1$ possible locations the mismatch can occupy, numbered from 0 to $n - 1$. It starts in the leftmost position (position 0). Each step, the mismatch moves to an adjacent position. If the mismatch is in position 0, it always moves right. Otherwise, it moves right with probability p and left with probability $1 - p$, until it moves right from position $n - 1$, ending the process at position n (representing the mismatch moving off the end of the scaffold). The expected number of steps s that this process takes is given by $s = n^2$ if $p = 0.5$ and $s = \frac{2p(1-p)}{(2p-1)^2} \cdot ((p^{-1} - 1)^n - 1) + \frac{n}{2p-1}$ otherwise.

The following theorem tells us that the system retains its speed even in the face of domain energies that increase along the entire scaffold (leading to backward bias). Even if this total energy differential increases logarithmically, the kinetics still operate in polynomial time. Corollary 18 shows that we only need domains of length $O(\ln N)$ for good thermodynamic properties, and $O(\ln N)$ -length domains cannot differ in energy by more than $O(\ln N)$.

► **Theorem 29.** Fix some temperature T , and let $S = (S, \xrightarrow{1}, R)$ be a biased SDC bind/replace model where S is a BITCOPY SDC of length N . Let compute domains have the same free energy (i.e., $\Delta G_{cd}^\circ(T, 0, 0^*) = \Delta G_{cd}^\circ(T, 1, 1^*)$), and scaffold domains become stronger linearly from left to right: that is, if p_j is the scaffold domain in position j , then $\Delta G_{sd}^\circ(T, p_j, p_j^*) = a - bj$ for some constants a, b with $b > 0$. Then the expected time to completion for S from any full configuration is $O(N^2)$ if $b \in O(\frac{1}{N})$, and is polynomial in N if $b \in O(\frac{\ln N}{N})$.

References

- 1 David A Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC^1 . *Journal of Computer and System Sciences*, 41(3):274–306, 1990. doi:10.1016/0022-0000(90)90022-d.
- 2 Charles H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, 1973. doi:10.1147/rd.176.0525.
- 3 Charles H. Bennett. The thermodynamics of computation—a review. *International Journal of Theoretical Physics*, 21(12):905–940, 1982. doi:10.1007/bf02084158.
- 4 Frits Dannenberg, Katherine E Dunn, Jonathan Bath, Marta Kwiatkowska, Andrew J Turberfield, and Thomas E Ouldridge. Modelling DNA origami self-assembly at the domain level. *The Journal of chemical physics*, 143(16), 2015. doi:10.1063/1.4933426.
- 5 Robert M Dirks, Justin S Bois, Joseph M Schaeffer, Erik Winfree, and Niles A Pierce. Thermodynamic analysis of interacting nucleic acid strands. *SIAM Review*, 49(1):65–88, 2007. doi:10.1137/060651100.
- 6 David Doty, Niels Kornerup, Austin Luchsinger, Leo Orshansky, David Soloveichik, and Damien Woods. Harvesting Brownian motion: Zero energy computational sampling, 2023. Arxiv preprint: arXiv:2309.06957. doi:10.48550/arXiv.2309.06957.
- 7 David Doty, Trent A Rogers, David Soloveichik, Chris Thachuk, and Damien Woods. Thermodynamic binding networks. In *DNA23: The 23rd International Conference on DNA Computing and Molecular Programming*, volume 10467 of *LNCS*, pages 249–266. Springer, 2017. doi:10.1007/978-3-319-66799-7_16.
- 8 Katherine E Dunn, Frits Dannenberg, Thomas E Ouldridge, Marta Kwiatkowska, Andrew J Turberfield, and Jonathan Bath. Guiding the folding pathway of DNA origami. *Nature*, 525(7567):82–86, 2015. doi:10.1038/nature14860.
- 9 Constantine Evans and Angel Cervera Roldan. Rgrow tile self-assembly simulator, December 2025. doi:10.5281/zenodo.15270539.
- 10 Jonathan Lee Tin Wah, Christophe David, Sergii Rudiuk, Damien Baigl, and André Estevez-Torres. Observing and controlling the folding pathway of DNA origami at the nanoscale. *ACS nano*, 10(2):1978–1987, 2016. doi:10.1021/acsnano.5b05972.
- 11 Cristopher Moore and Stephan Mertens. *The Nature of Computation*. Oxford University Press, 2011. doi:10.1093/acprof:oso/9780199233212.001.0001.
- 12 Niall Murphy and Damien Woods. AND and/or OR: Uniform polynomial-size circuits. In *MCU 2013: Machines, Computations and Universality. Electronic Proceedings in Theoretical Computer Science (EPTCS)*, volume 128, pages 150–166, 2013. doi:10.4204/EPTCS.128.20.
- 13 Christos H Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.
- 14 Paul WK Rothmund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297–302, 2006. doi:10.1038/nature04586.
- 15 John SantaLucia Jr and Donald Hicks. The thermodynamics of DNA structural motifs. *Annu. Rev. Biophys. Biomol. Struct.*, 33:415–440, 2004. doi:10.1146/annurev.biophys.32.110601.141800.
- 16 Fabian Schneider, Natalie Möritz, and Hendrik Dietz. The sequence of events during folding of a DNA origami. *Science advances*, 5(5):eaaw1412, 2019. doi:10.1126/sciadv.aaw1412.

- 17 Ahmed Shalaby, Chris Thachuk, and Damien Woods. Minimum free energy, partition function and kinetics simulation algorithms for a multistranded scaffolded dna computer. In *DNA 29: Proceedings of the 29th International Meeting on DNA Computing and Molecular Programming*, volume 276 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:22, 2023. doi:10.4230/LIPIcs.DNA.29.1.
- 18 Jean-Philippe J Sobczak, Thomas G Martin, Thomas Gerling, and Hendrik Dietz. Rapid folding of DNA into nanoscale shapes at constant temperature. *Science*, 338(6113):1458–1461, 2012. doi:10.1126/science.1229919.
- 19 David Soloveichik and Erik Winfree. Complexity of self-assembled shapes. *SIAM Journal on Computing*, 36(6):1544–1569, 2007. doi:10.1137/s0097539704446712.
- 20 Tristan Stérin, Abeer Eshra, Janet Adio, Constantine Evans, and Damien Woods. A thermodynamically favoured molecular computer: Robust, fast, renewable, scalable, 2025. bioRxiv preprint: bioRxiv doi.org/10.1101/2025.07.16.664196.
- 21 Boya Wang, Cameron Chalk, David Doty, and David Soloveichik. Molecular computation at equilibrium via programmable entropy. *Science Advances*, 12(2):eadx3969, 2026. doi:10.1126/sciadv.adx3969.
- 22 Erik Winfree. Simulations of computing by self-assembly. Technical Report CS-TR:1998.22, Caltech, May 1998. URL: <https://www.dna.caltech.edu/Papers/simulation.pdf>.

A **Appendix**

A.1 Proof of Lemma 8

Proof of Lemma 8. In an SDC bind/unbind system, every tile attachment event $C \xrightarrow{1} C'$ of tile t_i has a corresponding reverse (detachment) event $C' \xrightarrow{1} C$ of tile t_i , and there are no other events in the system. The system is in chemical equilibrium, that is, the concentration of every configuration is constant over time, when every such bind/unbind pair of events has equal forward and reverse rates, i.e. $\text{prob}[C]r_{\text{att}} = \text{prob}[C']r_{\text{det}}$ for such a triple t_i, C, C' :

$$\text{prob}[C'] = \text{prob}[C]e^{-(\Delta G^\circ(T, s_i, s_i^*) - \mu(t_i) + \Delta G^\circ(T, r_{i-1}, l_i) + \Delta G^\circ(T, r_i, l_{i+1}))/RT}. \quad (2)$$

Furthermore, any configuration C is reachable from the empty configuration by $\leq N$ tile additions, at most one per scaffold position. Thus the equilibrium probability of C is the equilibrium probability of $C_\emptyset$ times the product of the Equation (2) weight from each step.

Applying Equation (2), starting with $C = C_\emptyset$ one tile bind step at a time, implies that the energy terms involved for each tile binding event are summed: Each additional tile binding contributes a free energy term of $\Delta G^\circ(T, s_i, s_i^*) - \mu(t_i)$, and each matching compute domain (binding) between adjacent tiles contributes a free energy term of $\Delta G^\circ(T, r_i, l_{i+1})$ (each bond contributes exactly once). These N binding event terms sum to give $\Delta G_c(T, C)$ in Equation (1). Thus, the equilibrium probability of an assembly C is $\text{prob}[C] = \text{prob}[C_\emptyset]e^{-\Delta G_c(T, C)/RT}$.

Since configuration probabilities should sum to 1, we set $\sum_{C \in \mathcal{C}} \text{prob}[C] = 1$, hence $\text{prob}[C_\emptyset] \sum_{C \in \mathcal{C}} e^{-\Delta G_c(T, C)/RT} = \text{prob}[C_\emptyset]Z = 1$, and $\text{prob}[C_\emptyset] = Z^{-1}$, yielding the claim: $\text{prob}[C] = e^{-\Delta G_c(T, C)/RT} \cdot Z^{-1}$. ◀

A.2 Proof of Lemma 12

Proof of Lemma 12. We describe a nondeterministic logspace machine M on input $x \in \{0, 1\}^*$, simulates the SDC to decide x . Given $x \in \{0, 1\}^*$, M first generates an SDC instance via the SDC program’s DLOGTIME-computable encoding functions. M does not explicitly write out the SDC as it does not have enough workspace, however using a well-known procedure [13] it can generate $O(\log |x|)$ -size parts of the SDC on the fly any

time they are needed. Since $N = |x|^{O(1)}$ by DLOGTIME-computable encodings, we get that $O(\log |x|) = O(\log N)$, and hence individual domains and tiles can be generated in this way, which is all that is required to run the construction in the proof of Lemma 11. Via Lemma 11, the nondeterministic logspace machine is capable of outputting full, mismatch-free output configurations. However, instead of outputting the full configuration it merely outputs the decoded output (via the uniformity condition), which in this case is a single bit: 1 if $x \in L$ and 0 otherwise. ◀

A.3 Proof of Lemma 15

Proof. Let C be any full configuration. Since mismatch energy never increases (by definition of $\xrightarrow{1}$), starting from C_0 guarantees exactly one mismatch until completion. We can couple the evolution of C and C_0 (using these names from here on to refer to these evolving configurations) as follows: Whenever the mismatch in C_0 is further left than the leftmost mismatch of C , we pause C and let C_0 run until they are in the same position (which must eventually happen). Whenever the two trajectories have their leftmost mismatch at the same index, we force the two trajectories to move this mismatch in tandem: specifically, suppose the mismatch is between tiles t_i and t_{i+1} . As soon as C replaces either of these tiles (switching its domains from 0 to 1 or 1 to 0), C_0 replaces the same tile. Because of the conditions placed on R , these replacement rates from C must be the same or higher than those of C_0 : the replacement of t_i has the same rate by the first condition, and the replacement of t_{i+1} has the same or a faster rate by both conditions (t_{i+1} and t_{i+2} may possibly mismatch in C , but they do not in C_0). This continues until no mismatches remain.

The resulting trajectory of C_0 must take time at least that of C , as it has all the same replacement steps (possibly some extra). From the perspective of C_0 , the trajectory was either sampled accurately or sped up: the mismatch may have moved more quickly than it would have and may have been more likely to move right, both of which result in a faster trajectory. It follows that the expected time from C_0 is at least as high as that from C . ◀

A.4 Proof of Theorem 22

Proof of Theorem 22. Whether a scaffold will grow by a left-to-right random walk depends on two competing processes: the random walk of attachments by and detachments at a growth front by $1\text{SD} + 1\text{SD}$, and the probability of a stable region of tiles “nucleating” at some other point before it is covered by the left-to-right growth.

Left-to-right growth, given the $\Delta G_{\text{id}}^\circ - \mu < 0$ assumption, will have no barrier to starting. It will then proceed via individual attachments at rate $k_f[t]$, and detachments at rate $k_{\text{fe}}(\Delta G_{\text{sd}}^\circ + \Delta G_{\text{cd}}^\circ)/RT$, resulting in a mean growth rate of $r_g = k_f([t] - u_0 e^{(\Delta G_{\text{sd}}^\circ + \Delta G_{\text{cd}}^\circ)/RT})$. Here, we ignore detachments at points other than the growth front, which would occur at the lower rate of $k_{\text{fe}}(\Delta G_{\text{sd}}^\circ + 2\Delta G_{\text{cd}}^\circ)/RT$, and would be very likely to be quickly filled in again.

Formation of stable regions at other points requires a nucleation process, with an initially unfavorable attachment. However, the critical nucleus, or least-favored state, is a single tile (attached by $\Delta G_{\text{sd}}^\circ$), and all subsequent attachments are favorable. Thus, nucleation can be treated as similar to the kinetic trapping model of errors in DNA tile self-assembly [22]: the rate of a tile initially attaching at a given site is $k_f[t]$, and the probability of an additional tile attaching on either side before the initial tile detaches is $2[t]/(2[t] + u_0 e^{\Delta G_{\text{sd}}^\circ/RT})$, as we are in a regime where $[t] \ll u_0 e^{\Delta G_{\text{sd}}^\circ/RT}$, this probability is approximately $2[t]u_0^{-1}e^{-\Delta G_{\text{sd}}^\circ/RT}$, and thus, the rate of nucleation at a site is $r_{\text{nuc}} \approx k_f 2[t]^2 u_0^{-1} e^{-\Delta G_{\text{sd}}^\circ/RT}$.

There are N sites in the system, and the left-to-right growth will complete in mean time Nr_g^{-1} , meaning that the total sites and time available for nucleation is $N^2/(2r_g)$. The probability that no nucleation takes place during the left-to-right growth is thus $p_{\text{ltr}} = \exp(-r_{\text{nuc}}N^2/(2r_g))$, and thus $\ln p_{\text{ltr}} \approx -\frac{1}{2}k_f r_g^{-1}N^2[t]^2 u_0^{-1} e^{-\Delta G_{\text{sd}}^{\circ}/RT}$. ◀

A.5 Definition 30 and Proof of Theorem 25

► **Definition 30.** Let S be an SDC with scaffold length N . Given another SDC $\bar{S}$ using the same compute domains, we say that $\bar{S}$ *simulates* S at scale factor $k \in \{1, 2, \dots\}$ under some reachability function $\xrightarrow{1}$ if it has scaffold length kN and every configuration C of S has a corresponding configuration $\bar{C}$ of $\bar{S}$ that represents it as follows: the tile $t = C[i]$ at position i in S is represented by the tile $\bar{C}[ki]$ at position ki in $\bar{S}$ via some representation function $f_i : \bar{T}_{ki} \cup \{\emptyset\} \rightarrow T_i \cup \{\emptyset\}$ (i.e., $f_i(\bar{C}[ki]) = C[i]$) where $f_i(\emptyset) = \emptyset$ (and no other tile maps to $\emptyset$). We require that $\bar{C}$ has a mismatch between $\bar{C}[ki]$ and $\bar{C}[ki+1]$ if and only if C has a mismatch between $C[i]$ and $C[i+1]$, with no other mismatches at any other positions in $\bar{C}$ except for those between two adjacent $\emptyset$ s. Further, we require that $C \xrightarrow{1} C'$ implies $\bar{C} \xrightarrow{k} \bar{C}'$.

Proof of Theorem 25. Each scaffold position $i \geq 1$ in S is simulated by a block of k scaffold positions $k(i-1)+1, \dots, ki$ in $\bar{S}$, where $k = |D_R|$. We call this set of indices “block i of $\bar{S}$ ”. Fix some cyclic permutation σ on D_R . There are two types of tilesets in each block of $\bar{S}$. The tileset at the last index ki of block i is given by $\bar{T}_{ki} = \{(\sigma^{-1}(l^*)^*, ki, r) \mid (l, i, r) \in T_i\}$. Each other tileset at *any* other position j in block i (i.e., with $k(i-1) < j < ki$) is given by $\bar{T}_j = \{(r^*, j, \sigma(r)) \mid r \in D_R\}$. The representation f_i , for all i , is bijective and maps the tiles based on the tileset construction – that is, $f_i((\sigma^{-1}(l), ki, r)) = (l, i, r)$. Given a full configuration C of S , the representative configuration $\bar{C}$ of $\bar{S}$ is the unique configuration of $\bar{C}$ such that $f_i(\bar{C}[ki]) = C[i]$ for each i , and no mismatches occur anywhere except possibly between tiles ki and $ki+1$ for any value(s) of i . This exists uniquely, as after fixing each tile at each index ki , there is always a unique choice of tile to place to the immediate left of each position ki because σ is a permutation. This gives a unique way from each position ki to place tiles leftward until reaching position $k(i-1)$. If C is a partial configuration (containing one or more $\emptyset$ s), $\bar{C}$ is constructed in the same way, except the tiles within the block whose rightmost tile maps to $\emptyset$ are *all* $\emptyset$.

In any configuration $\bar{C}$ representing a full configuration C , we observe that, starting from the left of each block and moving rightward, each tile has a distinct left domain, with each domain appearing exactly once. The leftmost tile t_j in any block is given by $(r^*, j, \sigma(r))$ for some $r \in D_R$. As there are no mismatches within any block, the next tile must be $(\sigma(r)^*, j+1, \sigma(\sigma(r)^*)^*) = (\sigma(r)^*, j+1, \sigma^2(r))$. By straightforward induction, the $(k-1)$ st tile has right glue $\sigma^{k-1}(r) = \sigma^{-1}(r)$, as σ is a permutation.

Any full configuration of $\bar{S}$ looks like this, so it follows that the sum of the left compute domain energies within each block are the same across all blocks. Therefore, all target configurations of $\bar{S}$ have almost the same free energy, differing only by which compute domain is the unbound leftmost compute domain of the leftmost block. With ϵ as in Definition 24, this implies that these configurations are all within ϵ configuration energy of each other, satisfying Definition 24 for $m = 0$ mismatches. For $m > 0$, each configuration contains at most m blocks that have at least one mismatch; all other blocks contribute the same amount of energy in each. These configurations therefore differ by at most km compute domain bonds, and therefore by total energy at most $km\epsilon$, satisfying Definition 24.

To show that Definition 30 is satisfied, all requirements are true by construction of $\bar{C}$; reachability is straightforward, as $C \xrightarrow{1} C'$ implies that $\bar{C}$ and $\bar{C}'$ differ by exactly k tiles, which are all empty in one and all full in the other, implying that k steps is sufficient to get from one to the other. ◀

We remark that this construction and argument work equally well (though more tediously, and with more complex definitions) by having each scaffold position i be represented by a block of size $|\{r|(l, i, r) \in T_i\}|$, that is, by using a cyclic permutation that cycles through only the right domains of tiles in T_i . This may lead to a much smaller scale factor if compute domains are not frequently reused.

A.6 Proofs for Section 6.2: Lemmas 27 and 28 and Theorem 29

Proof of Lemma 27. Given exponential random variables with rates λ_l and λ_r (describing total tile replacement rates), the probability that a given value sampled from the latter is smaller than a given value sampled from the former is well known to be given by $\frac{\lambda_r}{\lambda_l + \lambda_r}$. The total replacement rate for each tile has one contribution per tile that can replace it (given by Definition 26), and each contribution is divided by the total number of tiles. Therefore, we can substitute the replacement rates from Definition 26 for λ_r and λ_l , and after cancelling constants and removing the irrelevant contribution from the mismatch between t_i and t_{i+1} , it follows that the probability we seek is given by $e^{\Delta G_r} / (e^{\Delta G_l} + e^{\Delta G_r}) = 1 / (1 + e^{\Delta G_l - \Delta G_r})$. ◀

Proof of Lemma 28. Let $E(i)$ be the expected number of steps when the mismatch is at position i . We seek the value of $s = E(0)$. At each position, the mismatch moves one step and then occupies (and has the expected value of) a new position. Therefore, $E(i) = 1 + p \cdot E(i+1) + (1-p) \cdot E(i-1)$ for all $0 < i < n$ (where $E(n) = 0$, as the process is finished then).

Let $a_i = s - E(i)$. Then $a_0 = 0$, and $a_1 = a_0 + 1 = 1$ because, from the leftmost position, the mismatch can only move right. Further, as $E(n) = 0$, $a_n = s$, so we must find a_n . Rewriting and re-indexing the above equation in terms of a_i and rearranging it gives a recurrence relation for a_i : $a_i = p^{-1} + p^{-1}a_{i-1} + (1-p^{-1})a_{i-2}$. We solve this recurrence relation in a standard way: we first homogenize it and find that its characteristic polynomial is $f(\lambda) = \lambda^3 + (-1-p^{-1})\lambda^2 + (2p^{-1}-1)\lambda + (1-p^{-1})$, with roots 1, 1, and $p^{-1}-1$.

When $p = \frac{1}{2}$, the multiplicity 3 root at 1 implies that any solution has the form $a_n = c_0 + c_1n + c_2n^2$, and plugging in initial conditions $a_0 = 0$, $a_1 = 1$ and $a_2 = \frac{2}{p}$ gives $a_n = n^2$. When $p \neq \frac{1}{2}$, the roots of $f(\lambda)$ imply solutions have the form $a_n = c_0 + c_1n + c_2(p^{-1}-1)^n$. Plugging in initial conditions again yields the given solution. ◀

Proof of Theorem 29. As we wish for an upper bound, by Lemma 15, we need only consider the configuration with one mismatch in the leftmost position. Because compute domains are isoenergetic and scaffold domains get linearly stronger, Lemma 27 says that the probability of a mismatch stepping forward will be the same at each scaffold position, and is given by $p = \frac{1}{1+e^b}$. Therefore, since $b > 0$, $p \neq 0.5$ and we can plug it in to the expression in Lemma 28 to obtain an upper bound on the expected number of steps (which is exact in the case of one mismatch in the leftmost position). This is sufficient for an asymptotic bound on the expected time as well: when $b \in O(\frac{1}{N})$, the strongest domain has constant strength (so each step takes constant expected time), and when $b \in O(\frac{\ln N}{N})$ it has strength $O(\ln N)$ (so each step takes at most polynomial expected time). Note in this expression that $n = N-1$, as a scaffold of size N has this many potential mismatch positions. After plugging in p and simplifying, this expression becomes

$$\frac{1+e^b}{1-e^b}(N-1) + \frac{2e^b}{(1-e^b)^2} \cdot (e^{b(N-1)} - 1).$$

7:22 Scaling up Thermodynamically Favoured Scaffolded DNA Computing

As we're looking for an upper bound, we ignore the first (negative) term, and we may replace the denominator of the second term with b^2 because $e^b - 1 > b$ for $b > 0$. We may also remove the final -1 . This simplifies the upper bound further to $\frac{2e^{bN}}{b^2}$. When $b = c/N$ for some constant c , this becomes $\frac{2e^c}{c^2} N^2 \in O(N^2)$. When $b = \frac{c \ln N}{N}$, it becomes $\frac{2N^{c+2}}{\ln^2 N} \in O(N^{c+2})$. ◀

Amplification at Equilibrium: Structural and Thermodynamic Limitations, and Implementation

Hamidreza Akef 

Department of Electrical and Computer Engineering, The University of Texas at Austin, TX, USA

Chia-Yu Sung 

Department of Molecular Biosciences, The University of Texas at Austin, TX, USA

Aneesh Vanguri 

Department of Molecular Biosciences, The University of Texas at Austin, TX, USA

David Soloveichik 

Department of Electrical and Computer Engineering, The University of Texas at Austin, TX, USA

Abstract

Amplifying weak molecular signals is essential in both natural and engineered biochemical systems. While most amplification schemes operate out of equilibrium, relying on kinetic barriers and fuel-driven cascades, it is also possible to amplify at thermodynamic equilibrium by shifting the energy landscape upon addition of an analyte. Equilibrium amplification is appealing because, in principle, the system can remain indefinitely in the untriggered state. In this work, we establish fundamental structural and thermodynamic limits on equilibrium-based amplification. We first prove that dimerization networks – systems restricted to complexes of at most two monomers – are inherently incapable of equilibrium amplification. This no-go theorem explains the absence of amplification in prior undercomplementary “strand commutation” designs. We then show that allowing trimeric complexes breaks this barrier. We propose an isometric trimer-based equilibrium amplifier whose output preserves the size of the input, enabling modular composition, and validate it experimentally, achieving an amplification factor close to the expected $2\times$. Finally, we derive universal thermodynamic bounds applicable regardless of complex size: the maximum amplification factor scales linearly with the free energy of interaction between the analyte and the amplifier components. For nucleic acid systems, this implies that the analyte length must grow linearly with the desired amplification factor, and that composing modular amplifiers yields diminishing returns for a fixed analyte. Together, these results delineate the structural and energetic boundaries of equilibrium amplification and rigorously justify the necessity of out-of-equilibrium approaches for achieving high gain.

2012 ACM Subject Classification Applied computing → Systems biology; Computer systems organization → Molecular computing; Mathematics of computing → Mathematical analysis

Keywords and phrases Equilibrium amplification, Dimerization networks, Strand commutation, Thermodynamics of amplification

Digital Object Identifier 10.4230/LIPIcs.DNA.32.8

Related Version *Full Version:* <https://arxiv.org/abs/2604.04285> [1]

Funding Schmidt Sciences Polymath Award to D.S., DOE grant DE-SC0024248, NSF SemiSynBio III: GOALI grant 2227578.

Acknowledgements We thank Antoni Szegłowski for performing experiments and Marko Vasić for sequence design on an earlier variation of the trimeric equilibrium amplifier.



© Hamidreza Akef, Chia-Yu Sung, Aneesh Vanguri, and David Soloveichik;
licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 8; pp. 8:1–8:24

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

In both natural and engineered systems, amplifying weak molecular signals is crucial for reliable function. For example, signal transduction cascades employ multiple mechanisms – including second messengers, protein phosphorylation, and enzyme cascades – to convert low-level extracellular inputs into robust intracellular responses [25]. Similarly, regulatory RNAs act as ultrasensitive switches to tightly govern gene expression [11]. This broad capacity for signal amplification is essential for maintaining cellular specificity and responsiveness across diverse biological contexts, from hormone signaling to receptor-mediated pathways that drive cellular differentiation and metabolism [2, 13].

Parallel to these natural architectures, amplification in molecular diagnostics and synthetic biology enables the detection of scarce biomarkers and the construction of complex biochemical circuits. Enzyme-free DNA circuits have demonstrated highly sophisticated functions: Seelig *et al.* implemented modular logic gates with signal restoration and amplification [23], and Qian and Winfree scaled strand-displacement cascades into multi-layer logic devices [22]. Likewise, dynamic amplification schemes such as the hybridization chain reaction and catalytic hairpin assembly rely on cascading strand displacements to robustly boost signal levels [12].

Chemical amplifiers typically rely on bridging an energy barrier in the presence of the analyte and thus necessarily operate out of equilibrium. For example, most DNA-based amplifiers rely on fuel-driven kinetic cascades. In the hybridization chain reaction, two DNA hairpins remain kinetically trapped until an initiator strand opens one, setting off a chain of strand-displacement reactions [5]. Likewise, catalytic hairpin assembly uses a series of metastable hairpin fuels: analyte binding to one hairpin exposes a toehold that catalytically opens the next, releasing the analyte to repeat the cycle and yielding hundreds-fold signal amplification [10].

Nonetheless, it is possible to perform signal amplification at thermodynamic equilibrium. In such systems, prior to the addition of the analyte, the thermodynamic equilibrium has a relatively small amount of output (detection species). The addition of the analyte shifts the energy landscape, and subsequent relaxation to the new equilibrium increases the output concentration by more than the amount of analyte added. An important potential advantage of equilibrium amplifiers is that they do not rely on a kinetic barrier, and thus can indefinitely remain in the untriggered state. Equilibrium amplifiers also necessarily avoid complex enzymes typical of natural signaling cascades since the only role an enzyme can play is to modify a kinetic barrier. Equilibrium amplifiers also support dynamic signal processing: changing the input (e.g., swapping input strands) causes the system to re-equilibrate accordingly [24], whereas kinetically controlled systems are generally single-use, requiring their metastable complexes to be externally regenerated for each new input. A family of equilibrium amplifiers was recently proposed in the Thermodynamic Binding Network (TBN) model [6, 20].

Equilibrium-based chemical computation recently received a significant boost with the work of Nikitin on “strand commutation” networks, who showed that diverse computations can be performed by weak hybridization of DNA strands [15]. Specifically, the inputs are DNA strands and the computation arises via the competition of multiple possible binding partners between pairs of strands. The thermodynamic equilibrium of the input together with the rest of the system then contains the desired amount of output strands. Undercomplementarity appears to allow significantly more compact constructions. For example, Nikitin contrasts prior work on the square root circuit using 130 strands [22] with his undercomplementary implementation using only nine strands (plus four inputs). Nonetheless, the computations Nikitin demonstrated suffer from significant signal attenuation, especially in the more complex constructions, which exhibit output concentrations many orders of magnitude lower than

the input concentration (e.g. analog computation). In none of the proposed systems in [15] is the amount of output greater than the amount of input, suggesting that some aspect of Nikitin’s paradigm prevents amplification.

In the current paper, we study amplification at thermodynamic equilibrium, using both theoretical arguments and experimental realization. Our theoretical results are general and apply to DNA-based systems as well as other substrates. In Section 2, we consider the natural classification of equilibrium systems according to maximum complex size, and show that dimerization networks (maximum complex size two) cannot amplify. This result explains the lack of amplification in Nikitin’s work despite the otherwise extensive functional diversity of dimerization networks.

In contrast to dimerization networks, allowing trimers permits equilibrium amplification. In Section 3, we develop a novel undercomplementary trimeric $2\times$ DNA amplifier, and demonstrate that it is able to achieve close to ideal amplification in wet-lab experiments. In contrast to the previously proposed equilibrium amplifier in the TBN model [20], our undercomplementary implementation is substantially more compact in terms of the number of strands overall, the size of the largest complexes, and the number of logical domains. Similar to the TBN amplifier, our amplifier is “composable”: the output strand has the same form as the input strand, and could act as an input to a downstream amplifier of the same design. Composability suggests a method to achieve overall increased amplification.

In the final part of this paper (Section 4), rather than focusing on structural constraints that limit equilibrium amplification, we derive bounds on the amplification factor based on the free energy of interaction between the analyte and the rest of the system. In the context of nucleic acid systems, our bound implies that the length of the detected analyte must scale linearly with the desired equilibrium amplification factor. These results formalize the intuition that a larger difference between the triggered and untriggered configurations – necessary for a greater degree of amplification – necessarily incurs an energetic cost, and we show that this cost scales linearly with the amplification factor. Since the analyte is the only strand added to the system, and all other strands remain unchanged, with only the complexes they form being altered, the requisite energy must originate from the direct interaction of the analyte with the other amplifier strands. For modular equilibrium amplifiers such as our trimeric system and the TBN amplifier [20], the impossibility result implies that the composition of more amplifier modules yields diminishing returns for a fixed analyte.

Thermodynamic limits on biochemical response have been studied most extensively for nonequilibrium sensitivity and ultrasensitivity. For example, Qian showed that the free energy from ATP/GTP hydrolysis controls the achievable ultrasensitivity in phosphorylation-dephosphorylation switches [21], and subsequent work derived universal thermodynamic response bounds for nonequilibrium steady states and tight energy bounds for covalent-modification switches [16, 18]. A related line of work bounds Hill-type sensitivity by structural quantities such as the number of binding sites or the support of the perturbation [17]. These results are complementary to the question addressed here, but do not directly address equilibrium amplification as we define it.

Our results establish fundamental principles of equilibrium-based amplification based on structural network topology (i.e., dimerization versus trimerization), as well as thermodynamic constraints. Our trimeric equilibrium amplifier could potentially be the missing piece for Nikitin-style undercomplementary schemes, allowing computation with less signal loss than is possible in dimerization-only equilibrium networks. More generally, however, our thermodynamic impossibility results rigorously justify out-of-equilibrium amplifiers as unavoidable for greater amplification in the next-generation ultrasensitive molecular sensors and signal processing architectures.

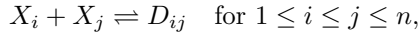
2 Structural Limitations: Dimerization Networks Cannot Amplify

In this section, we analyze the amplification capacity of a chemical reaction network limited to dimerization reactions (i.e., complexes of at most two monomers). By amplification, we mean that an increase in the input monomer leads to a larger-than-proportional increase in the concentration of some monomer species or dimer species in the network. We rigorously demonstrate that dimerization networks are structurally incapable of signal amplification, regardless of the thermodynamic parameters. Specifically, we prove that the sensitivity of any monomer or dimer species in the network to an input signal is bounded by unity.

While previous work [14] used numerical simulations to argue that dimerization networks cannot amplify, rigorous systems-level understanding of these architectures remains elusive [19]. This gap persists even as researchers increasingly push the boundaries of these architectures, such as utilizing competitive dimerization networks as analog physical computers trained via directed evolution [26].

2.1 Model Definition and Equilibrium

Consider a system comprising n distinct monomeric species $\{X_1, \dots, X_n\}$. These monomers can interact to form dimeric species D_{ij} according to the reversible reactions:



with the convention $D_{ji} = D_{ij}$. The system is assumed to be in thermodynamic equilibrium. The equilibrium concentrations, denoted by lowercase variables x_i and d_{ij} , satisfy the mass-action law:

$$d_{ij} = K_{ij} x_i x_j, \tag{1}$$

where $K_{ij} \geq 0$ is the equilibrium association constant. The case $K_{ij} = 0$ represents an absent dimer, while $K_{ij} > 0$ corresponds to an allowed dimerization reaction with finite formation free energy.

We define the total concentration of monomer X_i , denoted by C_i , as the sum of the free monomer concentration and the concentration of X_i sequestered in dimers. According to the conservation of mass:

$$C_i = x_i + \sum_{j \neq i} d_{ij} + 2d_{ii}. \tag{2}$$

2.2 Sensitivity Analysis

We examine the system's equilibrium response to adding total concentration σ of the input monomer X_1 . Specifically, we aim to derive an equation for the relative sensitivity of each monomer species, defined as the derivative of its equilibrium concentration with respect to the added input amount, normalized by its concentration. Formally, we say the *sensitivity* of species X_i is $x'_i = \frac{dx_i}{d\sigma}$, and its *relative sensitivity* is $\frac{x'_i}{x_i}$.

For added input amount σ , the total concentration of monomer X_i is $C_i = C_i^{\text{init}} + \sigma \delta_{1i}$, where δ_{1i} is the Kronecker delta. Substituting Equation (1) into Equation (2), we obtain the closed algebraic mass-balance equations:

$$x_i + \sum_{j \neq i} K_{ij} x_i x_j + 2K_{ii} x_i^2 = C_i^{\text{init}} + \sigma \delta_{1i}.$$

To determine the sensitivity of the system, we differentiate with respect to the signal σ to obtain:

$$x'_i + \sum_{j \neq i} K_{ij}(x'_i x_j + x_i x'_j) + 4K_{ii} x_i x'_i = \delta_{1i}.$$

Using the equilibrium relationship $K_{ij}x_i x_j = d_{ij}$, we can rewrite $K_{ij}x_j = d_{ij}/x_i$ and $K_{ij}x_i = d_{ij}/x_j$ to relate the relative sensitivities to concentrations of dimeric species:

$$\frac{x'_i}{x_i} \left(x_i + \sum_{j \neq i} d_{ij} + 4d_{ii} \right) + \sum_{j \neq i} d_{ij} \frac{x'_j}{x_j} = \delta_{1i}. \quad (3)$$

2.3 The Upper Bound on Amplification

The following theorem shows that no monomer species in the dimerization network can exhibit a relative sensitivity magnitude greater than that of the input monomer, and that the input monomer species itself cannot be amplified beyond the added signal. The theorem is proven in Appendix A.

► **Theorem 1** (The Input Domination Theorem). *Let σ represent the total added concentration of input monomer X_1 . In a dimerization network at equilibrium, the sensitivity of the free concentration x_1 to changes in the total input is bounded by 1, i.e., $0 < \frac{dx_1}{d\sigma} \leq 1$. Furthermore, for any monomer species X_j , the relative sensitivity satisfies $|\frac{x'_j}{x_j}| \leq |\frac{x'_1}{x_1}|$.*

2.4 System-Wide Attenuation of Equilibrium Responses

Having established that $|\frac{x'_1}{x_1}|$ is the global maximum for relative sensitivity, we now analyze the network-wide response. At the fixed equilibrium under consideration, keep the input species indexed as 1 and re-index the remaining species, breaking ties arbitrarily, so that the relative sensitivities are ordered by magnitude:

$$\left| \frac{x'_1}{x_1} \right| \geq \left| \frac{x'_2}{x_2} \right| \geq \dots \geq \left| \frac{x'_n}{x_n} \right|. \quad (4)$$

Now, let $d'_{ij} = \frac{dd_{ij}}{d\sigma}$ denote the sensitivity of the dimer D_{ij} .

► **Lemma 2** (Sign Coherence). *Given the ordering in Equation (4), for any dimer species D_{ij} with $i < j$, the sign of the sensitivity d'_{ij} is determined by the monomer species with the lower index (higher sensitivity). Specifically, if $d'_{ij} \neq 0$, then $\text{sgn}(d'_{ij}) = \text{sgn}(x'_i)$.*

Proof. From Equation (1), $d'_{ij} = d_{ij}(\frac{x'_i}{x_i} + \frac{x'_j}{x_j})$. Since $i < j$, our ordering implies $|\frac{x'_i}{x_i}| \geq |\frac{x'_j}{x_j}|$. Thus, the first term weakly dominates the sum, and the sign of d'_{ij} follows the sign of x'_i . In the symmetric case where $\frac{x'_i}{x_i} = -\frac{x'_j}{x_j}$, the terms cancel, yielding $d'_{ij} = 0$. ◀

We can now rewrite the derivative of the mass conservation equation (Equation (2)) for species i by splitting the sum into terms with indices $j > i$ and $j < i$:

$$x'_i + \sum_{j > i} d'_{ij} + 2d'_{ii} = \delta_{1i} - \sum_{j < i} d'_{ji}. \quad (5)$$

By Lemma 2, for all $j > i$, d'_{ij} has the same sign as x'_i or is zero. Similarly, d'_{ii} clearly has the same sign as x'_i . Therefore, all non-zero terms on the left-hand side of Equation (5) share the same sign. This allows us to take the absolute value of the sum as the sum of absolute values:

$$|x'_i| + \sum_{j>i} |d'_{ij}| + 2|d'_{ii}| = \left| \delta_{1i} - \sum_{j<i} d'_{ji} \right| \leq \delta_{1i} + \sum_{j<i} |d'_{ji}|. \quad (6)$$

This inequality bounds the changes in monomer species i and its dimers with higher-indexed monomers ($j > i$) by the changes in its dimers with lower-indexed monomers ($j < i$). We now generalize this to show system-wide attenuation.

► **Theorem 3** (No-Go Theorem for Dimerization Amplification). *For any monomer species X_i in the ordered dimerization network, the sensitivities of X_i and its dimers D_{ij} with $j \geq i$ satisfy:*

$$|x'_i| + \sum_{j>i} |d'_{ij}| + 2|d'_{ii}| \leq 1 - \sum_{j<i} \left(|x'_j| + 2|d'_{jj}| + \sum_{k>j} |d'_{jk}| \right). \quad (7)$$

Here, the left-hand side represents the total concentration sensitivity directly involving monomer i , including its free form and its dimers with higher-indexed species ($j \geq i$). The right-hand side shows that this quantity is bounded by the remaining portion of the unit input sensitivity after the lower-indexed terms are accounted for; the proof of Theorem 3 is given in Appendix A. The main takeaway of the theorem is that the sensitivities of both monomer species and dimers are individually bounded: $|x'_i| \leq 1$ and $|d'_{ij}| \leq 1$ for all i, j .

► **Corollary 4** (Total Variation Bound). *The sum of the absolute sensitivities of the monomer species is bounded by unity; homodimer formation strictly tightens this bound.*

$$\sum_{i=1}^n |x'_i| \leq 1 - 2 \sum_{i=1}^n |d'_{ii}|. \quad (8)$$

Proof. This follows directly by substituting $i = n$ into Equation (7). ◀

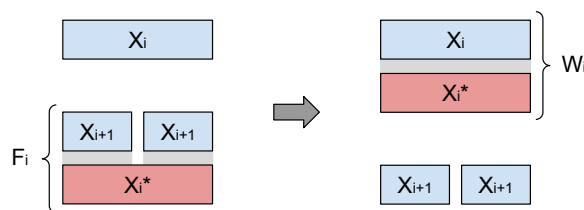
This corollary can be thought to strengthen Theorem 1 in two ways. First, Theorem 1 bounds the relative sensitivities of non-input monomers, but a small relative sensitivity could still correspond to a large absolute concentration change for an abundant species. The total variation bound rules this out. Second, even if no single monomer changes by more than the input, one might imagine a large distributed response spread across many monomer species. The corollary rules this out as well.

3 Breaking the Limit: Trimerization-Based Amplification

In Section 2, we established a fundamental structural constraint: reaction networks limited to dimerization cannot amplify an input signal, regardless of the underlying thermodynamic parameters. In this section, we show that by relaxing the constraint to allow for the formation of higher-order complexes (specifically, only trimers are needed), signal amplification becomes realizable. We propose concrete trimerization-based architectures, and present wet-lab experimental data showing amplification with a trimeric amplifier design.

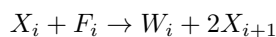
3.1 Entropy-Driven Trimer-Based Amplifier

A simple construction for a trimer-based amplifier is the entropy-driven design illustrated in Figure 1. This system leverages the thermodynamic drive toward greater global entropy (in the sense of the number of separate complexes) following the introduction of the input X_0 .



■ **Figure 1** Reaction schematic of the entropy-driven amplifier.

Intuitively, the system implements the following abstract reaction:



Such idealized reactions, when composed, implement amplification exponential in the number of modules: one copy of upstream input X_i can release two copies of downstream input X_{i+1} , doubling at every module.¹ Thus idealized, the overall stoichiometric transformation from initial reactants to final products is given by

$$X_0 + \sum_{i=0}^{n-1} 2^i F_i \rightarrow \sum_{i=0}^{n-1} 2^i W_i + 2^n X_n$$

and the theoretical (maximum) amplification factor of this network is $\beta = 2^n$.

There are two fundamental problems with this idealized analysis that limit amplification. The first is the obvious structural limitation: If the system is implemented with DNA, then, as described, each signal X_i is half the length of the previous one.² The second, deeper limitation is based on general thermodynamic considerations, follows from Section 4, and will be explored later.

In the following section, we overcome the structural limitation by introducing an *isometric* amplifier – a design that preserves the size of the output relative to the input.

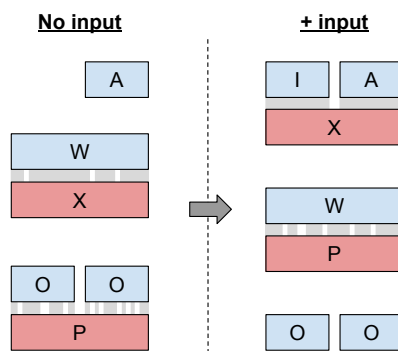
3.2 Isometric Trimer-Based Amplifier

To overcome the obvious scaling limitation of the above entropy-driven cascades, we introduce a minimal – still with maximum complex size three – network designed to amplify an input signal while strictly preserving the size of the output. This *isometric* amplification relies on a carefully engineered partial ordering of binding affinities among the system’s underlying complexes.

Figure 2 illustrates the proposed reaction pathway. Unlike the entropy-driven amplification scheme above, where the output species is smaller than the input, our isometric design generates an output molecule O of comparable size to the input I . This size preservation is a critical feature for building deeper, multi-stage cascading amplifiers so that the output of

¹ If higher-order complexes (larger than trimers) are allowed, such complexes can form by exchanging sequestered signals X_j among different fuel molecules F_i . As an example, $F_i + F_{i+1} \rightleftharpoons Q + W_{i+1}$, where Q is formed of one copy of X_i^* , one copy of X_{i+1} , and two copies of X_{i+2} . However, in the absence of input, the system still remains trapped in an equilibrium distribution without the final product X_n .

² Because each step in the cascade effectively “divides” the original input to generate multiple outputs, the amplification factor is limited by the total length of the input in base pairs $|X_0|$ and the minimum domain size κ – the minimum strand length necessary for reliable molecular recognition and binding: $\beta \leq \frac{|X_0|}{\kappa}$.



■ **Figure 2** Schematic of the proposed isometric trimerization-based amplifier. **(Left)** The thermodynamically favored configuration in the absence of the input signal I. Strand W exhibits a higher binding affinity for X than for P. Consequently, the output strands O remain sequestered in a stable trimeric complex with the “locking” molecule P, forming O:O:P. **(Right)** The thermodynamically favored configuration following the introduction of the input. Input I cooperatively binds with “helper” species A and X to form the I:A:X complex, displacing W from the W:X complex. The newly liberated strand W then drives the dissociation of the O:O:P complex by binding to P (forming W:P), thereby releasing the free output O. The forward directionality of this cascade is governed by the thermodynamic hierarchy of the complex affinities: $K_{I:A:X} > K_{W:X} > K_{W:P} > K_{O:O:P}$. Crucially, in the absence of I, species A possesses insufficient affinity to displace X or P. *Note: An increased density of white gaps within the gray binding regions visually denotes progressively weaker binding affinities.*

one stage can serve as an input for the subsequent stage. Thermodynamically, the system is designed such that enough energy is released upon the formation of the input complex I:A:X to effectively “pay” the energetic penalty required to dissociate the output complex O:O:P.

In order to avoid leak (release of O without I), we must limit the affinities in the affinity hierarchy noted in the figure caption. Specifically, without input I, molecule A can displace half of the W:X complex. Then this free half of W may displace an output molecule O from the O:O:P complex, i.e., $A + W:X + O:O:P \rightarrow A:W:X:O:P + O$.³ This reaction pathway is inhibited because it reduces the number of separate complexes by one, but we need to ensure that the increased bonding does not compensate for the loss of entropy.

By strategically engineering sequence mismatches, we can tune the free energies of the complexes, establishing the required partial order of binding affinities and their relationship to separate complex formation entropy. Furthermore, because of partial complementarity, the sequence of O can be sufficiently different from I that several identical isometric amplification modules could be composed, where the output of one is input to another.

Experimental Results

To validate the theoretical framework of the isometric trimer-based amplifier, we implemented an experimental model using synthetic DNA. First, we verified the theoretical performance of our designed sequences using NUPACK [8]. As observed in Figure 3a, while the simulated system exhibited some leakage (release of output in the absence of input) at a higher temperature (28°C), this leak was effectively eliminated by decreasing the temperature. Overall, the NUPACK-simulated response closely follows the ideal 2× slope until reaching saturation.

³ While our theoretical model restricts complex size to three, mitigating the formation of higher-multimericity off-model polymers remains a practical engineering goal.

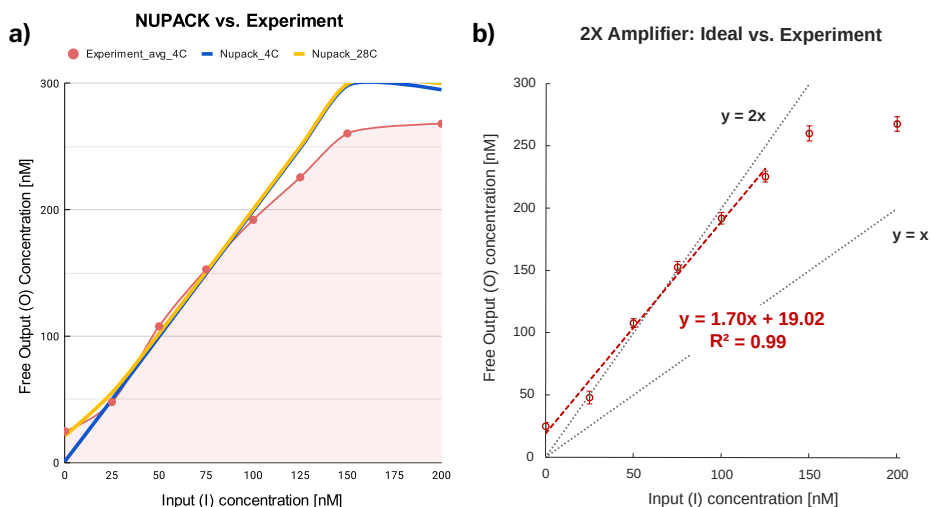


Figure 3 Experimental results of the isometric amplifier. The input concentration is varied from 0 to 200 nM, the total concentration of the output strand is 300 nM, and all other circuit strands are set at 150 nM. **(a)** Experimental data plotted against NUPACK estimated behavior. **(b)** Experimental data plotted against the ideal 2× response. Linear regression of the response regime yields an experimental amplification factor of 1.7×.

The experimental data shown in Figure 3(a)-(b) was collected at 4°C, but in contrast to NUPACK predictions, it still exhibits non-negligible leakage in the absence of the input (0 nM). This minor discrepancy may stem from several factors: imprecise stoichiometry of the added components, DNA synthesis impurities (as only the fluorophore-tagged output strand was HPLC-purified, while the rest were purified via standard PAGE), or fluorophore-induced energetic perturbations not captured by NUPACK.

As shown in Figure 3b, to quantify the experimental amplification factor, we performed linear regression. While the theoretical maximum amplification factor of this specific system is 2×, our experimental data demonstrates 1.7× signal amplification. The regression analysis excluded the saturated data points (input concentrations of 150 nM and 200 nM) to capture the true linear amplification regime. (If the leakage point is also excluded from the fit to strictly evaluate the active response slope, the calculated amplification factor rises to 1.77×.) Complete details of the experimental protocols, DNA sequences, and statistical methods are provided in the full version of this paper [1].

Since there is no physical limit imposed by shrinking molecule sizes as in the naive construction of Section 3.1, one might intuitively expect that arbitrary signal amplification is physically realizable. However, escaping structural limitations does not grant unlimited amplification. In the following section, we establish that regardless of a system’s specific structural architecture or local mechanisms, there exists a fundamental, inescapable thermodynamic limit on signal amplification for *all* thermodynamic equilibrium-based networks.

4 Thermodynamic Bounds on Amplification

In the previous sections, we demonstrated that fundamental structural differences (dimerization versus trimerization) can disallow or permit amplification. We have also seen that the trimeric amplifier could in principle be composed to seemingly increase its amplification factor. In this section, we turn to general and unavoidable limits on amplification at thermodynamic equilibrium determined by thermodynamic considerations, regardless of

structural assumptions. Unlike prior thermodynamic bounds on nonequilibrium biochemical response [21, 16, 18], the systems considered here have no persistent chemical-potential driving: both the pre-input and post-input configurations are equilibria. The relevant resource is therefore the binding free energy gained when the added input analyte interacts with amplifier components.

There is a strong intuition that chemical amplification should require significant energy expenditure since the system must traverse substantial “configurational distance” from a state with little output to one with a lot of output. Since we are requiring the system before and after the addition of input to be at thermodynamic equilibrium, the only source of this energy must be from the input itself. This section formalizes this reasoning, showing that the free energy with which the input binds amplifier components must scale linearly with the amplification factor. In the context of DNA nanotechnology, this implies that for an analyte (DNA molecule) of a fixed length, there is only so much amplification that can be realized. Moreover, we show that the achievable amplification is further discounted by the *output contrast*, defined as the ratio of the triggered output (signal) to the untriggered output (leak). For a fixed input free energy, demanding a sharper contrast leaves less room for amplification.

4.1 Model

We represent the monomer composition of any complex as a vector tuple $(u, v) \in \mathbb{N}^m \times \mathbb{N}$, where u is the vector of stoichiometric counts of non-input monomers, and v is the count of the input monomer I . Given a complex (u, v) , we define its *residual core* to be just its non-input monomers. The L_1 -norm $|u| = \sum_{i=1}^m u_i$ of (u, v) is the size of its residual core. Similarly, we define the *input core* of (u, v) as its constituent input monomers, with its size naturally given by v .

The concentration of a specific complex is denoted $x_{u,v}$. This vector representation naturally partitions the network into three distinct pools: pure residual complexes ($v = 0, u \neq \mathbf{0}$), pure input complexes ($u = \mathbf{0}, v \geq 1$), and mixed complexes ($u \neq \mathbf{0}, v \geq 1$).⁴ All concentrations are dimensionless mole fractions – the ratio of complexes to solvent molecules (i.e., water) – and are thus smaller than one. (The derivation of the energy function (Equation (9)) only holds in the regime of substantially more dilute complexes than solvent [4].)

To model the equilibrium behavior of these systems, we adopt the free-energy formulation utilized by Dirks *et al.* [4]. Let $g(x)$ denote the dimensionless free energy of the overall system configuration (formula given below), let $G_{u,v}$ represent the dimensionless free energy of a specific complex, and let $\mathcal{P}$ denote the set of all valid complexes in the network. We define the sign convention for $G_{u,v}$ such that more negative values correspond to more thermodynamically favorable complexes.

Let σ represent the total concentration of the input monomer I . The equilibrium concentrations are obtained by minimizing the system’s total free energy (corresponding to the pseudo-Helmholtz free energy utilized extensively throughout the chemical reaction network theory literature [9, 7]):⁵

$$g(x) = \sum_{(u,v) \in \mathcal{P}} x_{u,v} (\log x_{u,v} - 1 + G_{u,v}) \quad (9)$$

⁴ There is no complex ($u = \mathbf{0}, v = 0$).

⁵ In this work, $\log$ denotes the natural logarithm.

This minimization is subject to the mass conservation constraint $Ax = x^0(\sigma)$, where x is the vector of complex concentrations, $x^0(\sigma)$ is the vector representing the total concentrations of all monomers (including the amount σ of added input I), and A is the mass conservation matrix mapping the concentrations of complexes to their constituent monomers.

To formally manipulate the system configurations, we define two projection operators that isolate the cores from the overall complex concentration vector x .

The **Input-Core operator**, denoted $IC(x)$, extracts the input cores by stripping away all residual components. In other words, it removes all residual monomers, and for every $v \geq 1$, creates a complex containing v input monomers whose concentration is the sum of the concentrations of all complexes containing exactly v input monomers in x . Formally, we define the resulting vector $w = IC(x)$:

$$w = IC(x) \implies \begin{cases} w_{(0,v)} = \sum_u x_{(u,v)} \\ w_{(u,v)} = 0 \end{cases} \quad \text{for } u \neq 0 \quad (10)$$

Conversely, the **Residual-Core operator**, denoted $RC(x)$, extracts the residual cores by stripping away all input monomers. Analogously to Equation (10), we define the resulting vector $w = RC(x)$:

$$w = RC(x) \implies \begin{cases} w_{(u,0)} = \sum_v x_{(u,v)} \\ w_{(u,v)} = 0 \end{cases} \quad \text{for } v \geq 1 \quad (11)$$

4.2 Energy Change upon Input Addition

We now investigate the total change in system free energy upon the introduction of the input. To quantify this transition, we compare the thermodynamic equilibria of the network before and after the input is introduced – configurations we formalize as the *Resting* and *Active* states, respectively. To establish a rigorous mathematical bound on this free-energy difference, we will also define a specific configuration termed the *Drained* state.

The Active state is the thermodynamic equilibrium of the entire system with concentration σ of the input I . Formally:

► **Definition 5.** For any input concentration $\sigma \geq 0$, the **Active state** $\mathbf{x}$ is the concentration vector that minimizes the total system free energy $g(x)$ subject to mass conservation:

$$\begin{aligned} \mathbf{x} = \operatorname{argmin}_{x \geq 0} \quad & g(x) \\ \text{s.t.} \quad & Ax = x^0(\sigma) \end{aligned} \quad (12)$$

Next we define the Drained state, which represents “ripping” the input out of the Active state without allowing the system to relax to a new equilibrium. Formally, the Drained state is constructed by separating the input cores and residual cores of the Active state:

► **Definition 6.** *Drained state* $\mathbf{y}$ is defined by:

$$\mathbf{y} = IC(\mathbf{x}) + RC(\mathbf{x}) \quad (13)$$

Note that the dissociation into input cores and residual cores preserves mass conservation $A\mathbf{y} = x^0(\sigma)$.

Finally, we define the Resting state, which represents the thermodynamic equilibrium without input to which the input is added without being able to interact with the rest of the system. More precisely, the Resting state is the thermodynamic equilibrium of the system without the input, plus $IC(\mathbf{x})$. Formally:

► **Definition 7.** *Resting state $\mathbf{z}$ is defined by:*

$$\begin{aligned} \mathbf{z} &= IC(\mathbf{x}) + \operatorname{argmin}_{x \geq 0} g(x) \\ \text{s.t. } Ax &= x^0(0) \end{aligned} \quad (14)$$

► **Lemma 8 (Free Energy Hierarchy).** *The free energies of the Active, Resting, and Drained states satisfy the following inequality:*

$$g(\mathbf{x}) \leq g(\mathbf{z}) \leq g(\mathbf{y})$$

The proof of Lemma 8 is given in Appendix B.

We now define the following quantities with respect to the Active state $\mathbf{x}$. Let $\mathcal{U} = \{u \neq \mathbf{0} \mid \exists v \geq 1: \mathbf{x}_{u,v} > 0\}$ denote the set of residual stoichiometries u that can form complexes with the input in the Active state. Let $X_{\text{inp}} = \sum_{v \geq 1} \mathbf{x}_{\mathbf{0},v}$ be the total concentration of the pure input complexes, $X_{\text{res}} = \sum_{u \in \mathcal{U}} \mathbf{x}_{u,0}$ be the total concentration of the pure residual complexes, and let $X_{\text{mix}} = \sum_{u \in \mathcal{U}} \sum_{v \geq 1} \mathbf{x}_{u,v}$ be the total concentration of the mixed complexes.

Next, we define the binding free energy, $\Delta G_{u,v}^{\text{bind}} := G_{u,v} - G_{u,0} - G_{\mathbf{0},v}$, which quantifies the thermodynamic cost of forming the complex (u, v) from its isolated residual and input cores. Finally, we let $\Delta G_{\text{stg}}^{\text{inp}} = \min_{u,v} \Delta G_{u,v}^{\text{bind}}$ denote the most favorable binding energy achievable within the network upon interaction with the input. Note that we use the convention that $\Delta G_{\text{stg}}^{\text{inp}} < 0$, and the more negative it is, the more favorable the interaction.

Having formally defined the Active, Drained, and Resting states, our primary objective is to quantify the total change in the pseudo-Helmholtz free energy of the system upon the addition of an input species with concentration σ . The Drained state serves as a crucial analytical intermediate. Starting from the Active state, where the input has already propagated through the network to trigger the targeted outputs, we conceptually decouple all bound input cores from the residual network. If we then allow the residual cores of this Drained state to thermodynamically relax while remaining strictly isolated from the input pool, the network equilibrates into the Resting state.

Altogether, by comparing the Active, Drained, and Resting states, and incorporating the strongest potential energy released by input binding ($\Delta G_{\text{stg}}^{\text{inp}}$), we can rigorously bound the total pseudo-Helmholtz free energy change of the system as shown below. We show that the bound depends on the following dimensionless parameter, which we term amplification capacity. Intuitively, the upper bound on amplification capacity is the maximum energy with which the input binds ($-\Delta G_{\text{stg}}^{\text{inp}}$). From this upper bound, we subtract the log of “concentration of amplifier components” (as captured by $X_{\text{res}} + X_{\text{mix}}$) – with smaller concentrations leading to smaller amplification capacity.⁶

► **Definition 9.** *We define the amplification potential as $\psi(X, \Delta G) := -\Delta G + \log(X)$. Hence, the input amplification potential is denoted by $\psi_{\text{inp}} := \psi(X_{\text{res}} + X_{\text{mix}}, \Delta G_{\text{stg}}^{\text{inp}})$. We then define the amplification capacity ϕ as follows:*

$$\phi := \begin{cases} e^{\psi_{\text{inp}}} & \text{if } \psi_{\text{inp}} \leq 0, \\ \psi_{\text{inp}} + 1 & \text{if } \psi_{\text{inp}} > 0. \end{cases} \quad (15)$$

⁶ Recall that all concentrations are strictly less than one, as they are expressed as dimensionless mole fractions (the ratio of complexes to solvent molecules). Consequently, $X_{\text{res}} + X_{\text{mix}} < 1$.

Below we will derive upper bounds on the amplification factor as a function of the amplification capacity, justifying this terminology.

► **Theorem 10** (Input-Driven Energy Change). *Let σ be the total added concentration of the input species. Let $g(\mathbf{y})$ denote the free energy of the Drained state, let $g(\mathbf{x})$ denote the free energy of the Active state. Then, the amplification capacity ϕ satisfies:*

$$g(\mathbf{y}) - g(\mathbf{x}) \leq \sigma \phi \quad (16)$$

The proof of Theorem 10 is given in Appendix B.

► **Corollary 11.** *A consequence of the free energy hierarchy (Lemma 8) is that the intermediate free energy transitions are likewise bounded. Specifically, because $g(\mathbf{x}) \leq g(\mathbf{z}) \leq g(\mathbf{y})$, it holds that both $g(\mathbf{z}) - g(\mathbf{x}) \leq \sigma \phi$ and $g(\mathbf{y}) - g(\mathbf{z}) \leq \sigma \phi$.*

To connect these thermodynamic constraints to system performance, we now translate the bounded free energy transitions into concentrations, yielding an explicit upper bound on the amplification factor.

4.3 Amplification Factor Upper Bound

Fix an input amount σ . Intuitively, we want the output amount to be much larger than the leak, which is the output amount in the absence of input. For actual amplification, we also want the output amount to be much larger than the input amount σ . The bound on energy release as a function of amplification capacity reported in Theorem 10 can be recast into an explicit relationship between these quantities, properly formalized.

Let O denote the designated free-output species, and define its concentration at the Active and Resting states as:

$$a := \mathbf{x}_O, \quad r := \mathbf{z}_O$$

In other words, a is the signal amount of output when the amplifier is activated, and r is the leak amount. We define the amplification factor β and the output contrast γ as:

$$\beta := \frac{a - r}{\sigma}, \quad \gamma := \frac{a}{r}.$$

Thus the amplification factor β represents the amount of output increase between leak and signal, relative to the input amount. Output contrast γ represents the ratio of the free output species between leak and signal states.

► **Theorem 12** (Thermodynamic Bound on Amplification). *Under the natural assumption for amplification that the free-output concentration in the Active state exceeds its Resting leak ($\gamma > 1$), the amplification factor β is upper-bounded by amplification capacity ϕ discounted by the logarithmic output contrast:*

$$\beta \leq \frac{\phi}{\log \gamma}. \quad (17)$$

In the regime relevant for amplification ($\psi_{\text{inp}} > 0$), the amplification capacity $\phi = \psi_{\text{inp}} + 1$ grows linearly with the strength of the input interaction $-\Delta G_{\text{stg}}^{\text{inp}}$. A key takeaway of this theorem is therefore that the input must bind the amplifier components with a free energy that scales (at least) linearly with the desired amplification factor (at fixed contrast).

8:14 Amplification at Equilibrium: Limitations and Implementation

► **Remark 13.** In the positive input potential regime ($\psi_{\text{inp}} > 0$), expanding ϕ in Theorem 12 reveals:

$$\beta \leq \frac{-\Delta G_{\text{stg}}^{\text{inp}} + \log(X_{\text{res}} + X_{\text{mix}}) + 1}{\log \gamma} < \frac{-\Delta G_{\text{stg}}^{\text{inp}} + 1}{\log \gamma}. \quad (18)$$

Proof of Theorem 12. The Resting and Drained states share the same isolated input pool, so their free-energy difference is entirely a residual-subsystem difference. The residual part of $\mathbf{z}$ is the equilibrium minimizer subject to the isolated residual mass constraint. Hence the pseudo-Helmholtz Bregman identity gives [3]:

$$g(\mathbf{y}) - g(\mathbf{z}) = \sum_i \left(\mathbf{y}_i \log \frac{\mathbf{y}_i}{\mathbf{z}_i} - \mathbf{y}_i + \mathbf{z}_i \right),$$

where the sum is over the pure residual complexes. Each summand is nonnegative. Keeping only the output O :

$$g(\mathbf{y}) - g(\mathbf{z}) \geq \mathbf{y}_O \log \frac{\mathbf{y}_O}{\mathbf{z}_O} - \mathbf{y}_O + \mathbf{z}_O.$$

We know the output O is a pure residual species and that draining the input from the Active state may increase its concentration $\mathbf{y}_O \geq \mathbf{x}_O = a$. This holds because a fraction of the total output mass might be bound with input species in the Active state. Letting $r = \mathbf{z}_O$, define

$$F(t) := t \log \frac{t}{r} - t + r.$$

Since $F'(t) = \log(t/r)$, the function F is increasing for $t \geq r$. Applying $\mathbf{y}_O \geq a$ along with $a > r$ yields:

$$g(\mathbf{y}) - g(\mathbf{z}) \geq a \log \frac{a}{r} - a + r. \quad (19)$$

A sharper estimate is obtained by also retaining the output contribution between the Resting and Active states. Since the Active state is the equilibrium minimizer for the full system under the same total mass constraint, the pseudo-Helmholtz Bregman identity similarly gives:

$$g(\mathbf{z}) - g(\mathbf{x}) \geq r \log \frac{r}{a} - r + a. \quad (20)$$

Adding Equations (19) and (20) gives:

$$\begin{aligned} g(\mathbf{y}) - g(\mathbf{x}) &\geq \left(a \log \frac{a}{r} - a + r \right) + \left(r \log \frac{r}{a} - r + a \right) \\ &= (a - r) \log \frac{a}{r}. \end{aligned}$$

Substituting $\gamma = a/r$ and $a - r = \beta\sigma$ gives:

$$g(\mathbf{y}) - g(\mathbf{x}) \geq \beta\sigma \log \gamma.$$

► **Remark 14.** One of the reasons Theorem 12 is not strictly tight is the truncation of terms in Equations (19) and (20). Nonetheless, looking at the entire sum may be appropriate to generate design strategies and effective amplifiers. One such heuristic is apparent: limit the variety of distinct chemical species that do not contain the output, which decreases the number of non-zero terms in the sum.

Combining this lower bound with Theorem 10,

$$g(\mathbf{y}) - g(\mathbf{x}) \leq \sigma\phi,$$

and cancelling σ completes the proof. $\blacktriangleleft$

To illustrate the order of magnitude of these thermodynamic limits, we evaluate the upper bound using the parameters of the isometric trimeric amplifier from Section 3.2. Our input strand has a maximum binding affinity of roughly -49 kcal/mol at the operating temperature of 4°C (277.15 K). Assuming an overestimate of $X_{\text{mix}} + X_{\text{res}} < 1$ μM , we obtain the following dimensionless parameter values⁷:

$$\Delta G_{\text{stg}}^{\text{inp}} \approx -89, \quad X_{\text{res}} + X_{\text{mix}} \lesssim 1.8 \times 10^{-8}.$$

Substituting these values into the definition for the amplification capacity yields:

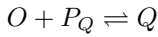
$$\phi = \psi_{\text{inp}} + 1 = -\Delta G_{\text{stg}}^{\text{inp}} + \log(X_{\text{res}} + X_{\text{mix}}) + 1 \approx 72.$$

Under these conditions, requiring a modest 10-fold target output contrast ($\gamma = 10$) restricts the maximum achievable amplification factor to $\beta \lesssim 31$. This concrete example demonstrates how a seemingly vast input free-energy budget (hybridization energy of -49 kcal/mol is typically considered all but irreversible) is naturally attenuated by the system's concentration scales and signal contrast requirements, leaving a ceiling on the final equilibrium gain.⁸

4.4 Output Contrast Upper Bound

In the previous section, Theorem 12 bounds the amplification factor β as a function of amplification capacity ϕ and the contrast γ . In this section, our results address the following intuition in turn: to have high contrast, the output must be capable of binding strongly with something in order to have low leak. Thus, we turn from the free energy of binding the input to that of binding the output.

In order to link contrast to the underlying thermodynamics of output sequestration in the Resting state, suppose a set of pure residual complexes $\mathcal{Q}$ sequesters the output species in the Resting state. Each complex $Q \in \mathcal{Q}$ forms via the reversible binding of one free output molecule O to a distinct residual partner P_Q ⁹:



We define the sequestration free-energy gain for each complex as $\Delta G_Q^{\text{out}} := G_Q - G_O - G_{P_Q}$. Let $\Delta G_{\text{stg}}^{\text{out}} := \min_{Q \in \mathcal{Q}} \Delta G_Q^{\text{out}}$ denote the most favorable binding energy achievable within the network upon the output sequestration. Note that similar to $\Delta G_{\text{stg}}^{\text{inp}}$, we use the convention that $\Delta G_{\text{stg}}^{\text{out}} < 0$, and the more negative it is, the more favorable the interaction.

By the law of mass action, the Resting equilibrium configuration satisfies:

$$\mathbf{z}_Q = \mathbf{z}_O \mathbf{z}_{P_Q} e^{-\Delta G_Q^{\text{out}}} \quad (21)$$

⁷ Using the molarity of water (55.49 M at 4°C) as the solvent to non-dimensionalize the system, and adopting the ideal gas constant $R = 0.00198717$ kcal/mol.K.

⁸ As discussed in Section 5, the tightness of our results remains open. While we have not demonstrated multi-stage composition and the bound need not be tight, it leaves open the possibility that 4 layers of the isometric trimeric amplifier could be composed, since each contributes a factor of 2 and $2^4 = 16 \leq 31 < 32 = 2^5$.

⁹ If Q includes more than one copy of O , then P_Q includes O as well, which does not cause a problem for the subsequent reasoning steps.

Let $Z_{\text{res}}^{\text{out}} := \sum_{Q \in \mathcal{Q}} \mathbf{z}_{P_Q}$. This quantity serves as the output-side counterpart to X_{res} : whereas X_{res} represents the total concentration of complexes capable of binding the input species in the Active state, $Z_{\text{res}}^{\text{out}}$ represents the total concentration of complexes P_Q capable of sequestering the output species in the Resting state. Furthermore, let ν_{max} be the largest output stoichiometry of any complex Q .

Finally, the output potential ψ_{out} is defined as follows:

$$\psi_{\text{out}} := \psi(\nu_{\text{max}} Z_{\text{res}}^{\text{out}}, \Delta G_{\text{stg}}^{\text{out}}) = -\Delta G_{\text{stg}}^{\text{out}} + \log(\nu_{\text{max}} Z_{\text{res}}^{\text{out}}).$$

► **Lemma 15** (Contrast Upper Bound). *The output contrast γ of an amplifier is bounded by: $\gamma \leq 1 + e^{\psi_{\text{out}}}$.*

Proof. The net increase in the free output concentration, $a - r$, cannot exceed the total amount of sequestered output available in the Resting state. Recalling that the Resting free output concentration is $\mathbf{z}_O = r$, we apply Equation (21) to bound this difference:

$$a - r \leq \nu_{\text{max}} \sum_{Q \in \mathcal{Q}} \mathbf{z}_Q = \nu_{\text{max}} r \sum_{Q \in \mathcal{Q}} \mathbf{z}_{P_Q} e^{-\Delta G_Q^{\text{out}}} \leq r \left(\nu_{\text{max}} Z_{\text{res}}^{\text{out}} e^{-\Delta G_{\text{stg}}^{\text{out}}} \right).$$

Recognizing that the term in the parentheses is exactly equivalent to $e^{\psi_{\text{out}}}$, we obtain $a - r \leq r e^{\psi_{\text{out}}}$. Dividing the entire inequality by r and rearranging for $\gamma := a/r$ completes the proof. ◀

Lemma 15 together with Theorem 12 reveals the following big picture. Because the output contrast γ acts as the denominator in the overall amplification bound (Theorem 12), enforcing a high signal contrast inherently restricts the maximum achievable amplification factor β . Conversely, designing a system with weaker output sequestration (a less negative $\Delta G_{\text{stg}}^{\text{out}}$) mathematically expands the potential for amplification, but leaves the system susceptible to leak and poor contrast. Therefore, to increase the amplification upper bound *without* sacrificing signal clarity, one must instead drive the system with a more favorable input free energy $\Delta G_{\text{stg}}^{\text{inp}}$, thereby raising the overall amplification capacity ϕ .

► **Remark 16.** Lemma 15 implies that to achieve a target output contrast $\gamma > 1$, the output sequestration free-energy must be favorable enough (recall $\Delta G_{\text{stg}}^{\text{out}} < 0$ by convention):

$$\Delta G_{\text{stg}}^{\text{out}} \leq \log \left(\frac{\nu_{\text{max}} Z_{\text{res}}^{\text{out}}}{\gamma - 1} \right). \quad (22)$$

In practice, bounding the leakage is critical to avoid prematurely triggering downstream cascades or violating diagnostic sensitivity thresholds. To guarantee a target contrast γ for these applications, Equation (22) provides a direct design heuristic: the intrinsic sequestration affinity $\Delta G_{\text{stg}}^{\text{out}}$ must be driven sufficiently negative, taking into account the total concentration of available residual partners $Z_{\text{res}}^{\text{out}}$ and their maximum output sequestration size ν_{max} .

To complete our analysis of the isometric amplifier example, we examine the thermodynamic constraints on the output side of the network. In our experiment, the hybridization energy of the output O is -24 kcal/mol under our conditions. While this is substantially less favorable than the hybridization of the input, it is in principle compatible with very large contrast depending on $Z_{\text{res}}^{\text{out}}$. Although $Z_{\text{res}}^{\text{out}}$ is not directly under our control, we could imagine that adding a reasonable excess of strand P results in nanomolar quantities of free P in equilibrium. Then the corresponding dimensionless parameters are evaluated as:

$$\Delta G_{\text{stg}}^{\text{out}} \approx -44, \quad \nu_{\text{max}} = 2, \quad \text{and} \quad Z_{\text{res}}^{\text{out}} \lesssim 1.8 \times 10^{-11}.$$

Substituting these values into the expression for the output amplification potential yields:

$$\psi_{\text{out}} = -\Delta G_{\text{stg}}^{\text{out}} + \log(\nu_{\text{max}} Z_{\text{res}}^{\text{out}}) \approx 19.$$

By Lemma 15, this potential yields a bound on the output contrast of $\gamma \lesssim e^{19}$. While operating near this extreme contrast limit would be highly effective for minimizing leak, it heavily penalizes the system's amplification β by Theorem 12. Alongside an amplification capacity of $\phi \approx 72$ from the previous section, we see that the maximum achievable amplification factor becomes constrained to:

$$\beta \leq \frac{\phi}{\log \gamma} \approx \frac{72}{19} \approx 3.8.$$

5 Conclusion

In this work, we first provided a rigorous proof demonstrating that equilibrium dimerization networks are fundamentally incapable of signal amplification, a problem previously addressed only through numerical approximations due to its inherent algebraic complexity. Our results help explain the lack of signal amplification in undercomplementary designs proposed by Nikitin [15] as they all rely on a maximum complex size of two. We subsequently evaluated the amplification potential of equilibrium trimerization networks. Our investigation of an entropy-driven trimerization network confirmed its capacity to amplify, but revealed a size-shrinking feature that inherently limits its amplification power. To resolve this, we designed an isometric trimer-based equilibrium amplifier that successfully achieved signal amplification while strictly preserving the length of the output strands, potentially allowing for module composition. We validated this architecture through the wet-lab implementation of a $2\times$ amplifier, achieving an experimental amplification factor that closely matches this theoretical target.

We finally broadened our scope to investigate fundamental limits on equilibrium amplification in a more general case based on thermodynamic constraints, regardless of the multimericity of the network. We demonstrated that the free energy released by the input binding determines the theoretical ceiling of the amplification power; the network design then dictates how much of this capacity is attenuated. Specifically, we established that enforcing a high signal contrast fundamentally limits the maximum achievable amplification factor. Furthermore, we proved that the theoretical ceiling of this output contrast is itself dictated by the free energy of output sequestration. Ultimately, these thermodynamic bounds reveal a universal engineering trade-off: securing a robust, high-contrast signal requires deep output sequestration to suppress leak, which in turn demands a proportionally larger input free-energy budget to successfully drive molecular amplification.

There are two natural directions for future work. First, we aim to experimentally compose multiple amplifier modules to achieve greater than a factor-of-two amplification, using our component-level heuristics (e.g., Remarks 14 and 17) to guide the physical design. Second, we must investigate the tightness of our theorems by testing how closely these physical implementations can approach our established thermodynamic limits for amplification (Theorem 12) and contrast (Lemma 15). Ultimately, this leads to a fundamental open question: subject to these thermodynamic limits, does an “optimal” equilibrium amplifier or amplifier family exist?

References

- 1 Hamidreza Akef, Chia-Yu Sung, Aneesh Vanguri, and David Soloveichik. Amplification at equilibrium: Structural and thermodynamic limitations, and implementation, 2026. doi:10.48550/arXiv.2604.04285.
- 2 Hui Chen, Shaohong Zhou, Kleins Ngocho, Jing Zheng, Xiaoxiao He, Jin Huang, Kemin Wang, Hui Shi, and Jianbo Liu. Oriented triplex DNA as a synthetic receptor for transmembrane signal transduction. *Nature Communications*, 15(1):9789, 2024. doi:10.1038/s41467-024-53960-5.
- 3 Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 3(1):146–158, 1975. doi:10.1214/aop/1176996454.
- 4 Robert M Dirks, Justin S Bois, Joseph M Schaeffer, Erik Winfree, and Niles A Pierce. Thermodynamic analysis of interacting nucleic acid strands. *SIAM review*, 49(1):65–88, 2007. doi:10.1137/060651100.
- 5 Robert M. Dirks and Niles A. Pierce. Triggered amplification by hybridization chain reaction. *Proceedings of the National Academy of Sciences*, 101(43):15275–15278, 2004. doi:10.1073/pnas.0407024101.
- 6 David Doty, Trent A. Rogers, David Soloveichik, Chris Thachuk, and Damien Woods. Thermodynamic binding networks. In Robert Brijder and Lulu Qian, editors, *DNA Computing and Molecular Programming*, pages 249–266, Cham, 2017. Springer International Publishing. doi:10.1007/978-3-319-66799-7_16.
- 7 Martin Feinberg. *Foundations of chemical reaction network theory*. Springer, 2019.
- 8 Mark E. Fornace, Jining Huang, Cody T. Newman, Avinash Nanjundiah, Nicholas J. Porubsky, Marshall B. Pierce, and Niles A. Pierce. NUPACK: Analysis and design of nucleic acid structures, devices, and systems. ChemRxiv preprint, 2025. Version 2. doi:10.26434/chemrxiv-2022-xv981-v2.
- 9 Fritz Horn and Roy Jackson. General mass action kinetics. *Archive for rational mechanics and analysis*, 47:81–116, 1972.
- 10 Yu (Sherry) Jiang, Bingling Li, John N. Milligan, Sanchita Bhadra, and Andrew D. Ellington. Real-time detection of isothermal amplification reactions with thermostable catalytic hairpin assembly. *Journal of the American Chemical Society*, 135(20):7430–7433, 2013. PMID: 23647466. doi:10.1021/ja4023978.
- 11 Erel Levine, Zhongge Zhang, Thomas Kuhlman, and Terence Hwa. Quantitative characteristics of gene regulation by small RNA. *PLOS Biology*, 5(9):e229, 2007. doi:10.1371/journal.pbio.0050229.
- 12 Bingling Li, Andrew D. Ellington, and Xi Chen. Rational, modular adaptation of enzyme-free DNA circuits to multiple detection methods. *Nucleic Acids Research*, 39(16):e110–e110, June 2011. doi:10.1093/nar/gkr504.
- 13 Yanhao Liu, Wan Zhao, Yingming Zhao, Xiangxiang Zhang, Jingjing Zhao, Shubin Li, Yongshuo Ren, and Xiaojun Han. An artificial cell capable of signal transduction mediated by adrb2 for the regulation of glycogenolysis. *Nature Communications*, 17(1):1795, 2026. doi:10.1038/s41467-026-68503-3.
- 14 Sergei Maslov and I. Ispolatov. Propagation of large concentration changes in reversible protein-binding networks. *Proceedings of the National Academy of Sciences*, 104(34):13655–13660, 2007. doi:10.1073/pnas.0702905104.
- 15 Maksim P. Nikitin. Non-complementary strand commutation as a fundamental alternative for information processing by dna and gene regulation. *Nature Chemistry*, 15(1):70–82, 2023. doi:10.1038/s41557-022-01111-y.
- 16 Jeremy A. Owen, Todd R. Gingrich, and Jordan M. Horowitz. Universal thermodynamic bounds on nonequilibrium response with biochemical applications. *Physical Review X*, 10(1):011066, 2020. doi:10.1103/PhysRevX.10.011066.
- 17 Jeremy A. Owen and Jordan M. Horowitz. Size limits the sensitivity of kinetic schemes. *Nature Communications*, 14(1):1280, 2023. doi:10.1038/s41467-023-36705-8.

- 18 Jeremy A. Owen, Pranay Talla, John W. Biddle, and Jeremy Gunawardena. Thermodynamic bounds on ultrasensitivity in covalent switching. *Biophysical Journal*, 122(10):1833–1845, 2023. doi:10.1016/j.bpj.2023.04.015.
- 19 Jacob Parres-Gold, Matthew Levine, Benjamin Emert, Andrew Stuart, and Michael B. Elowitz. Contextual computation by competitive protein dimerization networks. *Cell*, 188(7):1984–2002.e17, 2025. doi:10.1016/j.cell.2025.01.036.
- 20 Joshua Petrack, David Soloveichik, and David Doty. Thermodynamically Driven Signal Amplification. In Ho-Lin Chen and Constantine G. Evans, editors, *29th International Conference on DNA Computing and Molecular Programming (DNA 29)*, volume 276 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 8:1–8:22, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.29.8.
- 21 Hong Qian. Thermodynamic and kinetic analysis of sensitivity amplification in biological signal transduction. *Biophysical Chemistry*, 105(2–3):585–593, 2003. doi:10.1016/S0301-4622(03)00068-1.
- 22 Lulu Qian and Erik Winfree. Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332(6034):1196–1201, 2011. doi:10.1126/science.1200520.
- 23 Georg Seelig, David Soloveichik, David Yu Zhang, and Erik Winfree. Enzyme-free nucleic acid logic circuits. *Science*, 314(5805):1585–1588, 2006. doi:10.1126/science.1132493.
- 24 Tristan Stérin, Abeer Eshra, Janet Adio, Constantine Glen Evans, and Damien Woods. A thermodynamically favoured molecular computer: Robust, fast, renewable, scalable. *bioRxiv*, 2025. doi:10.1101/2025.07.16.664196.
- 25 Jimeng Su, Ying Song, Zhipeng Zhu, Xinyue Huang, Jibiao Fan, Jie Qiao, and Fengbiao Mao. Cell–cell communication: new insights and clinical implications. *Signal Transduction and Targeted Therapy*, 9(1):196, 2024. doi:10.1038/s41392-024-01888-z.
- 26 Alexei V. Tkachenko, Bortolo Matteo Mognetti, and Sergei Maslov. Evolutionary chemical learning in dimerization networks, 2025. doi:10.48550/arXiv.2506.14006.

A Proofs for Section 2 (Structural Limitations: Dimerization Networks Cannot Amplify)

Proof of Theorem 1. Let k be the index of the monomer species with the maximum absolute relative sensitivity. That is $\left| \frac{x'_k}{x_k} \right| = \max_j \left| \frac{x'_j}{x_j} \right|$.

We analyze Equation (3) for the index k with maximum relative sensitivity:

$$\frac{x'_k}{x_k} \left(x_k + \sum_{j \neq k} d_{kj} + 4d_{kk} \right) + \sum_{j \neq k} d_{kj} \frac{x'_j}{x_j} = \delta_{1k}. \quad (23)$$

We first establish the sign of x'_k . Assume for the sake of contradiction that $x'_k < 0$. By the definition of k , we know $\left| \frac{x'_j}{x_j} \right| \leq \left| \frac{x'_k}{x_k} \right|$. It follows that:

$$\sum_{j \neq k} d_{kj} \frac{x'_j}{x_j} \leq \sum_{j \neq k} d_{kj} \left| \frac{x'_k}{x_k} \right|.$$

Substituting this bound into Equation (23) yields:

$$\frac{x'_k}{x_k} \left(x_k + \sum_{j \neq k} d_{kj} + 4d_{kk} \right) + \sum_{j \neq k} d_{kj} \left| \frac{x'_k}{x_k} \right| \geq \delta_{1k}.$$

Since we assumed $x'_k < 0$, we have $\frac{x'_k}{x_k} = - \left| \frac{x'_k}{x_k} \right|$. The terms involving $\sum_{j \neq k} d_{kj}$ cancel out, simplifying the inequality to:

$$\frac{x'_k}{x_k} (x_k + 4d_{kk}) \geq \delta_{1k}.$$

8:20 Amplification at Equilibrium: Limitations and Implementation

The left-hand side is strictly negative (as $x_k > 0$ and $d_{kk} \geq 0$), while the right-hand side is non-negative ($\delta_{1k} \in \{0, 1\}$). This is a contradiction. Therefore, x'_k must be non-negative.

Having established $x'_k \geq 0$, we now derive the upper bound. Rearranging Equation (23):

$$\frac{x'_k}{x_k} \left(x_k + \sum_{j \neq k} d_{kj} + 4d_{kk} \right) = \delta_{1k} - \sum_{j \neq k} d_{kj} \frac{x'_j}{x_j}.$$

Taking the absolute value and applying the triangle inequality:

$$\left| \frac{x'_k}{x_k} \right| \left(x_k + \sum_{j \neq k} d_{kj} + 4d_{kk} \right) \leq \delta_{1k} + \left| \sum_{j \neq k} d_{kj} \frac{x'_j}{x_j} \right| \leq \delta_{1k} + \sum_{j \neq k} d_{kj} \left| \frac{x'_j}{x_j} \right|.$$

Subtracting $\sum_{j \neq k} d_{kj} \left| \frac{x'_j}{x_j} \right|$ from both sides yields:

$$\left| \frac{x'_k}{x_k} \right| (x_k + 4d_{kk}) \leq \delta_{1k}.$$

Since $x_k > 0$ and $d_{kk} \geq 0$, the term $(x_k + 4d_{kk})$ is strictly positive. If $k \neq 1$, then $\delta_{1k} = 0$, which implies $|x'_k/x_k| \leq 0 \implies x'_k = 0$. However, since k is the index of maximum sensitivity, this would imply all sensitivities are zero, which contradicts the existence of the perturbation σ . Therefore, the maximum relative sensitivity must occur for the input species, i.e., $k = 1$.

Substituting $k = 1$ back into the inequality gives $x'_1 (x_1 + 4d_{11}) \leq x_1$. Solving for the sensitivity x'_1 :

$$x'_1 \leq \frac{1}{1 + 4\frac{d_{11}}{x_1}} \leq 1.$$

Thus, the concentration of the free monomer species X_1 cannot increase more than the concentration of the added signal σ . $\blacktriangleleft$

Proof of Theorem 3. To prove the theorem, we must bound the term $\sum_{j < i} |d'_{ji}|$. Applying Equation (6) to each lower-indexed species $j < i$ and isolating the term d'_{ji} gives:

$$|d'_{ji}| \leq \delta_{1j} + \sum_{k < j} |d'_{kj}| - |x'_j| - \sum_{\substack{k > j \\ k \neq i}} |d'_{jk}| - 2|d'_{jj}|.$$

Summing this over all $j < i$:

$$\sum_{j < i} |d'_{ji}| \leq \sum_{j < i} \left(\delta_{1j} + \sum_{k < j} |d'_{kj}| - |x'_j| - \sum_{\substack{k > j \\ k \neq i}} |d'_{jk}| - 2|d'_{jj}| \right). \quad (24)$$

We now simplify the double summations over the interaction terms d . The expression contains two sums: one over $k < j$ (complexes made of lower-indexed species) and one over $k > j$ (complexes made of higher-indexed species).

$$\sum_{j < i} \left(\sum_{k < j} |d'_{kj}| - \sum_{\substack{k > j \\ k \neq i}} |d'_{jk}| \right) = - \sum_{j < i} \sum_{k > i} |d'_{jk}|.$$

This equality holds because for any pair $\{u, v\}$ both less than i , the term $|d'_{uv}|$ appears once positively (when $j = v, k = u$) and once negatively (when $j = u, k = v$). These internal terms cancel out, leaving only the negative terms where $k > i$.

Substituting this simplification back into Equation (24):

$$\sum_{j < i} |d'_{ji}| \leq \sum_{j < i} \left(\delta_{1j} - |x'_j| - 2|d'_{jj}| - \sum_{k > i} |d'_{jk}| \right). \quad (25)$$

Finally, we merge Equation (25) into Equation (6):

$$|x'_i| + \sum_{j > i} |d'_{ij}| + 2|d'_{ii}| \leq \delta_{1i} + \sum_{j < i} \delta_{1j} - \sum_{j < i} \left(|x'_j| + 2|d'_{jj}| + \sum_{k > i} |d'_{jk}| \right).$$

Since $\sum_{j \leq i} \delta_{1j} = 1$ (as the sum includes the only perturbed species X_1), we arrive at the inequality stated in Equation (7). ◀

B Proofs for Section 4 (Thermodynamic Bounds on Amplification)

Proof of Lemma 8. First, we evaluate the relationship between the Active and Resting states. By definition, $\mathbf{x}$ is the absolute global minimizer of the free energy $g(x)$ over the entire feasible concentration space subject to the total mass constraint $Ax = x^0(\sigma)$. The Resting state, defined as $\mathbf{z} = \text{IC}(\mathbf{x}) + \text{argmin}_{Ax=x^0(0)} g(x)$, is also a physically valid configuration in this same feasible space; its decoupled input mass and minimized residual mass exactly sum to the total network mass $x^0(\sigma)$. Because $\mathbf{x}$ is the unconstrained minimizer over the entire valid space, and $\mathbf{z}$ represents just one specific state within that space (specifically, one restricted to have no mixed complexes), it necessarily follows that $g(\mathbf{x}) \leq g(\mathbf{z})$.

Second, we compare the Resting and Drained states. Based on Definition 6 and Definition 7, both states share the exact same isolated input pool, $\text{IC}(\mathbf{x})$. Therefore, any difference in their total free energies depends entirely on their respective residual pools.

The residual pool of the Drained state is given by the projection $\text{RC}(\mathbf{x})$. Because this projection conserves all residual monomers from the Active state, it perfectly satisfies the isolated residual mass constraint $A \cdot \text{RC}(\mathbf{x}) = x^0(0)$. Thus, $\text{RC}(\mathbf{x})$ is a valid concentration vector in the isolated residual space. However, the residual component of the Resting state is explicitly defined as the global free energy minimizer over this exact same isolated space (i.e., $\text{argmin}_{Ax=x^0(0)} g(x)$). Because the Resting state pairs the shared input pool with the thermodynamically optimal residual configuration, whereas the Drained state pairs it with the unoptimized active projection, we must conclude that $g(\mathbf{z}) \leq g(\mathbf{y})$. ◀

Proof of Theorem 10.

Step 1: Expressing $g(\mathbf{y}) - g(\mathbf{x})$. Expanding the free energy function for the Drained and Active states yields:

$$\begin{aligned} g(\mathbf{y}) - g(\mathbf{x}) = & \sum_{v \geq 1} \left[\mathbf{y}_{\mathbf{0},v} (\log \mathbf{y}_{\mathbf{0},v} - 1 + G_{\mathbf{0},v}) - \mathbf{x}_{\mathbf{0},v} (\log \mathbf{x}_{\mathbf{0},v} - 1 + G_{\mathbf{0},v}) \right] \\ & + \sum_{u \neq \mathbf{0}} \left[\mathbf{y}_{u,0} (\log \mathbf{y}_{u,0} - 1 + G_{u,0}) - \sum_v \mathbf{x}_{u,v} (\log \mathbf{x}_{u,v} - 1 + G_{u,v}) \right] \end{aligned}$$

By definition, the Drained state uses RC and IC operators mapping all mixed complexes onto their respective pure residual and pure input pools. Thus, for any pure residual core u (where $u \neq \mathbf{0}$), its concentration in the Drained state is exactly the sum over all its possible input components v , which means $\mathbf{y}_{u,0} = \sum_v \mathbf{x}_{u,v}$.

Since all extracted input cores get mapped onto the pure input pool complexes of the same size, the concentration of pure input complexes of size v (where $v \geq 1$) becomes $\mathbf{y}_{\mathbf{0},v} = \sum_u \mathbf{x}_{u,v}$.

Substituting these distributions into the free energy function $g(x)$ and grouping the terms by their input cores v and residual cores u , the energy difference resolves to:

$$\begin{aligned} g(\mathbf{y}) - g(\mathbf{x}) = & \sum_{v \geq 1} \left[\sum_u \mathbf{x}_{u,v} (\log \sum_u \mathbf{x}_{u,v} - 1 + G_{\mathbf{0},v}) - \mathbf{x}_{\mathbf{0},v} (\log \mathbf{x}_{\mathbf{0},v} - 1 + G_{\mathbf{0},v}) \right] \\ & + \sum_{u \neq \mathbf{0}} \left[\sum_v \mathbf{x}_{u,v} (\log \sum_v \mathbf{x}_{u,v} - 1 + G_{u,0}) - \sum_v \mathbf{x}_{u,v} (\log \mathbf{x}_{u,v} - 1 + G_{u,v}) \right] \end{aligned}$$

By expanding the equations and canceling the constant -1 terms (noting that these terms in the second line perfectly cancel each other out, while the terms of the first line leave the total scalar concentration of all mixed complexes), the energy difference resolves to:

$$\begin{aligned} g(\mathbf{y}) - g(\mathbf{x}) = & \underbrace{\sum_{v \geq 1} \left[\left(\sum_u \mathbf{x}_{u,v} \right) \log \left(\sum_u \mathbf{x}_{u,v} \right) - \mathbf{x}_{\mathbf{0},v} \log \mathbf{x}_{\mathbf{0},v} \right]}_{\mathcal{A}_{\text{inp}}} \\ & + \underbrace{\sum_{u \neq \mathbf{0}} \left[\left(\sum_v \mathbf{x}_{u,v} \right) \log \left(\sum_v \mathbf{x}_{u,v} \right) - \sum_v \mathbf{x}_{u,v} \log \mathbf{x}_{u,v} \right]}_{\mathcal{A}_{\text{res}}} \\ & - \sum_{v \geq 1} \sum_{u \neq \mathbf{0}} [\mathbf{x}_{u,v} (1 + \Delta G_{u,v}^{\text{bind}})] \end{aligned} \quad (26)$$

where the quantities $\mathcal{A}_{\text{inp}}$ and $\mathcal{A}_{\text{res}}$ capture the Shannon-like entropy of mixing for the pure input complexes and the pure residual complexes, respectively.

Step 2: Establishing an Upper Bound for the Free Energy Difference. To establish the upper bound on the free energy difference $g(\mathbf{y}) - g(\mathbf{x})$, we must evaluate the entropic terms $\mathcal{A}_{\text{inp}}$ and $\mathcal{A}_{\text{res}}$. We rely on a fundamental algebraic inequality for logarithms: for any $A, B > 0$, the relation $A \log(1 + B/A) \leq B$ holds true. Consequently, we can bound the entropy of a sum as follows:

$$(A + B) \log(A + B) - A \log A = A \log \left(1 + \frac{B}{A} \right) + B \log(A + B) \leq B + B \log(A + B)$$

We apply this inequality directly to the sums defining $\mathcal{A}_{\text{inp}}$ and $\mathcal{A}_{\text{res}}$. For $\mathcal{A}_{\text{inp}}$, let $A = \mathbf{x}_{\mathbf{0},v}$ and $B = \sum_{u \neq \mathbf{0}} \mathbf{x}_{u,v}$, such that $A + B = \mathbf{y}_{\mathbf{0},v}$. This yields:

$$\mathcal{A}_{\text{inp}} \leq \sum_{v \geq 1} \left[\left(\sum_{u \neq \mathbf{0}} \mathbf{x}_{u,v} \right) (\log \mathbf{y}_{\mathbf{0},v} + 1) \right] = \sum_{u \neq \mathbf{0}} \sum_{v \geq 1} [\mathbf{x}_{u,v} (\log \mathbf{y}_{\mathbf{0},v} + 1)]$$

Similarly, for $\mathcal{A}_{\text{res}}$, we set $A = \mathbf{x}_{u,0}$ and $B = \sum_{v \geq 1} \mathbf{x}_{u,v}$, yielding $A + B = \mathbf{y}_{u,0}$. Expanding the subtracted sum over v , we find:

$$\begin{aligned} \mathcal{A}_{\text{res}} & \leq \sum_{u \neq \mathbf{0}} \left[\left(\sum_{v \geq 1} \mathbf{x}_{u,v} \right) (\log \mathbf{y}_{u,0} + 1) - \sum_{v \geq 1} \mathbf{x}_{u,v} \log \mathbf{x}_{u,v} \right] \\ & = \sum_{u \neq \mathbf{0}} \sum_{v \geq 1} [\mathbf{x}_{u,v} (\log \mathbf{y}_{u,0} + 1 - \log \mathbf{x}_{u,v})] \end{aligned}$$

Substituting these bounded entropic quantities back into Equation (26), the total subtracted scalar concentration of the mixed complexes (the isolated sum in the third line) perfectly absorbs one of the +1 constants generated by our inequalities. The expression simplifies into a unified single summation strictly over the mixed complexes:

$$\begin{aligned} g(\mathbf{y}) - g(\mathbf{x}) &\leq \sum_{u \neq \mathbf{0}} \sum_{v \geq 1} \mathbf{x}_{u,v} \left[\log \mathbf{y}_{\mathbf{0},v} + 1 + \log \mathbf{y}_{u,0} + 1 - \log \mathbf{x}_{u,v} - 1 - \Delta G_{u,v}^{\text{bind}} \right] \\ &= \sum_{u \neq \mathbf{0}} \sum_{v \geq 1} \mathbf{x}_{u,v} \left[\log \left(\frac{\mathbf{y}_{\mathbf{0},v} \mathbf{y}_{u,0}}{\mathbf{x}_{u,v}} \right) + 1 - \Delta G_{u,v}^{\text{bind}} \right] \end{aligned}$$

Because each term in the preceding expression is multiplied by $\mathbf{x}_{u,v}$, any complex (u, v) with a zero Active-state concentration vanishes and can be excluded. Consequently, instead of summing over all $u \neq \mathbf{0}$, we can restrict the summation to $u \in \mathcal{U}$. Recall that this set is defined as: $\mathcal{U} = \{u \neq \mathbf{0} \mid \exists v \geq 1: \mathbf{x}_{u,v} > 0\}$.

$$g(\mathbf{y}) - g(\mathbf{x}) \leq \sum_{u \in \mathcal{U}} \sum_{v \geq 1} \mathbf{x}_{u,v} \left[\log \left(\frac{\mathbf{y}_{\mathbf{0},v} \mathbf{y}_{u,0}}{\mathbf{x}_{u,v}} \right) + 1 - \Delta G_{u,v}^{\text{bind}} \right] \quad (27)$$

► **Remark 17.** Before applying simplifications and decoupling steps to derive a closed-form tractable bound, evaluating the component-level dependencies of the preceding inequality may be appropriate to generate design strategies and effective amplifiers. One such heuristic is apparent: maximize the concentration of unbound “input partners” ($\mathbf{x}_{u,0}$) for any residual core u where the corresponding mixed complex ($\mathbf{x}_{u,v}$) is significant, which increases the drained concentration $\mathbf{y}_{u,0}$ and mitigates the negative logarithmic penalty associated with complex formation.

The formulation in Equation (27) casts the free energy transition strictly as an expectation over the mixed complexes. To globally decouple the specific complex concentrations, we apply the Log-Sum Inequality: $\sum a_i \log(b_i/a_i) \leq (\sum a_i) \log(\sum b_i / \sum a_i)$.

For the numerator sum $\sum b_i$, we evaluate $\sum_{u \in \mathcal{U}} \sum_{v \geq 1} \mathbf{y}_{\mathbf{0},v} \mathbf{y}_{u,0}$. Because all concentration terms are non-negative, we can create a separable upper bound, $\sum_{v \geq 1} \mathbf{y}_{\mathbf{0},v} \cdot \sum_{u \in \mathcal{U}} \mathbf{y}_{u,0}$. Crucially, because the Drained state is defined as sums over the Active state complexes, summing these over their respective domains exactly partitions them into the pure and mixed Active state pools:

$$\begin{aligned} \sum_{v \geq 1} \mathbf{y}_{\mathbf{0},v} &= \underbrace{\sum_{v \geq 1} \mathbf{x}_{\mathbf{0},v}}_{X_{\text{inp}}} + \underbrace{\sum_{u \in \mathcal{U}} \sum_{v \geq 1} \mathbf{x}_{u,v}}_{X_{\text{mix}}} & \sum_{u \in \mathcal{U}} \mathbf{y}_{u,0} &= \underbrace{\sum_{u \in \mathcal{U}} \mathbf{x}_{u,0}}_{X_{\text{res}}} + \underbrace{\sum_{u \in \mathcal{U}} \sum_{v \geq 1} \mathbf{x}_{u,v}}_{X_{\text{mix}}} \end{aligned}$$

Substituting this separated numerator back into the Log-Sum framework, the upper bound resolves to a thermodynamic evaluation dependent solely on the Active state partitioning:

$$g(\mathbf{y}) - g(\mathbf{x}) \leq X_{\text{mix}} \left[\log \left(\frac{(X_{\text{inp}} + X_{\text{mix}}) \cdot (X_{\text{res}} + X_{\text{mix}})}{X_{\text{mix}}} \right) + 1 - \Delta G_{\text{stg}}^{\text{inp}} \right]$$

To make the expression easier to work with, we observe that $X_{\text{inp}} + X_{\text{mix}} \leq \sigma$, so:

$$g(\mathbf{y}) - g(\mathbf{x}) \leq \underbrace{X_{\text{mix}} \left[\log \left(\frac{\sigma (X_{\text{res}} + X_{\text{mix}})}{X_{\text{mix}}} \right) + 1 - \Delta G_{\text{stg}}^{\text{inp}} \right]}_{h(X_{\text{mix}})}$$

8:24 Amplification at Equilibrium: Limitations and Implementation

To find the global maximum of this bound, we treat $(X_{\text{res}} + X_{\text{mix}})$ as a fixed parameter. Under this condition, $h(X_{\text{mix}})$ is a strictly concave function with respect to X_{mix} . By setting the derivative to zero, the unconstrained maximum of h occurs at $X_{\text{mix}} = \sigma (X_{\text{res}} + X_{\text{mix}}) e^{-\Delta G_{\text{stg}}^{\text{inp}}} = \sigma e^{\psi_{\text{inp}}}$.

Because X_{mix} is physically constrained by the total input concentration ($X_{\text{mix}} \leq \sigma$), we must evaluate the boundary conditions. If $\psi_{\text{inp}} \leq 0$, the optimal value lies within the valid physical domain, yielding a maximum of $\sigma e^{\psi_{\text{inp}}}$. If $\psi_{\text{inp}} > 0$, the unconstrained optimal value exceeds σ ; because h is concave, the constrained maximum must occur at the boundary $X_{\text{mix}} = \sigma$, which evaluates to $\sigma(\psi_{\text{inp}} + 1)$. Consequently, Equation (16) is proven:

$$g(\mathbf{y}) - g(\mathbf{x}) \leq \begin{cases} \sigma e^{\psi_{\text{inp}}} & \text{if } \psi_{\text{inp}} \leq 0 \\ \sigma (\psi_{\text{inp}} + 1) & \text{if } \psi_{\text{inp}} > 0 \end{cases} \quad \blacktriangleleft$$

Thermodynamic Binding Networks Capture Equilibrium Monomer-Polymer Thermodynamics

And Kaan Ata Yilmaz ✉ 

Electrical and Computer Engineering, University of Texas at Austin, TX, USA

David Soloveichik ✉ 

Electrical and Computer Engineering, University of Texas at Austin, TX, USA

Austin Luchsinger ✉ 

Computer Science, University of Texas Rio Grande Valley, Edinburg, TX, USA

Abstract

Thermodynamic Binding Networks (TBNs) are a minimal model for molecular thermodynamics. Despite their simplicity, the stable behavior of these systems can be unexpectedly nontrivial. To better understand the expressive power of the model, we compare TBNs with Thermodynamic Affinity Networks (TANs), a partition-scoring formalism in which polymer-level favorability is specified directly, rather than derived from domain-level interactions. We show that these two models are thermodynamically equivalent at the level of stable configurations: every TBN can be represented directly by a TAN, and every TAN can be indirectly realized by a TBN whose stable configurations map exactly to the stable configurations of the TAN. The key idea in the latter direction is to use auxiliary monomers that allow the system to convert polymer-level free-energy rewards specified by the TAN into entropic gains from separate complexes in the TBN. This correspondence shows that the local rules of TBNs do not limit their equilibrium expressive power nearly as much as one might expect. In this minimum-free-energy sense of equilibrium, TBNs capture the coarse-grained monomer-polymer thermodynamics formalized by TANs.

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases Thermodynamic Binding Networks, Thermodynamic Affinity Networks, Equilibrium Thermodynamics

Digital Object Identifier 10.4230/LIPIcs.DNA.32.9

Funding Schmidt Sciences Polymath Award to D.S., DOE grant DE-SC0024248, NSF SemiSynBio III: GOALI grant 2227578.

1 Introduction

Thermodynamic Binding Networks (TBNs) were introduced as a deliberately simplified model of molecular thermodynamics [7]. In this model, monomers are multisets of domain types, polymers are assemblies of monomers (which may or may not be joined by pairwise complementary domain bindings), and configurations are collections of polymers. Each configuration is assigned an energy based on the number of bonds (enthalpy) and the number of polymers it contains (entropy), and the configurations with the lowest (most favorable) energy are said to be stable. See Figure 1 for an example TBN system.

At first glance, the expressive power of TBNs seems limited. The model is intentionally simplified: it lacks several mechanisms that have proved powerful in other models, including geometry [9, 19], non-complementary domain interactions [1, 12], and higher-order regulatory effects such as allostery [13, 16]. One might therefore expect TBNs to admit only a narrow and fairly transparent class of thermodynamic behaviors. However, despite this simplicity, TBNs have proven surprisingly difficult to analyze, and establishing formal properties of their behavior is often quite subtle [3, 4, 10, 14]. This raises a natural question: to what extent can TBNs capture arbitrary monomer-polymer thermodynamics?



© And Kaan Ata Yilmaz, David Soloveichik, and Austin Luchsinger;
licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 9; pp. 9:1–9:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

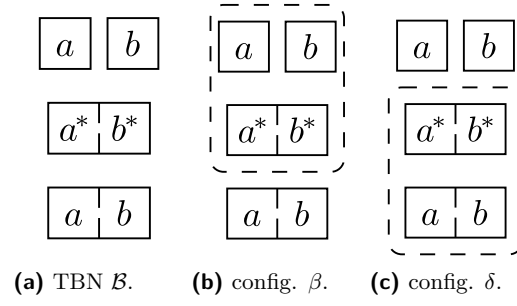
To make this question more precise, we compare TBNs with a more abstract thermodynamic formalism: Thermodynamic Affinity Networks (TANs). Equilibrium chemical thermodynamics is often understood at the level of macrostates that record only coarse-grained information about the system, such as which molecular complexes are present and in what counts, rather than the full microscopic arrangement of the molecules. The thermodynamic favorability of such a macrostate is determined by the total statistical weight of the microstates that realize it. This statistical weight combines energetic effects (such as favorable molecular interactions within complexes) with entropic effects (such as translational freedom and combinatorial multiplicity). TANs formalize this coarse-grained macrostate viewpoint for finite monomer-polymer systems.

A TAN consists of a finite multiset of monomer types M along with an affinity function $\mathcal{F} : \mathcal{P}(M) \rightarrow \mathbb{Q}_{\geq 0}$ that assigns a nonnegative value to every possible polymer. A TAN configuration is simply a partition of the monomers into polymers, and a configuration’s thermodynamic preference is determined by the total affinity of those polymers as well as the number of polymers in the partition. This abstraction removes the domain-level mechanics entirely and instead treats polymer favorability as a primitive. TANs should be viewed as a finite formal language for monomer-polymer macrostate preferences rather than as a new model for chemical thermodynamics. Mathematically, TANs are closely related to other partition-scoring formalisms such as combinatorial auctions and coalition structure generation [5, 15].

The contrast between TBNs and TANs becomes apparent even in the simplest systems. As an example, consider a TAN system with three monomers m_1, m_2 , and m_3 where the polymer $\{m_1, m_2, m_3\}$ has positive affinity (i.e., $\mathcal{F}(\{m_1, m_2, m_3\}) > 0$), and all pairwise combinations of the monomers have zero affinity. This kind of polymer-level preference is natural in the TAN setting; however, it is not clear how to achieve this behavior in a TBN since the enthalpic contribution of a polymer comes from the pairwise complementary bonds within it. In that setting, any bond present in the ternary complex would already be a bond between some pair of monomers and would therefore create pairwise affinity.

Although a *direct* TBN implementation of the previous example is not possible, the main result of this work shows that TBNs can *indirectly* capture this behavior, as well as any other behavior described by an arbitrary affinity function. To accomplish this, our key technique takes the favorability assigned directly to polymers in a TAN and encodes it indirectly in a TBN through entropy and particle counts, rather than explicit bonds. The idea is that a TBN realization of this example consists of three large complexes representing the TAN monomers. When those complexes merge to form a single polymer, they release a precise number of auxiliary “helper” monomers. The construction is engineered so that the entropy increase from these released monomers lowers the free energy of the TBN configuration by an amount that is consistent with the TAN configuration (e.g., accounting for the affinity of the ternary complex).

Ultimately, we show that every TAN can be represented by a thermodynamically equivalent TBN, and conversely that every TBN has an equivalent TAN. This surprising equivalence shows that TANs do not extend the range of thermodynamic behaviors realizable by TBNs; rather, they provide a more direct language for specifying those behaviors. Since TANs formalize a highly general class of coarse-grained monomer-polymer thermodynamics, this equivalence suggests that TBNs capture this class as well. We note that throughout the paper, “equilibrium” is meant in the minimum-free-energy sense: our realization definition compares the stable configurations of each system, rather than full Boltzmann equilibrium distributions. With this convention, and given the natural implementation of TBNs with DNA strands, our result supports the view that DNA-based binding is a broadly expressive substrate for TAN-style monomer-polymer thermodynamics.



■ **Figure 1** (a) An example TBN $\mathcal{B}$ with $T = \{\{a\}, \{b\}, \{a, b\}, \{a^*, b^*\}\}$, (b) a saturated configuration β of $\mathcal{B}$, and (c) a stable configuration δ of $\mathcal{B}$.

Paper Organization

In Section 2, we introduce the necessary definitions of TBNs, TANs, and our notions of thermodynamic realization and equivalence. In Section 3, we present an initial result establishing that any given TBN can be directly realized by a TAN. Section 4 introduces a generalization of the grid gate from [3], which serves as a key module in our construction of a TBN that is thermodynamically equivalent to a given TAN. In Section 5, we present the main construction of this work: a TBN that thermodynamically realizes a given TAN in the large-volume regime. Section 6 extends the construction to the general TAN energy function by incorporating combinatorial entropy. Finally, Section 7 concludes with a discussion of implications and directions for future work.

2 Preliminaries

We first provide definitions for the two models, Thermodynamic Affinity Networks and Thermodynamic Binding Networks, and then provide our definition of thermodynamic equivalence.

2.1 Thermodynamic Binding Network

A *Thermodynamic Binding Network* $\mathcal{B} = (D, T)$ is a tuple where:

- D is a finite set of domain types,
- T is a finite multiset of monomer types over $D \cup D^*$

A *domain type* is a formal symbol. D^* denotes the set of complementary domain types for D . We often refer to domain types from D as *unstarred domains* and domain types from D^* as *starred domains*. A *monomer type* is a multiset of domain types. We call an instance of a domain type a *domain* and an instance of a monomer type a *monomer*. A *configuration* β of a TBN is a partition of T into parts, which we call *polymers*. A *bond* is a pairwise matching of complementary domains within a polymer. Note that a polymer is simply a block of the configuration partition, and has no requirement on the formation of bonds between its constituent monomers. Furthermore, bonds are not explicitly specified in a configuration; we just intuitively think of polymers as having a maximal matching¹.

¹ We use the term matching in the graph theory sense. Indeed, if we created a graph of the polymer using domains as vertices with edges between complementary domains, a matching of that graph would yield a set of bonds in the polymer.

The *entropy* $S(\beta)$ of configuration β is the number of polymers in β . The *enthalpy* $H(\beta)$ of configuration β is the number of bonds in β . We use the convention that TBNs are *star-limiting*: for each domain type, there are always at least as many instances of the unstarred type as the starred type among all monomers. A starred (unstarred) domain within a polymer is said to be *exposed* if, among the monomers in that polymer, the number of instances of the domain type exceeds the number of unstarred (starred) instances of the complementary domain type. A polymer is *self-saturated* if it contains no exposed starred domains. A self-saturated polymer is *splittable*, if it can be split into two self-saturated polymers and *unsplittable*, otherwise. For a star-limiting TBN, a configuration is *saturated* if every polymer in the configuration is self-saturated. Intuitively, this means that no polymer contains an unmatched starred domain and equivalently, all possible bonds are formed.

The *energy* of β is $E(\beta) = -wH(\beta) - S(\beta)$ where w is an enthalpy weighting parameter. The configurations that minimize E are said to be *stable*. As is common when discussing TBNs, we operate under a regime where w is high enough for enthalpy to be considered infinitely more favorable than entropy. The following lemma identifies a key structural constraint of stable configurations for $w > 1$ and shows that stability is insensitive to the exact value of w .

► **Lemma 1.** *For a given TBN $\mathcal{B}$, if $w > 1$, a stable configuration of $\mathcal{B}$ is an entropy maximizing saturated configuration. Furthermore, the set of stable configurations of $\mathcal{B}$ is the same for any $w > 1$.*

Proof. Suppose β is not a saturated configuration. Then there exist two polymers P_1 and P_2 in β such that P_1 contains an exposed starred domain, without loss of generality a^* , and P_2 contains an exposed unstarred domain a . Consider a configuration β' obtained from β by replacing P_1 and P_2 with their union $P_1 \uplus P_2$, while leaving all other polymers unchanged. In this case, we have $H(\beta') \geq H(\beta) + 1$ and $S(\beta') = S(\beta) - 1$. Therefore, $E(\beta') - E(\beta) \leq -(w - 1)$. As $w > 1$, β cannot be stable. Consequently, for $w > 1$, every stable configuration is saturated, i.e., it realizes all possible bonds.

Among the saturated configurations, stability further favors configurations that maximize entropy, namely those with the largest number of free complexes. Moreover, since all saturated configurations achieve the same maximum enthalpy (i.e., number of bonds), which is fixed for a given TBN, the set of stable configurations is identical for all values of $w > 1$. ◀

Note that if there is a splittable polymer in a saturated configuration, the entropy of the configuration can be increased by 1 by splitting it. Hence, in a stable configuration of a TBN, there cannot be any splittable polymers.

2.2 Thermodynamic Affinity Network

The TAN model provides a more abstract formalism for thermodynamic networks.

A *Thermodynamic Affinity Network* $\mathcal{A} = (M, \mathcal{F}, u, v)$ is a 4-tuple consisting of:

- M , a finite multiset of monomer types,
- $\mathcal{F}$, an affinity function (defined below),
- $u, v \in \mathbb{Q}_{\geq 0}$, enthalpy and entropy weight parameters.²

A *monomer type* is a formal symbol, and a *polymer type* is a multiset of monomer types. We define $\mathcal{P}(M) = \{N \mid N \text{ is a nonempty sub-multiset of } M\}$, i.e., the set of all polymer types over M . The affinity function $\mathcal{F} : \mathcal{P}(M) \rightarrow \mathbb{Q}_{\geq 0}$ assigns to each polymer

² Intuitively, one may view $\frac{v}{u}$ as the temperature of the system.

its enthalpic contribution upon formation. As in the TBN setting, a *configuration* α of a TAN is a partition of M , and may be viewed as a multiset of polymers. We write $\text{supp}(\alpha) = \{P_1, P_2, \dots, P_n\}$ for the set of distinct polymers appearing in α , and let $\underline{\alpha}(s)$ denote the multiplicity of $P_s \in \text{supp}(\alpha)$. Throughout the paper, we alternate between indexing directly over α (treating identical polymers as distinct copies, as in Section 5) and indexing over $\text{supp}(\alpha)$ (as here and in Section 6).

The energy of a TAN configuration α is defined by

$$E_{\text{TAN}}(\alpha) = - \sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s) w \log(V) \mathcal{F}(P_s) - \sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s) \log(V) + \sum_{P_s \in \text{supp}(\alpha)} \sum_{j=1}^{\underline{\alpha}(s)} \log j,$$

where V is the volume and $w \in \mathbb{Q}$ is a constant. A derivation from first principles is provided in Appendix A. We approximate $\log V$ to five digits, making it rational, and define $u = w \log V$ and $v = \log V$, so that $u, v \in \mathbb{Q}$. The entropy $S(\alpha)$ is the number of polymers in α , and the enthalpy $H(\alpha)$ is $\sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s) \mathcal{F}(P_s)$. With these definitions, the energy can be written compactly as

$$E(\alpha) = -uH(\alpha) - vS(\alpha) + \sum_{P_s \in \text{supp}(\alpha)} \sum_{j=1}^{\underline{\alpha}(s)} \log j.$$

The final term, arising from indistinguishability of polymers (see Appendix A), is referred to as the *combinatorial entropy*. A configuration α is *stable* if it minimizes $E(\alpha)$. Notice that, since the TAN model has no notion of bonds, there is no need to define saturated configurations.

2.3 Configuration Mappings and Realizations

To compare the expressive power of these thermodynamic models, we relate their state spaces via mappings. Intuitively, such mappings interpret configurations of one system as configurations of another, allowing us to formalize when two systems exhibit equivalent behavior. Two notions of equivalence are natural in this setting. The first is purely thermodynamic: under a given mapping, two systems have equivalent equilibria (i.e., roughly the same stable configurations). The second is kinetic: under a given mapping, two systems have equivalent dynamics³. In this work, we focus only on thermodynamic equivalence.

Here we define what it means for a Thermodynamic Binding Network (TBN) to be thermodynamically equivalent to a Thermodynamic Affinity Network (TAN) by way of configuration mappings that we refer to as realizations. Throughout this subsection, let X denote the *realizing* system and Y denote the *realized* system. The definitions in this subsection are model-independent: both the realizing system and the realized system may be either a TAN or a TBN.

Mappings

Our main object of interest is a configuration-level map (that we call R_{conf}) which interprets a configuration of the realizing system X as a corresponding configuration of the realized system Y . Rather than allowing an arbitrary mapping between configurations, we ensure R_{conf} is “reasonable” by building up from more fundamental mappings between monomer

³ This kinetic equivalence is well studied as simulation/bisimulation [2, 8, 11]

types (R_{type}) and polymers (R_{poly}). Informally, R_{type} specifies which monomer types of Y (if any) each monomer type from X corresponds to. R_{poly} naturally extends R_{type} to map polymers, and R_{conf} extends R_{poly} in the same way.

Both models have a finite multiset of monomer types and configurations are given by partitions of those multisets into polymers. Let M_X and M_Y denote the multisets of monomer types of X and Y , respectively, and let S_X and S_Y denote their underlying sets of monomer types. Let $\text{Conf}(X)$ and $\text{Conf}(Y)$ denote the sets of configurations of X and Y , and let $\text{Stab}(X) \subseteq \text{Conf}(X)$ and $\text{Stab}(Y) \subseteq \text{Conf}(Y)$ denote their stable configurations.

A *type map* is a partial function $R_{\text{type}} : S_X \rightarrow S_Y$ that maps monomer types from system X to system Y . We say R_{type} is *valid* if for every monomer type $y \in S_Y$, there exists exactly one monomer type $x \in S_X$ such that $R_{\text{type}}(x) = y$ and the multiplicity of x in M_X equals the multiplicity of y in M_Y . If $R_{\text{type}}(x)$ is undefined, we say $R_{\text{type}}(x) = \emptyset$. Unmapped monomer types represent auxiliary components of the realizing system X that do not correspond to monomers of the realized system Y . We can naturally extend R_{type} to map polymers and configurations as well. With P denoting a multiset of monomer types (i.e., a polymer) of X and for a valid R_{type} , define $R_{\text{poly}}(P) = \{R_{\text{type}}(x) \mid x \in P \text{ and } x \in \text{dom}(R_{\text{type}})\}$. And similarly, with α denoting a multiset of polymers (i.e., a configuration) of X , define $R_{\text{conf}}(\alpha) = \{R_{\text{poly}}(P) \mid P \in \alpha \text{ and } R_{\text{poly}}(P) \neq \emptyset\}$. In other words, R_{poly} ignores unmapped monomers and R_{conf} ignores polymers of unmapped monomers. Note that multiple configurations of X may be mapped by R_{conf} to the same configuration of Y under this definition.

Thermodynamic Realizations

We say system X *thermodynamically realizes* system Y under a valid type map R_{type} if the following conditions hold:

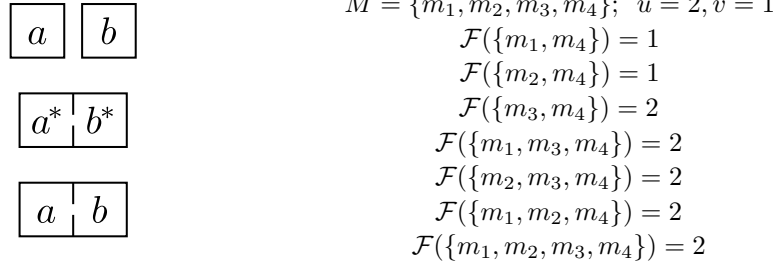
1. For every $\alpha \in \text{Conf}(Y)$, there exists a $\beta \in \text{Conf}(X)$ such that $R_{\text{conf}}(\beta) = \alpha$,
2. if $\omega \in \text{Stab}(X)$ then $R_{\text{conf}}(\omega) \in \text{Stab}(Y)$, and
3. for every $\rho \in \text{Stab}(Y)$, there exists a $\varphi \in \text{Stab}(X)$ such that $R_{\text{conf}}(\varphi) = \rho$.

The first condition says that every possible configuration of the realized system has at least one representation in the realizing system. The second says that stable configurations of the realizing system must map to stable configurations of the realized system. The third says that every stable configuration of the realized system is represented by at least one stable configuration of the realizing system.

A valid realization is a *direct realization* if R_{type} is total (i.e., every monomer type of the realizing system maps to a monomer type of the realized system), and it is an *indirect realization* otherwise. Thus, indirect realizations permit extra “helper” monomer types in the realizing system.

We say two systems are *thermodynamically equivalent (under realization)* if each one thermodynamically realizes the other, possibly under different realization maps. Again, this notion is completely based on comparing the stable configurations of the systems under a configuration map. It does not, by itself, require the two systems to have the same kinetic pathways, transition barriers, or local minima.

In Section 3, we instantiate these definitions with a TAN as the realizing system and a TBN as the realized system. In Sections 5 and 6 we reverse this direction and consider a TBN as the realizing system and a TAN as the realized system.



■ **Figure 2** An explicit example of a TBN $\mathcal{B}$ (left) and a TAN $\mathcal{A}$ (right) that thermodynamically realizes it. For brevity, all instances where $\mathcal{F}(p) = 0$ are omitted.

3 Realizing TBNs with TANs

The first (rather straightforward) result is that for any TBN, there exists a TAN that directly realizes it. Figure 2 shows an explicit example of a TAN that thermodynamically realizes a TBN.

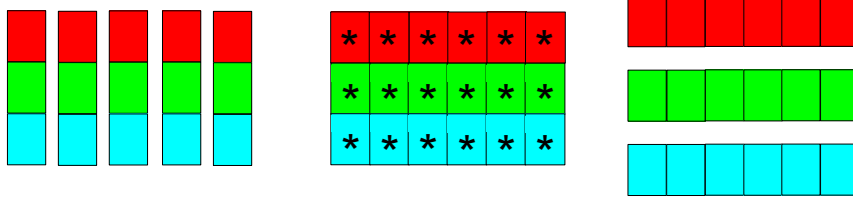
► **Lemma 2.** *For any TBN $\mathcal{B}$, there exists a TAN $\mathcal{A}$ such that $\mathcal{A}$ directly realizes $\mathcal{B}$.*

Proof. Given $\mathcal{B} = (D, T)$, construct $\mathcal{A} = (M, \mathcal{F}, u, v)$ as follows: For every monomer type $t \in T$, create a monomer type $m \in M$ and let $R_{\text{type}}(m) = t$, and set the multiplicity of each monomer type m to match that of $R_{\text{type}}(m) = t$. This implies that the first condition of thermodynamic realization is satisfied. Furthermore, since there are no unmapped monomers in $\mathcal{A}$, this is a direct realization and it follows that for any TBN configuration β there is exactly one TAN configuration α such that $R_{\text{conf}}(\alpha) = \beta$. As R_{conf} is defined for all TAN configurations, this implies that R_{conf} is a bijection. Now, for every possible polymer type P in $\mathcal{P}(M)$, let $\mathcal{F}(P)$ be the number of bonds in $R_{\text{poly}}(P)$ if $R_{\text{poly}}(P) \neq \emptyset$, and 0 otherwise. By Lemma 1, a stable configuration of $\mathcal{B}$ minimizes the energy function $E_{\text{TBN}}(\beta') = -wH_{\text{TBN}}(\beta') - S(\beta')$, for any value $w > 1$. Then, it suffices to set $u = 2C_{\text{Vol}}$ and $v = C_{\text{Vol}}$ (effectively $w = 2$), where $C_{\text{Vol}} = \log V$ is sufficiently large such that the combinatorial entropy term is negligible. The choice of $\mathcal{F}$ implies $H_{\text{TAN}}(\alpha) = H_{\text{TBN}}(R_{\text{conf}}(\alpha))$. Moreover, $S(\alpha) = S(R_{\text{conf}}(\alpha))$. Then $E_{\text{TAN}}(\alpha) = C_{\text{Vol}}E_{\text{TBN}}(R_{\text{conf}}(\alpha))$. As R_{conf} is a bijection, α minimizes E_{TAN} if and only if $R_{\text{conf}}(\alpha)$ minimizes E_{TBN} . Hence, $\mathcal{A}$ directly realizes $\mathcal{B}$. ◀

4 Rectangular Grid Gates

The work of [3] introduced a very useful module known as the grid gate. Our construction in Section 5 will employ a generalized version of this module that is rectangular, rather than square (Figure 3). Following the definition of square grid gates in [3], the rectangular grid gate B_{hw} with parameters h (for height) and w (for width) includes the following monomer types: “horizontal” $\mathcal{H}_i = (x_i)^w$ for $i \in \{1, 2, \dots, h\}$, where $(x_i)^w$ denotes w -many copies of the binding site x_i , “grid base” $G = \{(x_i^*)^w\}_{i=1}^h$, and w -many copies of “verticals” $\mathcal{V} = \{x_i\}_{i=1}^h$. Figure 3 shows an example of a rectangular grid gate B_{35} . In the figure, we use colors to indicate domains and the star sign to indicate the starred versions of those domains.

Note that both the set of horizontal monomers, $\{\mathcal{H}_i\}_{i=1}^h$, and the w copies of vertical monomers, $(\mathcal{V})^w$, can saturate the grid base individually. As each pair of horizontal and vertical monomers of a rectangular grid gate shares exactly one domain, as in the square grid gate, we can immediately generalize Lemma 24 of [3] as:



■ **Figure 3** An example of a rectangular grid gate with parameters $h = 3$, $w = 5$.

► **Lemma 3.** *In a saturated configuration of B_{hw} , a polymer containing the grid base G also contains $\{\mathcal{H}_i\}_{i=1}^h$ or $(\mathcal{V})^w$ as a subset.*

5 Realizing TANs with TBNs: Large-Volume Regime

In Sections 5 and 6, we present the main constructions of this paper, showing how to construct a TBN $\mathcal{B}$ that indirectly realizes a given TAN $\mathcal{A} = (M, \mathcal{F}, u, v)$. We first consider the regime in which the volume parameter V is sufficiently large so that the combinatorial entropy contribution becomes negligible. In this case, the TAN energy function reduces to $E_{\text{TAN}}(\alpha) = -uH(\alpha) - vS(\alpha)$. Without loss of generality, we assume $u, v \in \mathbb{N}$ and $\mathcal{F} : \mathcal{P}(M) \rightarrow \mathbb{Z}_{\geq 0}$. The construction presented in Section 6 for the general energy function setting is built directly upon the construction introduced in this section. For the remainder of this section, we fix $k = u \max_P \mathcal{F}(P) + 2|M| + v + 1$.

Construction sketch: At a high level, polymers of $\mathcal{A}$ are managed by rectangular grid gates in $\mathcal{B}$ whose horizontal monomers map to monomers in $\mathcal{A}$. Rectangular grid gates in the vertical configuration map to nothing in the TAN. Rectangular grid gates in the horizontal configuration represent the formation of that polymer in the TAN. The number of vertical monomers freed by placing a rectangular grid gate in the horizontal configuration corresponds to the energy reward for forming that polymer in the TAN. The formal construction is detailed below.

Mapped Monomers. For each monomer type m_i in $\mathcal{A}$, we introduce a distinct domain type in $\mathcal{B}$ associated with m_i , which we denote without loss of generality by m_i , and define a corresponding monomer type consisting of k copies of this domain. The multiplicity of each such monomer type in $\mathcal{B}$ is set to match the multiplicity of m_i in $\mathcal{A}$. We refer to these as *mapped monomers*, since the domain of R_{type} consists precisely of their types. Note that R_{type} is valid and the first condition of thermodynamic realization is satisfied. We refer to a mapped monomer consisting of k copies of domain m_i as a *mapped monomer encoding m_i* .

Polymer Grid Gate. Inspired by rectangular grid gates, for each polymer $P \in \mathcal{P}(M)$ that can be formed in $\mathcal{A}$, we include $|M|$ -many *polymer grid gates* (or simply, *polymer gates*) in $\mathcal{B}$. Although these gates share the same overall structure, the monomers in different polymer gates corresponding to the same polymer either share only a subset of domains or are entirely disjoint; this distinction is described below. An example of a polymer gate for $P = \{m_1, m_1, m_2\}$ is shown in Figure 4, where $|P| = 3$. Each polymer gate consists of three groups of monomers:

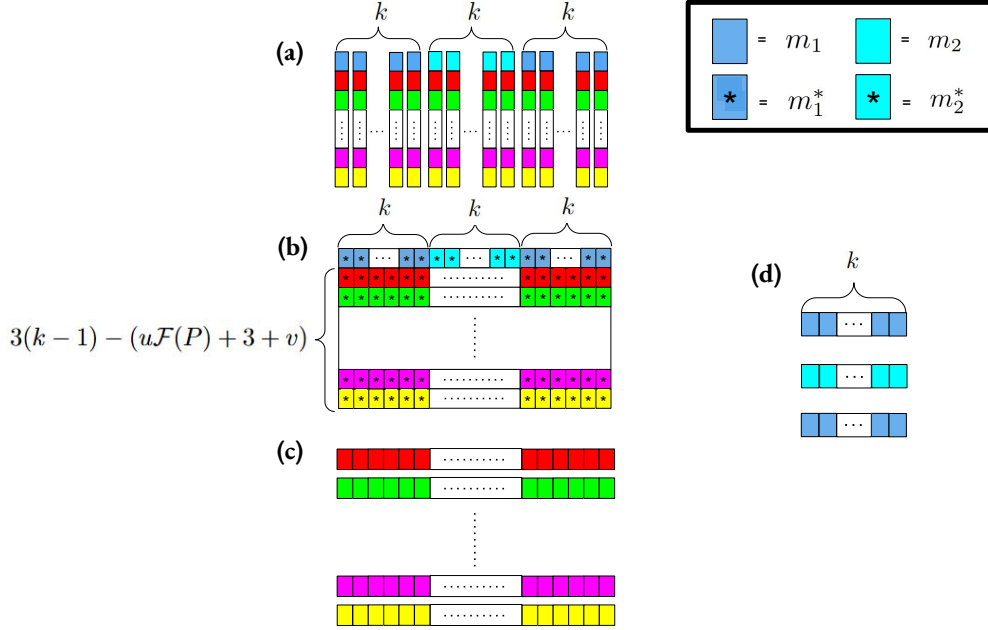
1. Polymer Grid Base. The polymer grid base, or *polymer base* for short, is the monomer that serves as the mechanism that forms polymers in the TBN corresponding to target polymers in the TAN. For a TAN polymer P , a polymer base G_P , included in one of the $|M|$ -many polymer gates, is constructed as follows. For each monomer $m_i \in P$, we include k copies of the domain m_i^* so that the mapped monomer encoding m_i can bind to the base; we refer to this portion as the *mapped part*. In addition, the polymer base includes the grid base of a rectangular grid gate of width $k|P|$ and height $(k-1)|P| - (u\mathcal{F}(P) + |P| + v)$. The domains used in this grid base are disjoint from any m_i^* domains in $\mathcal{B}$. We denote the height of the polymer base, i.e., the height of the underlying rectangular grid base, by $h(G_P)$. As illustrated in Figure 4(b), the mapped part (forming the first row) and the rectangular grid base (extending from the red to the yellow rows) can be arranged so that the polymer base forms a single rectangular structure. Unlike the mapped part, which may share domains with other polymer bases corresponding to TAN polymers that share monomers with P , the domains in the grid base are unique to G_P . Note that even two polymer bases encoding the same polymer have disjoint sets of domains in their rectangular grid bases. When necessary, we use G_P^j to denote the j -th polymer base where $j \leq |M|$.

2. Base-Specific Horizontals. Each polymer gate includes a collection of horizontal monomers of length $k|P|$ that can bind to the rows of the rectangular grid base in G_P . Since the domains of this grid base are specific to the polymer base, these horizontal monomers are likewise base-specific, and we refer to them as *base-specific horizontals*. Figure 4(c) illustrates the base-specific horizontal monomers for a polymer gate.

3. Base-Specific Verticals. The final group consists of vertical monomers that bind to the columns of the rectangular representation of G_P . More precisely, each vertical monomer contains domains complementary to those in a single column of the grid base, along with exactly one additional domain that binds to the mapped part. For each TAN monomer m_i encoded in the polymer base, there are exactly k vertical monomers containing the domain m_i ; in total, there are $k|P|$ base-specific vertical monomers. Figure 4(a) illustrates these base-specific vertical monomers. Note that each vertical monomer contains exactly one domain, its top domain in Figure 4 (a), that binds to the mapped part of the polymer base.

Note that $\mathcal{B}$ is star-limiting. Moreover, since each monomer in $\mathcal{B}$ consists either entirely of starred domains (a *star monomer*) or entirely of unstarred domains (an *unstar monomer*), it follows that in a self-saturated polymer, every starred domain of a star monomer must form a bond with an unstarred domain of an unstar monomer. As each saturated configuration consists of self-saturated polymers, a star monomer (necessarily a polymer base G_P^j) cannot be free in such a configuration. Consequently, the entropy of a saturated configuration is upper bounded by the total number of unstar monomers in the network and a polymer containing d unstar monomers decreases this upper bound by $d - 1$. If a polymer includes d unstar monomers, we say that the polymer does $d - 1$ unstar merge operations. Then, the entropy of a saturated configuration is equal to the value obtained by subtracting the total number of unstar merge operations across all polymers in the configuration from the total number of unstar monomers in the network.

We will consider two specific polymers: 1) $D_P^{j,1}$ is the polymer that includes the j -th polymer base encoding TAN polymer P , G_P^j , all base-specific horizontal monomers, and, for each $m_i \in P$, the mapped monomer encoding m_i with multiplicity equal to that of m_i in P . 2) $D_P^{j,2}$ is the polymer that includes G_P^j and all base-specific vertical monomers for the polymer base.



■ **Figure 4** One polymer gate for polymer $\{m_1, m_1, m_2\}$. (a) Base-specific vertical monomers (b) Polymer base encoding the polymer: the first row corresponds to the mapped part, and the remaining rows form the rectangular grid base. (c) Base-specific horizontal monomers (d) Mapped monomers that can bind to the mapped part of the polymer base. The legend inside the black frame shows the color encoding of the domains of mapped monomers and mapped part of the polymer base.

First, note that since neither $D_P^{j,1}$ nor $D_P^{j,2}$ has any exposed starred or unstarred domains, if either is contained in a self-saturated polymer, it can be split off without harming saturation. Furthermore, both polymers include only one star monomer, the polymer base G_P^j . Therefore, if either of these two polymers were splittable, as they have no exposed starred or unstarred domains, after splitting, the resulting polymer containing the polymer base would necessarily have an exposed starred domain and hence would not be self-saturated. Hence, both $D_P^{j,1}$ and $D_P^{j,2}$ are unsplittable polymers. Further note that the polymer $D_P^{j,1}$ includes $h(G_P) + |P| = k|P| - (u\mathcal{F}(P) + |P| + v)$ unstar monomers whereas $D_P^{j,2}$ includes $k|P|$ unstar monomers.

Our first observation, which is characterized by the following lemma, formalizes the local grid-gate-type behavior of each polymer gate in an arbitrary saturated configuration of $\mathcal{B}$:

► **Lemma 4.** *In a saturated configuration β of $\mathcal{B}$, if a polymer including a polymer base G_P^j does not contain all the base-specific horizontal monomers, it is either $D_P^{j,2}$ or splittable.*

Proof. If β is a saturated configuration, then every polymer in β is self-saturated; that is, no polymer contains an exposed starred domain. Consequently, in any self-saturated polymer in $\mathcal{B}$, all domains of G_P^j must be bound to an unstarred domain. Now suppose a polymer does not include all base-specific horizontal monomers. Since the domains of the grid base contained in G_P^j are not shared with any other polymer base, it follows by Lemma 3 that the polymer must include all base-specific vertical monomers in order to remain self-saturated. However, this implies that the polymer is either $D_P^{j,2}$ or $D_P^{j,2}$ can be split off without harming saturation. ◀

Next, we will show that no stable configuration of $\mathcal{B}$ can have a polymer including multiple polymer bases within.

► **Definition 5.** We say a polymer in $\mathcal{B}$ is complete with respect to a polymer base G_P^j if the polymer includes G_P^j and, for each $m_i \in P$, at least as many mapped monomers encoding m_i as the multiplicity of m_i in P .

► **Lemma 6.** If there exists a self-saturated polymer P' of $\mathcal{B}$ including multiple polymer bases that is complete with respect to at least one of them, the polymer is splittable and hence cannot appear in a stable configuration.

Proof. Suppose there exists a stable configuration β of $\mathcal{B}$ including a polymer P' which is complete for G_P^j . By Lemma 4, P' necessarily includes all the base-specific horizontals for G_P^j , since otherwise $S(\beta)$ could be increased by splitting off $D_P^{j,2}$, contradicting β being stable. Then, if P' is complete and includes all base-specific horizontals for G_P^j , $D_P^{j,1}$ can be split off from P' without harming saturation, which again contradicts β being stable. ◀

Lemma 4 and Lemma 6 imply that $\mathcal{B}$ has four types of unsplittable polymers:

1. A single polymer base G_P^j together with all base-specific vertical monomers (necessarily of the form $D_P^{j,2}$, since otherwise $D_P^{j,2}$ can be split off from the polymer).
2. A single polymer base G_P^j together with all base-specific horizontal monomers and complete with respect to G_P^j (necessarily of the form $D_P^{j,1}$, since otherwise $D_P^{j,1}$ can be split off from the polymer).
3. One or more polymer bases, each with all base-specific horizontal monomers, but not complete with respect to any base.
4. A free (unstarred) monomer, possibly a mapped monomer.

Next, we will show that the only way for a polymer of the form (3) to be unsplittable is to include exactly one polymer base:

► **Lemma 7.** If a self-saturated polymer of the form (3) includes multiple polymer bases, the polymer is splittable.

Proof. For a self-saturated polymer P of the form (3) containing multiple polymer bases, let R denote the multiset of TAN monomers whose mapped monomers are included in P . We construct a polymer P'' that can be split from P , such that both P'' and the remaining polymer P_r are self-saturated.

The polymer P'' consists of the polymer base $G_{P'}^i$, which is contained in P , together with the mapped monomers encoding the monomers in the multiset intersection $P' \cap R$, base-specific horizontal monomers and vertical monomers, base-specific or otherwise, whose unstarred domains correspond to the monomers in $P' \setminus R$. The number of such vertical monomers in P'' is exactly $k|P' \setminus R|$, so that they bind precisely to the starred domains in the mapped portion of $G_{P'}^i$, for which no mapped monomer in P is available to bind. Hence, every starred domain in P'' is saturated, and therefore P'' is self-saturated.

Now consider the remaining polymer P_r . By construction, P_r necessarily contains all base-specific horizontal monomers associated with each polymer base it includes. Consequently, every starred domain in the rectangular grid base of each polymer base it includes. Suppose, for contradiction, that P_r is not self-saturated. Then there exists an exposed starred domain in the mapped portion of some polymer base contained in P_r . However, in P'' , the number of unstarred domains capable of binding to the mapped portion is exactly equal to the number of starred domains in that mapped portion. Therefore, the existence of such an exposed starred domain in P_r would imply the existence of an exposed starred domain in P , contradicting the assumption that P is self-saturated. Thus, P_r is also self-saturated. ◀

Next, we show that the unsplittable polymers of the form (3) cannot appear in a stable configuration, by exhibiting an incentive to form $D_P^{j,2}$ when some or all of the mapped monomers encoding the monomers in P are absent:

► **Lemma 8.** *A stable configuration of TBN $\mathcal{B}$ cannot have a polymer of the form (3).*

Proof. Suppose there exists a stable configuration β of $\mathcal{B}$ such that β includes a polymer of the form (3). We will consider another saturated configuration β' . The polymers of the form (1) and form (2) in β are preserved in β' . On the other hand, β' includes $D_P^{j,2}$ for each G_P^j that is in a polymer of the form (3) in β . Note that the polymers of the form (1) in β only include base-specific vertical monomers for the polymer base they contain, and the polymers of the form (2) do not include any base-specific vertical monomers. Hence, β' can include polymers of the form $D_P^{j,2}$ for each G_P^j that are in a polymer of the form (3) in β while keeping form (1) and form (2) polymers of the same configuration intact.

Take a polymer of the form (3) from β , which, by Lemma 7, includes only a single polymer base G_P^j . As the polymer is self-saturated, each starred domain of a polymer base necessarily forms a bond. Moreover, any monomer other than a mapped monomer that can bind to the mapped part of G_P^j , precisely the vertical monomers, can only form one bond with the mapped part. As the polymer is incomplete for G_P^j , this implies that there are at least k -many additional vertical monomers included in the polymer. Furthermore, any polymer of the form (3) includes all base-specific horizontal monomers. Hence, the total number of unstar monomers in the polymer is at least $h(G_P^j) + k$, for which $h(G_P^j) + k = (k-1)|P| - (u\mathcal{F}(P) + |P| + v) + k = k|P| + k - (u\mathcal{F}(P) + 2|P| + v) > k|P|$ as $k = u \max_P \mathcal{F}(P) + 2|M| + v + 1 > (u\mathcal{F}(P) + 2|P| + v)$. Then a polymer of the form (3) including G_P^j causes a drop in entropy strictly greater than that caused by $D_P^{j,2}$. Hence, $S(\beta) < S(\beta')$ and β cannot be a stable configuration of $\mathcal{B}$. ◀

Lemma 8 implies that the polymers in a stable configuration of $\mathcal{B}$ are of the form (1), (2), or (4). Next, we show that there are further restrictions on the polymers of the form (4).

► **Lemma 9.** *No mapped monomer can be free in a stable configuration of $\mathcal{B}$.*

Proof. Suppose there exists a stable configuration β of $\mathcal{B}$ that includes at least one free mapped monomer, say m_1 . As $\mathcal{B}$ includes $|M|$ -many polymer bases for each TAN polymer, at least one polymer in β that includes a polymer base encoding $P = \{m_1\}$ is necessarily incomplete for the base. Hence, Lemma 8 implies that the polymer is $D_P^{j,2}$. The polymer $D_P^{j,2}$ includes $k|P| = k$ -many unstar monomers.

We consider another configuration, β' , where the only change from β is that the polymer base under consideration is in the form $D_P^{j,1}$, using the mapped monomer encoding m_1 (which is free in β) and the base-specific horizontal monomers (which are also free in β). The polymer $D_P^{j,1}$ consists of only $h(G_P^j) + |P| = h(G_P^j) + 1$ unstar monomers, where $k - (h(G_P^j) + 1) = u\mathcal{F}(P) + 1 + v$ by the definition of $h(G_P^j)$. Hence, forming $D_P^{j,1}$ uses $u\mathcal{F}(P) + 1 + v$ fewer unstar monomers. As $u, v \in \mathbb{Z}_{\geq 0}$, the term $u\mathcal{F}(P) + 1 + v$ is positive. Hence, $S(\beta') > S(\beta)$, which implies that β cannot be a stable configuration. ◀

Lemma 8 and Lemma 9 imply that in a stable configuration of $\mathcal{B}$, a polymer can be $D_P^{j,1}$, $D_P^{j,2}$, or a free unstar monomer that is not a mapped monomer. Thus, a mapped monomer encoding m_i is in the polymer $D_P^{j,1}$, where $m_i \in P$.

For an arbitrary configuration $\alpha = \{P_1, P_2, \dots, P_n\}$ of $\mathcal{A}$, there exists a saturated configuration of $\mathcal{B}$, stable or not, which consists of the polymers $D_{P_1}^{j_1,1}, D_{P_2}^{j_2,1}, \dots, D_{P_n}^{j_n,1}$ and $D_P^{j,2}$ for the remaining polymer bases, i.e., for $(j, P) \neq (j_i, P_i)$ for any $i \leq n$. We call such a

configuration a *base configuration* for α , since each mapped monomer is in a polymer of the form $D_P^{j,1}$ that includes a polymer base encoding P . If β is a base configuration for α , $R_{\text{conf}}(\beta) = \alpha$. Moreover, Lemma 4 through Lemma 9 imply that a stable configuration of $\mathcal{B}$ has the same form.

We denote by C the entropy of the configuration of $\mathcal{B}$ in which each polymer base forms the polymer $D_P^{j,2}$. As each formation of a $D_P^{j,1}$ polymer increases C by $u\mathcal{F}(P) + |P| + v$ and a stable configuration maximizes the entropy, the entropy of a stable configuration of $\mathcal{B}$ is equal to $\max_{\alpha}(C + \sum_{P \in \alpha}(u\mathcal{F}(P) + |P| + v))$.

► **Lemma 10.** *For each stable configuration β of $\mathcal{B}$, $R_{\text{conf}}(\beta)$ is a stable configuration of $\mathcal{A}$.*

Proof. Let β be a stable configuration of $\mathcal{B}$. As $\sum_{P \in \alpha} |P| = |M|$ for an arbitrary configuration α of $\mathcal{A}$, the expression for the entropy of a stable configuration of $\mathcal{B}$ can be rewritten as $C + |M| + \max_{\alpha}(\sum_{P \in \alpha}(u\mathcal{F}(P) + v))$. As β is a stable configuration, $R_{\text{conf}}(\beta)$ maximizes $\sum_{P \in \alpha}(u\mathcal{F}(P) + v)$. Hence, $R_{\text{conf}}(\beta) = \arg \max_{\alpha} \sum_{P \in \alpha}(u\mathcal{F}(P) + v) = \arg \max_{\alpha}(u(\sum_{P \in \alpha} \mathcal{F}(P)) + v|\alpha|) = \arg \max_{\alpha}(uH(\alpha) + vS(\alpha)) = \arg \max_{\alpha}(-E_{\text{TAN}}(\alpha)) = \arg \min_{\alpha} E_{\text{TAN}}(\alpha)$. Hence, $R_{\text{conf}}(\beta)$ is a stable configuration of $\mathcal{A}$. ◀

► **Lemma 11 (Relative energies).** *Given $\text{TAN } \mathcal{A} = (M, \mathcal{F}, u, v)$, let R_{conf} be the mapping from the TBN $\mathcal{B}$ to $\mathcal{A}$, constructed as above, $E_{\mathcal{A}}$ be the energy of a stable configuration of $\mathcal{A}$ and $S_{\mathcal{B}}$ be the entropy of a stable configuration of $\mathcal{B}$. For every configuration α of $\mathcal{A}$, there exists a saturated configuration $\beta \in R_{\text{conf}}^{-1}(\alpha)$ of $\mathcal{B}$ such that $S_{\mathcal{B}} - S(\beta) = E_{\text{TAN}}(\alpha) - E_{\mathcal{A}}$.*

Proof. For a configuration $\alpha = \{P_1, P_2, \dots, P_n\}$ of $\mathcal{A}$, consider β which is a base configuration for α . Note that $R_{\text{conf}}(\beta) = \alpha$. Therefore, $S(\beta) = C + |M| + \sum_{P \in \alpha}(u\mathcal{F}(P) + v)$. Furthermore, $S_{\mathcal{B}} = C + |M| + \max_{\alpha'} \sum_{P \in \alpha'}(u\mathcal{F}(P) + v)$. As $E_{\mathcal{A}} = \min_{\alpha'} -\sum_{P \in \alpha'}(u\mathcal{F}(P) + v)$ and $E(\alpha) = -\sum_{P \in \alpha}(u\mathcal{F}(P) + v)$, it follows that $S_{\mathcal{B}} - S(\beta) = \max_{\alpha'} \sum_{P \in \alpha'}(u\mathcal{F}(P) + v) - \sum_{P \in \alpha}(u\mathcal{F}(P) + v) = -E_{\mathcal{A}} + E(\alpha) = E_{\text{TAN}}(\alpha) - E_{\mathcal{A}}$. ◀

As any stable configuration α of $\mathcal{A}$ satisfies $E_{\mathcal{A}} = E(\alpha)$, the immediate corollary of Lemma 11 is the following:

► **Corollary 12.** *For every stable configuration α of $\mathcal{A}$, there exists a stable configuration β of $\mathcal{B}$ where $\beta \in R_{\text{conf}}^{-1}(\alpha)$.*

► **Theorem 13.** *For every $\text{TAN } \mathcal{A}$ with the simplified energy function, there exists a TBN $\mathcal{B}$ such that $\mathcal{B}$ indirectly realizes $\mathcal{A}$.*

Proof. The result is a direct consequence of Lemma 10 and Corollary 12. ◀

6 Realizing TANs with TBNs: Extending to the Full Energy Model

We now present the construction of the realizing TBN for the general TAN energy function:

$$E_{\text{TAN}}(\alpha) = -uH(\alpha) - vS(\alpha) + \sum_{P_s \in \text{supp}(\alpha)} \sum_{j=1}^{\alpha(s)} \log j$$

We assume that each value of $\log j$, for $j \in \{1, 2, \dots, |M|\}$, is represented with 5-digit precision and hence is rational. Since $u, v \in \mathbb{Q}_{\geq 0}$ and $\mathcal{F}$ is a non-negative rational valued function, there exists a smallest $C_{\text{TAN}} \in \mathbb{N}$ such that $C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j) \in \mathbb{N}$ for every $P \in \mathcal{P}(M)$ and $j \in \{1, 2, \dots, |M|\}$.

In contrast to the construction of Section 5, the new construction again consists of polymer gates and mapped monomers, and for every polymer P of the realized TAN $\mathcal{A}$, there are $|M|$ polymer bases G_P^j . However, the height of each polymer base now depends on its index. Specifically, for each $j \in \{1, 2, \dots, |M|\}$, we define $h(G_P^j) = (k - 1)|P| - C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j)$. Thus, unlike the construction in Section 5, the polymer bases associated with a fixed polymer P are no longer uniform in height. We further define $k = |M| + C_{\text{TAN}}(u \max_{P \in \mathcal{P}(M)} \mathcal{F}(P) + v + (\log |M| + 1)|M|) + 1$. Note that the definitions of C_{TAN} and k imply that $h(G_P^j) \in \mathbb{N}$.

The definition of $h(G_P^j)$ implies that the number of unstarred monomers in $D_P^{j,1}$, the polymer in which G_P^j is saturated by base-specific horizontal monomers and mapped monomers, is $h(G_P^j) + |P| = k|P| - C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j)$. Thus, compared to $D_P^{j,2}$, the polymer in which G_P^j is saturated by base-specific vertical monomers, the polymer $D_P^{j,1}$ contains $C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j)$ fewer unstarred monomers.

The choice of the height function $h(G_P^j)$ is motivated by the behavior of the combinatorial entropy term when polymers $D_P^{j,1}$ are formed for consecutive indices beginning with $j = 1$. In particular, the decrease arising from the entropy contribution of multiple copies of the same polymer is captured by the sum of the logarithmic terms appearing in the definition of $h(G_P^j)$. In Corollary 17, we show that this behavior is realized in the stable configurations of the construction.

Note that the new construction differs from the construction of Section 5 only in the definitions of $h(G_P^j)$ and k , while the overall structure of the construction remains unchanged. Consequently, Lemma 4 and Lemma 6 continue to hold immediately for the new construction, reducing the possible unsplittable polymers of the network to those of forms (1), (2), (3), and (4). Moreover, as in the original construction, unsplittable polymers of form (3) are further restricted by Lemma 7.

Accordingly, we focus on the new results, analogous to those from Lemma 8 through Theorem 13, which establish the intended behavior in the presence of the additional terms arising from combinatorial entropy. Throughout this section, we again denote the realized TAN by $\mathcal{A}$ and the realizing TBN by $\mathcal{B}$.

► **Lemma 14.** *A stable configuration of $\mathcal{B}$ cannot include a polymer of the form (3).*

Proof. Let G_P^j be the unique polymer base contained in an unsplittable polymer of form (3). Since the polymer is incomplete for P and contains all base-specific horizontal monomers, the number of unstarred monomers contained in the polymer is at least $h(G_P^j) + k = k|P| - |P| - C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j) + k$. Because $\log j \geq 0$ for all $j \geq 1$, the definition of k implies that $k - |P| - C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j) > 0$. Therefore, $h(G_P^j) + k > k|P|$, showing that a polymer of form (3) contains strictly more unstarred monomers than the polymer $D_P^{j,2}$.

The remainder of the argument proceeds analogously to Lemma 8. Suppose a configuration β of $\mathcal{B}$ contains a polymer of form (3). Then one can construct another configuration β' by replacing every such polymer with the corresponding polymer $D_P^{j,2}$, in which the polymer base is saturated using base-specific vertical monomers. This replacement strictly increases the number of polymers in the configuration, implying that $S(\beta) < S(\beta')$. This contradicts the assumption that β is stable. ◀

Lemma 14 implies that the only polymers that can be included in a stable configuration of $\mathcal{B}$ are of the form (1), (2), or (4). The result analogous to Lemma 9 is the following:

► **Lemma 15.** *A stable configuration of $\mathcal{B}$ cannot include a free mapped monomer.*

Proof. Similar to the proof of Lemma 9, suppose there exists a configuration β of $\mathcal{B}$ that contains a free mapped monomer m_1 . Then there exists a polymer base G_P^j , where $P = \{m_1\}$, that is included in a polymer of the form $D_P^{j,2}$.

Now consider a configuration β' obtained from β by keeping all polymers unchanged except that $D_P^{j,2}$ is replaced by $D_P^{j,1}$. Since $h(G_P^j) + |P| = k|P| - C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j)$ and $|P| = 1$, it follows that $S(\beta') - S(\beta) = C_{\text{TAN}}(u\mathcal{F}(P) + v + \log |M| + 1 - \log j)$, which is strictly positive, since $j \in \{1, 2, \dots, |M|\}$. Moreover, since $\mathcal{F}(P)$, u , and v are all non-negative, it follows that $S(\beta') > S(\beta)$, contradicting the stability of β . $\blacktriangleleft$

Next, we will show that the desired accumulating effect on the logarithmic terms is a necessary condition at stability:

► **Lemma 16.** *A stable configuration of $\mathcal{B}$ cannot include both $D_P^{j,2}$ and $D_P^{j+1,1}$ for $j \in \{1, 2, \dots, |M| - 1\}$.*

Proof. Suppose there exists a stable configuration β of $\mathcal{B}$ that contains both $D_P^{j,2}$ and $D_P^{j+1,1}$. The total number of unstarred monomers contained in these two polymers is $2k|P| - C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log(j + 1))$.

We construct another configuration β' , obtained from β by replacing the polymer bases under consideration so that they form polymers of type $D_P^{j,1}$ and $D_P^{j+1,2}$, respectively, instead of $D_P^{j,2}$ and $D_P^{j+1,1}$. The total number of unstarred monomers in $D_P^{j,1}$ and $D_P^{j+1,2}$ is then $2k|P| - C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j)$.

Hence, $S(\beta') - S(\beta) = C_{\text{TAN}}(\log(j + 1) - \log j)$, which is strictly positive. Therefore, β cannot be a stable configuration. $\blacktriangleleft$

As Lemma 14 implies that each polymer base is necessarily either in the polymer $D_P^{j,1}$ or $D_P^{j,2}$ at stability, the immediate corollary of Lemma 16 is the following:

► **Corollary 17.** *In a stable configuration of $\mathcal{B}$, for each polymer P of $\mathcal{A}$, one of the following is true:*

- G_P^j is in the polymer $D_P^{j,2}$ for all $j \in \{1, 2, \dots, |M|\}$
- There exists $j' \in \{1, 2, \dots, |M|\}$ such that each polymer base G_P^j for $j \leq j'$ is in the polymer $D_P^{j,1}$ and each polymer base G_P^j for $j' < j$ is in the polymer $D_P^{j,2}$.

For an arbitrary configuration α of $\mathcal{A}$, where $\text{supp}(\alpha) = \{P_1, P_2, \dots, P_n\}$, we define the saturated configuration of $\mathcal{B}$, stable or not, which, for each $s \in \{1, 2, \dots, n\}$, consists of the polymers $D_{P_s}^{j,1}$ for $j \in \{1, 2, \dots, \alpha(s)\}$ and $D_{P_s}^{j,2}$ for $j > \alpha(s)$, as the *base configuration* for α (all polymer bases encoding polymers that are not in $\text{supp}(\alpha)$ form $D_P^{j,2}$). In contrast to the construction of Section 5, the base configuration for α is unique, since the present construction explicitly depends on the indices of the polymer bases. Furthermore, if β is the base configuration for α , then $R_{\text{conf}}(\beta) = \alpha$. Hence, R_{conf} defines a bijection from the set of base configurations of $\mathcal{B}$ to the set of all configurations of $\mathcal{A}$.

Moreover, the fact that the base configuration contains polymers $D_{P_s}^{j,1}$ for consecutive indices beginning at $j = 1$ and ending at $j = \alpha(s)$ is consistent with Corollary 17. Together, Lemma 14 through Corollary 17 imply that every stable configuration of $\mathcal{B}$ is necessarily a base configuration.

As in Section 5, we denote by C_2 the entropy of the configuration of $\mathcal{B}$ in which every polymer base forms the polymer $D_P^{j,2}$. Each formation of a polymer $D_P^{j,1}$ increases this entropy by $C_{\text{TAN}}(u\mathcal{F}(P) + v + (\log |M| + 1)|P| - \log j)$. Since the base configuration for α contains polymers $D_{P_s}^{j,1}$ for consecutive indices, the logarithmic terms accumulate. Therefore, if β is the

base configuration corresponding to α , then $S(\beta) = C_2 + C_{\text{TAN}} \sum_{P_s \in \text{supp}(\alpha)} (\underline{\alpha}(s) u \mathcal{F}(P_s) + \underline{\alpha}(s) v + \underline{\alpha}(s) (\log |M| + 1) |P_s| - \sum_{j=1}^{\underline{\alpha}(s)} \log j)$. Using the identity $\sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s) |P_s| = |M|$, it follows that $S(\beta) = C_2 + C_{\text{TAN}} (\log |M| + 1) |M| - C_{\text{TAN}} E_{\text{TAN}}(\alpha)$.

The fact that every stable configuration of $\mathcal{B}$ is necessarily a base configuration, together with the fact that R_{conf} is a bijection when restricted to the set of base configurations of $\mathcal{B}$, allows us to establish both directions of the thermodynamic realization of $\mathcal{A}$ by $\mathcal{B}$.

► **Lemma 18.** *For each stable configuration β of $\mathcal{B}$, $R_{\text{conf}}(\beta)$ is a stable configuration of $\mathcal{A}$.*

Proof. Suppose that β is a stable configuration of $\mathcal{B}$ but $R_{\text{conf}}(\beta) = \alpha$ is not a stable configuration of $\mathcal{A}$. Then there exists α' such that $E_{\text{TAN}}(\alpha') < E_{\text{TAN}}(\alpha)$. Let β' be the base configuration for α' . Then $S(\beta') = C_2 + C_{\text{TAN}} (\log |M| + 1) |M| - C_{\text{TAN}} E_{\text{TAN}}(\alpha') > C_2 + C_{\text{TAN}} (\log |M| + 1) |M| - C_{\text{TAN}} E_{\text{TAN}}(\alpha) = S(\beta)$, which contradicts β being a stable configuration of $\mathcal{B}$. ◀

► **Lemma 19.** *For every stable configuration α of $\mathcal{A}$, there exists a stable configuration β of $\mathcal{B}$ where $\beta \in R_{\text{conf}}^{-1}(\alpha)$.*

Proof. Suppose α is a stable configuration of $\mathcal{A}$ but β , the base configuration for α is not a stable configuration of $\mathcal{B}$. Then there exists a base configuration β' , where $R_{\text{conf}}(\beta') = \alpha'$, such that $S(\beta') = C_2 + C_{\text{TAN}} (\log |M| + 1) |M| - C_{\text{TAN}} E_{\text{TAN}}(\alpha') > C_2 + C_{\text{TAN}} (\log |M| + 1) |M| - C_{\text{TAN}} E_{\text{TAN}}(\alpha)$, which implies $E_{\text{TAN}}(\alpha') < E_{\text{TAN}}(\alpha)$, which contradicts α being a stable configuration of $\mathcal{A}$. ◀

► **Theorem 20.** *For every TAN $\mathcal{A}$, there exists a TBN $\mathcal{B}$ such that $\mathcal{B}$ indirectly realizes $\mathcal{A}$.*

Proof. The result is a direct consequence of Lemma 18 and Lemma 19. ◀

Next, we show that a set of TBN monomers obtained via our construction can, without modifying the monomer types, indirectly realize a family of TANs that share the same underlying monomer set but differ in their monomer counts. In other words, the realizing TBN is fixed at the level of monomer types, and only the multiplicities are varied to realize different target TANs.

Before presenting the result, we define a *TAN template* as a 4-tuple $(M_{\text{mon}}^{\text{TAN}}, \mathcal{F}, u, v)$, where $M_{\text{mon}}^{\text{TAN}}$ denotes the set of monomer types. A TAN template specifies a family of TANs obtained by assigning different count vectors to the fixed monomer set. That is, each member TAN is determined by a count vector $\vec{m}_{\text{count}}^{\text{TAN}}$, which specifies the number of copies of each monomer type in $M_{\text{mon}}^{\text{TAN}}$, or equivalently $\vec{m}_{\text{count}}^{\text{TAN}} \in \mathbb{N}^{|M_{\text{mon}}^{\text{TAN}}|}$. Note that, for a fixed TAN template, the pair $(M_{\text{mon}}^{\text{TAN}}, \vec{m}_{\text{count}}^{\text{TAN}})$ uniquely determines a TAN instance. We use an analogous notation for TBNs, writing the underlying monomer set of a TBN as $M_{\text{mon}}^{\text{TBN}}$, and representing a particular TBN instance by the pair $(M_{\text{mon}}^{\text{TBN}}, \vec{m}_{\text{count}}^{\text{TBN}})$. We denote the case in which each entry of $\vec{m}_{\text{count}}^{\text{TAN}}$ is bounded above by a fixed value M' as $\vec{m}_{\text{count}}^{\text{TAN}} \leq M'$.

► **Theorem 21.** *For each TAN template $(M_{\text{mon}}^{\text{TAN}}, \mathcal{F}, u, v)$, there exists a set of TBN monomers $M_{\text{mon}}^{\text{TBN}}$ such that for every count vector $\vec{m}_{\text{count}}^{\text{TAN}}$ satisfying $\vec{m}_{\text{count}}^{\text{TAN}} \leq M'$, there exist TBN monomer counts $\vec{m}_{\text{count}}^{\text{TBN}}$ such that the TBN $(M_{\text{mon}}^{\text{TBN}}, \vec{m}_{\text{count}}^{\text{TBN}})$ indirectly realizes the TAN $(M_{\text{mon}}^{\text{TAN}}, \vec{m}_{\text{count}}^{\text{TAN}})$.*

Proof. For a given TAN template $(M_{\text{mon}}^{\text{TAN}}, \mathcal{F}, u, v)$, where $|M_{\text{mon}}^{\text{TAN}}| = n$, let P' be a polymer maximizing $\mathcal{F}$ among all polymers over $M_{\text{mon}}^{\text{TAN}}$ with at most nM' monomers.

We construct the set of TBN monomers $M_{\text{mon}}^{\text{TBN}}$ as the collection of indexed polymer gates (i.e., polymer bases, base-specific horizontal monomers, and base-specific vertical monomers) for all polymers over $M_{\text{mon}}^{\text{TAN}}$ containing at most M' copies of each monomer, together with mapped monomers encoding each monomer in $M_{\text{mon}}^{\text{TAN}}$. We define $h(G_P^j) = (k-1)|P| - C_{\text{unif}}(u\mathcal{F}(P) + v + (\log(nM') + 1)|P| - \log j)$, and $k = (nM') + C_{\text{unif}}(u\mathcal{F}(P') + v + (\log(nM') + 1)nM') + 1$. Let C_{unif} be the smallest natural number such that $C_{\text{unif}}(u\mathcal{F}(P) + v + (\log(nM') + 1)|P| - \log j)$ is a natural number for every P with $|P| \leq nM'$ and every $j \in \{1, 2, \dots, nM'\}$, where each $\log j$ is taken to 5-digit precision.

Given a TAN instance specified by $\vec{m}_{\text{count}}^{\text{TAN}}$, we instantiate the corresponding TBN counts as follows. First, we include mapped monomers encoding each monomer in $M_{\text{mon}}^{\text{TAN}}$ with multiplicities given by $\vec{m}_{\text{count}}^{\text{TAN}}$. Second, for each polymer over $M_{\text{mon}}^{\text{TAN}}$ compatible with these counts, we include $\sum_i \vec{m}_{\text{count}}^{\text{TAN}}(i)$ copies of the corresponding polymer gates, with heights determined by the above definition of $h(G_P^j)$.

Note that the resulting TBN is structurally identical to the construction used for realizing a fixed-count TAN, except for the specific values of $h(G_P^j)$ and k . These choices are uniform over all TAN instances with counts of each monomer bounded by M' , and correctness follows by adapting the proofs of Lemma 14 and Lemma 15. In particular, since P' maximizes $\mathcal{F}$ over all polymers of size at most nM' , we obtain $h(G_P^j) + k > k|P|$ and thus the analogue of Lemma 14 holds. Moreover, since $j \leq \sum_i \vec{m}_{\text{count}}^{\text{TAN}}(i) \leq nM'$, we have $\log j \leq \log(nM')$, from which the analogue of Lemma 15 follows. Finally, the arguments in Lemma 16 through Theorem 20 do not depend on monomer counts and therefore carry over directly. Hence, the constructed TBN thermodynamically realizes the given TAN. ◀

7 Discussion

Initially, TBNs looked too restricted to express arbitrary polymer-level preferences. Their only source of enthalpic favorability is pairwise complementary binding, so attaching a reward to a whole complex (rather than to the bonds within it) seemed out of reach. Our main result shows that this intuition is misleading and that every TAN, however its affinities are assigned, has a TBN whose stable configurations match it. The construction works by encoding the polymer-level rewards indirectly: when a polymer forms, it releases a calibrated number of helper monomers, matching the favorability described by the specific TAN it is realizing. Moreover, the construction has a limited form of uniformity: for any bounded family of TANs (that differ only by counts of monomers), one fixed TBN can realize the family by changing only multiplicities. In this way, TANs are not a more powerful model; rather, they are a higher-level language for the thermodynamically equivalent model of TBNs. An important caveat is that our equivalence means that we match stable, minimum-free-energy configurations, not full kinetics or Boltzmann distributions. A natural next step is to aim for capturing a more complete notion of equilibrium thermodynamics.

Towards that objective, we note that Lemma 11 (and the analogous entropy identity in Section 6) shows that the correspondence between TANs and TBNs is stronger than a mere agreement on which configurations are stable. For a TAN configuration α , the construction produces a saturated TBN configuration β mapping to α such that the gap from stability in the TAN is reflected in the gap from maximal entropy in the TBN. Thus, the construction preserves relative thermodynamic preferences, not just equilibrium outcomes. A direct line of future work is expanding our definitions of system equivalence. A “complete” thermodynamic realization may consider preserving all relative energies of a system, rather

than just stable configurations. Developing stronger notions of realization may also be a useful step toward comparing kinetic behavior as well, possibly allowing us to establish more traditional simulation results.

Lastly, because Thermodynamic Binding Networks were introduced as a domain-level abstraction inspired by DNA binding, one can naturally interpret each TBN domain as a designed DNA domain and each monomer as a multi-domain strand or complex [7, 18]. Under this interpretation, one implication of our equivalence result is that any TAN-style equilibrium polymer preference can, in principle, be compiled into DNA-level binding components whose stable complexes match the intended minimum-free-energy states. This perspective aligns with the broader tradition of DNA universality, where strand-displacement systems have already been shown to be universal for implementing arbitrary chemical kinetics [17]. Accordingly, this paper supports the view that DNA may be regarded as a conceptually complete substrate for equilibrium monomer-polymer thermodynamics, although further investigation is needed.

References

- 1 Gagan Aggarwal, Qi Cheng, Michael H Goldwasser, Ming-Yang Kao, Pablo Moisset De Espanes, and Robert T Schweller. Complexities for generalized models of self-assembly. *SIAM Journal on Computing*, 34(6):1493–1515, 2005. doi:10.1137/S0097539704445202.
- 2 Marco Antoniotti, Carla Piazza, Alberto Policriti, Marta Simeoni, and Bud Mishra. Taming the complexity of biochemical models through bisimulation and collapsing: theory and practice. *Theoretical Computer Science*, 325(1):45–67, 2004. doi:10.1016/J.TCS.2004.03.064.
- 3 Keenan Breik, Cameron Chalk, David Doty, David Haley, and David Soloveichik. Programming substrate-independent kinetic barriers with thermodynamic binding networks. *IEEE/ACM transactions on computational biology and bioinformatics*, 18(1):283–295, 2019.
- 4 Keenan Breik, Chris Thachuk, Marijn Heule, and David Soloveichik. Computing properties of stable configurations of thermodynamic binding networks. *Theoretical Computer Science*, 785:17–29, 2019. doi:10.1016/J.TCS.2018.10.027.
- 5 Sven De Vries and Rakesh V Vohra. Combinatorial auctions: A survey. *INFORMS Journal on computing*, 15(3):284–309, 2003. doi:10.1287/IJOC.15.3.284.16077.
- 6 Robert M Dirks, Justin S Bois, Joseph M Schaeffer, Erik Winfree, and Niles A Pierce. Thermodynamic analysis of interacting nucleic acid strands. *SIAM review*, 49(1):65–88, 2007. doi:10.1137/060651100.
- 7 David Doty, Trent A Rogers, David Soloveichik, Chris Thachuk, and Damien Woods. Thermodynamic binding networks. In *DNA Computing and Molecular Programming: 23rd International Conference, DNA 23, Austin, TX, USA, September 24–28, 2017, Proceedings 23*, pages 249–266. Springer, 2017. doi:10.1007/978-3-319-66799-7_16.
- 8 Jean-Claude Fernandez. An implementation of an efficient algorithm for bisimulation equivalence. *Science of Computer Programming*, 13(2-3):219–236, 1990. doi:10.1016/0167-6423(90)90071-K.
- 9 Bin Fu, Matthew J Patitz, Robert T Schweller, and Robert Sheline. Self-assembly with geometric tiles. In *International Colloquium on Automata, Languages, and Programming*, pages 714–725. Springer, 2012. doi:10.1007/978-3-642-31594-7_60.
- 10 David Haley and David Doty. Computing properties of thermodynamic binding networks: An integer programming approach. *arXiv preprint arXiv:2011.10677*, 2020. arXiv:2011.10677.
- 11 Robert Johnson, Qing Dong, and Erik Winfree. Verifying chemical reaction network implementations: a bisimulation approach. *Theoretical Computer Science*, 765:3–46, 2019. doi:10.1016/J.TCS.2018.01.002.
- 12 Maxim P Nikitin. Non-complementary strand commutation as a fundamental alternative for information processing by dna and gene regulation. *Nature Chemistry*, 15(1):70–82, 2023.
- 13 Tosan Omabegho. Allosteric linkages that model the chemical cycle of a molecular motor enzyme. *bioRxiv*, pages 2021–04, 2021.

- 14 Joshua Petrack, David Soloveichik, and David Doty. Thermodynamically driven signal amplification. In *29th International Conference on DNA Computing and Molecular Programming*, page 1, 2023.
- 15 Talal Rahwan, Tomasz P Michalak, Michael Wooldridge, and Nicholas R Jennings. Coalition structure generation: A survey. *Artificial Intelligence*, 229:139–174, 2015. doi:10.1016/J.ARTINT.2015.08.004.
- 16 Jason W Rocks, Nidhi Pashine, Irmgard Bischofberger, Carl P Goodrich, Andrea J Liu, and Sidney R Nagel. Designing allostery-inspired response in mechanical networks. *Proceedings of the National Academy of Sciences*, 114(10):2520–2525, 2017.
- 17 David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393–5398, 2010.
- 18 Boya Wang, Cameron Chalk, David Doty, and David Soloveichik. Molecular computation at equilibrium via programmable entropy. *Science Advances*, 12(2):eadx3969, 2026.
- 19 Erik Winfree. *Algorithmic self-assembly of DNA*. California Institute of Technology, 1998.

A Derivation of the Macrostate Energy Function

In Section 2 we assign to each configuration α the energy $-\log \Omega(\alpha)$, where $\Omega(\alpha)$ is the number of microstates consistent with the configuration α . This appendix derives that expression from first principles, by counting the microstates of an ideal dilute solution. We measure energy in units of $k_B T$ and entropy in units of k_B , so that $\log \Omega$ is an entropy and $-\log \Omega(\alpha)$ a dimensionless free energy; $\log$ denotes the natural logarithm. The derivation is based on [6], but is standard in the statistical mechanics literature.

As in Section 2, let M be the finite multiset of *monomer* types and $\mathcal{P}(M)$ the set of *polymer* types, each a nonempty sub-multiset of M . A *macrostate* is a configuration α , that is, a partition of M into polymers; equivalently, it records the molecular count $\underline{\alpha}(s)$ of each polymer type $P_s \in \text{supp}(\alpha) = \{P_1, P_2, \dots, P_n\}$. Let $|\alpha| = \sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s)$ denote the total number of polymers. We model the solution as V cells, where V is the volume in units of a reference cell, and we work in the dilute regime $V \gg |\alpha|$, so that polymers occupy cells independently (the ideal-solution assumption).

For each polymer type P , let Ω_P count the microstates of the universe that are made available by assembling a single polymer of type P from its constituent free monomers. These microstates include internal microstates of the polymer, such as different conformations, as well as those arising from heat released into the thermal reservoir due to the formation of bonds.

The number $\Omega(\alpha)$ of microstates factors into four contributions:

- (A) **Positioning the polymers.** Treating the $|\alpha|$ polymers as distinguishable objects, each independently occupies one of V cells: $A = V^{|\alpha|}$.
- (B) **Making the monomers distinguishable.** Writing μ_a for the number of copies of monomer type a in M , labeling the monomers of each type so that they may be told apart contributes $B = \prod_a \mu_a!$, the product ranging over monomer types.
- (C) **Indistinguishability of polymers.** Polymers of the same type are interchangeable, so the relabelings of identical polymers must be removed: $C = \frac{1}{\prod_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s)!}$.
- (D) **Internal microstates.** Each polymer of type P_s independently realizes one of Ω_{P_s} microstates: $D = \prod_{P_s \in \text{supp}(\alpha)} \Omega_{P_s}^{\underline{\alpha}(s)}$.

Collecting the four factors,

$$\Omega(\alpha) = A \cdot B \cdot C \cdot D = V^{|\alpha|} \left(\prod_a \mu_a! \right) \frac{1}{\prod_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s)!} \prod_{P_s \in \text{supp}(\alpha)} \Omega_{P_s}^{\underline{\alpha}(s)}.$$

For the fixed monomer multiset M , all microstates of the universe are equiprobable, so the probability of a configuration α is proportional to $\Omega(\alpha)$. Thus, the most probable configurations are those that minimize $-\log \Omega(\alpha)$, which we therefore adopt as the energy. The factor (B) depends only on the fixed multiset M and not on the configuration α , so it shifts $-\log \Omega(\alpha)$ by a constant and is irrelevant to the minimization. (Equivalently, whether the monomers are deemed distinguishable has no effect on equilibrium, the familiar Gibbs resolution of the mixing paradox.) Dropping this constant leaves

$$\Omega(\alpha) = \frac{V^{|\alpha|}}{\prod_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s)!} \prod_{P_s \in \text{supp}(\alpha)} \Omega_{P_s}^{\underline{\alpha}(s)}, \quad \text{up to a configuration-independent factor.}$$

Because the regimes of interest often involve only a few copies of each polymer, we retain the factorial exactly rather than approximating it via Stirling's formula $\log(n!) = n \log n - n + O(\log n)$ (as in [6]), which is accurate only when every count $\underline{\alpha}(s)$ is large. Writing each $\log(\underline{\alpha}(s)!) as a direct sum of logarithms,$

$$\log(\underline{\alpha}(s)!) = \log \underline{\alpha}(s) + \log(\underline{\alpha}(s) - 1) + \cdots + \log 2 = \sum_{j=1}^{\underline{\alpha}(s)} \log j,$$

an empty sum equal to 0 when $\underline{\alpha}(s) \leq 1$, then taking logarithms of $\Omega(\alpha)$ and negating gives the exact macrostate energy,

$$-\log \Omega(\alpha) = - \sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s) \log V - \sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s) \log \Omega_{P_s} + \sum_{P_s \in \text{supp}(\alpha)} \sum_{j=1}^{\underline{\alpha}(s)} \log j, \quad (1)$$

valid for every configuration, whatever the counts. Identifying $\log V = v$ and $\log \Omega_{P_s} = u \mathcal{F}(P_s)$ with $u = w \log V$, and recalling that $S(\alpha) = |\alpha|$ is the number of polymers and $H(\alpha) = \sum_{P_s \in \text{supp}(\alpha)} \underline{\alpha}(s) \mathcal{F}(P_s)$ the enthalpy, the first two sums become $-vS(\alpha)$ and $-uH(\alpha)$, so that

$$-\log \Omega(\alpha) = -uH(\alpha) - vS(\alpha) + \sum_{P_s \in \text{supp}(\alpha)} \sum_{j=1}^{\underline{\alpha}(s)} \log j = E_{\text{TAN}}(\alpha),$$

the energy of Section 2, whose final term is the combinatorial entropy.

Modularity of Signal Propagation Networks in the Thermodynamic Binding Network Model

And Kaan Ata Yilmaz ✉ 

Electrical and Computer Engineering, University of Texas at Austin, TX, USA

David Soloveichik ✉ 

Electrical and Computer Engineering, University of Texas at Austin, TX, USA

Abstract

The Thermodynamic Binding Network (TBN) model provides an equilibrium-based abstraction for molecular systems, but proving correctness of large networks built from individually verified modules remains difficult because modules can interact through shared binding-site types. We develop a method for proving correctness of modular TBNs by reducing a composition of arbitrarily many modules to a fixed coarse-grained TBN, computing the Hilbert basis of the reduced system, and lifting the resulting entropy bounds and polymer characterizations back to the original network. We apply this method to reversible signal-propagation modules: For the previously studied module implementing $A + B \longleftrightarrow C$, we show that stable configurations of an arbitrary network of such modules have the intended local structure, implying that firing an individual module preserves global stability. Under certain restrictions on the reaction set, these networks faithfully simulate a class of reversible chemical reaction networks: every bounded-length CRN execution can be realized by a height-2 path between stable TBN configurations, and any two stable TBN configurations map to CRN states in the same stoichiometric compatibility class. We further show that the same proof strategy extends to a novel module implementing reactions with two reactants and products like $A + B \longleftrightarrow C + D$.

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases Thermodynamic Binding Networks, Chemical Reaction Networks, Modular Composition, Verification, Simulation

Digital Object Identifier 10.4230/LIPIcs.DNA.32.10

Funding Schmidt Sciences Polymath Award to D.S., NSF SemiSynBio III: GOALI grant 2227578.

1 Introduction

Much like the field of digital electronics, progress in engineering molecular computation has depended on a collection of abstractions and models, ranging from those closer to the underlying hardware like strand displacement cascades [15, 12], to more general ones like algorithmic tile assembly [6] and chemical reaction networks [14, 4]. The Thermodynamic Binding Network (TBN) model [7, 3] was introduced to capture a class of systems designed to obey certain simplifying restrictions: (1) behavior is not programmed through geometric or steric constraints, (2) binding occurs only between complementary site types (aka domains in DNA terminology), (3) configurations maximizing the number of bonds are preferred, with entropy, measured by the number of separate polymers (aka complexes), breaking ties between fully saturated configurations. Recent experimental implementations of simple TBNs reinforce the practical relevance of the model as an engineering abstraction [17].

While the original intent of the TBN model was to capture thermodynamic equilibria, the model has subsequently been extended with a notion of kinetic paths and kinetic barriers [2]. Importantly, this is still quite different from a kinetics-first model such as strand-displacement cascades. Rather than assigning rates and focusing on the most likely trajectories, the TBN model considers all possible rearrangements of the components that are permitted by the



© And Kaan Ata Yilmaz and David Soloveichik;
licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 10; pp. 10:1–10:23

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

binding rules. This conservative viewpoint has an important advantage for verification: proving correctness in the TBN model rules out not only failure along the intended pathway, but also all unintended “leak” pathways. In this sense, it helps guard against errors that arise when thermodynamic driving forces conflict with the desired kinetic behavior.

Despite the apparent simplicity of the TBN model, it is surprisingly powerful. For example, there is a certain sense in which the stable configurations of TBNs can simulate the minimum free energy configurations of any other monomer-polymer system [18]. Another indication of the expressive power of the TBN model is that natural analysis problems for it are computationally hard (under reasonable computational complexity assumptions like $P \neq NP$) [3, 10]. Nonetheless, although worst-case analysis may be intractable, engineered TBNs may have certain properties that make their analysis and verification of correctness effective.

In this paper we develop a new method of proving the correctness of TBN systems that have been designed to be modular. Specifically, the monomers on such TBNs can be logically partitioned into “modules” that share a common underlying structure. Most TBNs are modular in this sense, including constructions for logic circuits [7], amplification [11], and signal propagation [17]. Since each individual module contains few monomers, it is easy to check its correctness in isolation using existing computational tools [3, 9, 16, 8]. However, verifying isolated module behavior in no way implies that the combined system behaves as intended. Indeed, modules necessarily share binding site types and the thermodynamically preferred configuration may involve some unintended cross-module hybrid complexes.

The method of analysis we develop is based on restricting the possible form of polymers in the combined system. Specifically, we employ a three-step argument: (1) Create a reduction from the original TBN into another TBN in which all modules collapse into a single one. (2) Compute the Hilbert-basis of the reduced TBN to enumerate all the polymers that could possibly appear in its stable configuration independently of the monomer counts. Importantly, step (2) leaves us with the same set of polymers in the reduced system for infinitely many original TBNs (i.e., arbitrary compositions of the given modules). (3) Eliminate the remaining undesired polymers based on monomer counts and how the modules are composed in the original TBN. We apply our method to two modular TBN signal propagation networks: one theoretically and experimentally studied in prior work [17], and one novel. While we cannot guarantee that other constructions will be amenable to the same analysis, we hope that our method is general enough to handle other systems.

While the emphasis of this work is on developing a general proof technique, of independent interest, our results prove the correctness (in the sense described below) of composition of modules for implementing systems of reversible reactions like $A + B \longleftrightarrow C$. We impose certain restrictions on the modules and how the modules are composed; for example, reactions repeating species such as $A + A \longleftrightarrow B$ are disallowed, as well as pairs of reactions that overlap in two or more species such as $A + B \longleftrightarrow C$ with $C + B \longleftrightarrow D$. Under these restrictions, we prove that all complexes that occur in the overall TBN can be mapped to individual modules (i.e., modules do not interfere with one another). This, in turn, allows us to prove faithful kinetic simulation in the following sense: First, given enough (fuel) monomers, the TBN simulates any number of CRN reaction firings. Specifically, there is a height-2 path as interpreted via TBN kinetics formalized in prior work [2]. Second, for the other direction, we show that stable configurations of the TBN correspond to CRN configurations in the same stoichiometric compatibility class: that is the TBN cannot reach a stable configuration that the CRN’s conservation laws forbid.

Note that if the target CRN violates our restrictions, we can add additional reactions to effectively “translate” between species. With enough additional fuel, valid paths in the original CRN have valid images in the new CRN. While it remains as future work to show that any height-2 path in the TBN corresponds to a valid CRN trajectory, our results show that rather simple TBNs can provably simulate reversible chemical reaction networks in the sense described. Further, since reaction $A + B \rightarrow C$ implements a logic AND gate, and two different reactions outputting the same species implement implicit OR, our results also imply simulation of dual-rail logic circuits. Boolean completeness of TBNs was previously established, but via different constructions [7, 5].

2 TBN Definitions

A *site type* is a formal symbol indexed by an integer, such as a_1 . Each site type has a complementary type, denoted a_1^* , with the interpretation that a_1 can bind to a_1^* . We refer to site types according to both their index and whether they carry a star: for example, a_1 is called a *1-unstarred site type*, while a_1^* is called a *1-starred site type*. A *monomer type* is defined as a multiset of site types. For instance, the monomer type $m = \{a_1, a_1, a_1^*, b_2^*, b_2^*\}$ contains two copies of a_1 , two copies of b_2^* , and one copy of a_1^* . The *length* of a monomer type m is $|m|$, i.e. the number of binding sites included in m .

A *thermodynamic binding network (TBN)* is a multiset of monomer types. An individual occurrence of a monomer type in a TBN is called a *monomer*, and an individual occurrence of a site type within a monomer is called a *site*. We say a TBN is *star-limiting* if for each site type, there are at least as many sites of the unstarred type as the starred type among all monomers.

A *configuration* of a TBN is a partition of its monomers into submultisets called *polymers*. The number of polymers in a configuration α , counted with multiplicity, is called the *entropy* of α and is denoted by $S(\alpha)$. A subconfiguration of a configuration α is a submultiset of polymers included in α . A site type within a polymer is said to be *exposed* if, among the monomers in that polymer, the number of starred instances of that site type exceeds the number of unstarred instances. A polymer is *self-saturated* if it contains no exposed site types. A self-saturated polymer is *splittable* if it can be split into two self-saturated polymers. For a star-limiting TBN, a configuration is *saturated* if every polymer in the configuration is self-saturated. Intuitively, this means that no polymer contains an unmatched starred site and equivalently, all possible bonds are formed. A configuration α of a TBN T is *stable* if it is saturated and no other saturated configuration of T contains more polymers than α . Equivalently, stable configurations are precisely the entropy-maximizing saturated configurations.

Configurations evolve through sequences of elementary merge and split operations. Specifically, if a configuration α can be transformed into a configuration β by replacing two polymers in α with their union, then α is said to *merge* to β , and correspondingly, β *splits* to α . A *path* is a nonempty sequence of configurations in which each configuration merges or splits to the next. Observe that any two configurations are connected by a path. A path between two saturated configurations is a saturated path if each configuration along the path is also saturated. In the TBN model, bonds are assumed to be strong and maximum bonding is assumed at all times. Since any transition to an unsaturated configuration necessarily involves losing a bond, we consider only saturated paths.

The *height* $h(p)$ of a saturated path p with initial configuration α is the maximum entropy decrease $S(\alpha) - S(\gamma)$ over all configurations γ along p . The height of a path captures the difficulty of a transition by measuring the largest entropy loss incurred along the path. For

this interpretation, we need the initial and final configurations to be stable; for example, a large height need not indicate a difficult transition if the initial configuration is unstable and the whole path is downhill.

3 Reversible Signal Propagation Module

We consider circuits of reversible signal propagation modules introduced in prior work ([7, 3]) and recently realized experimentally [17]. The construction consists of two closely related types of modules, forward and backward. The forward module represents a reaction in the stoichiometric form $A + B \rightarrow C$, and the backward module represents the reaction in the reverse direction, i.e., $C \rightarrow A + B$.

When a forward or backward module is considered independently from the rest of the network and without input, the module has exactly one stable configuration (forward module in Figure 1(a) and backward module in Figure 2(a)). We call these configurations *untriggered modular configurations* for the forward and backward modules, respectively. Note that the fact that these configurations are the unique stable ones of an isolated module without input can be easily checked via tools such as **StableGen** or **stable_tbn** [3, 9, 16, 8]. Note that the output monomers are not free in either of these configurations.

Parts (b), (c) and (d) of Figure 1 and Figure 2 show all stable configurations of the forward and backward modules in the presence of input monomers, i.e., input A and B for the forward module and input C for the backward module. The only stable configurations where the output monomer(s) is free are shown in Part (d) of each figure. Again, the fact that these are exactly the stable configurations can be verified with the above mentioned software tools.

Importantly, the stable configuration, where the output(s) is released (Part (d) of Figure 1 and 2), contains the untriggered modular configuration of the reverse reaction in addition to free output(s). This is the sense in which these modules are reversible [17]. When the configuration is changed from Part (b) to Part (d), we say that the reaction has fired. Firing a reaction necessitates a height-2 path.

► **Definition 1.** *For a given reaction, we define the monomers in the untriggered modular configuration as module monomers. For any input or output species X , a monomer of the form $\{x_1, x_2\}$ is called an input monomer or output monomer, respectively.*

Note that the module monomers of a reaction include its output monomer(s). For a particular reaction R , we denote the set of module monomers by M_R and the set of input monomers by I_R . The definition of input monomers immediately implies a natural correspondence between the monomers of the signal propagation module and the input species of the reaction through $\{x_1, x_2\} \leftrightarrow X$.

As an example, consider the reaction $R: A + B \rightarrow C$. The set of *module monomers* is

$$M_R = \{\{a_1^*, a_2^*, b_1^*, b_2^*\}, \{a_1, a_2, b_1, b_2, c_1\}, \{a_2^*, b_1^*, b_2^*, c_1^*\}, \{a_2, b_1\}, \{b_2, c_1, c_2\}, \{c_1^*, c_2^*\}, \{c_1, c_2\}\},$$

that is, the set of monomers appearing in the configuration shown in Figure 1(a). The set of *input monomers* for this reaction is $I_R = \{\{a_1, a_2\}, \{b_1, b_2\}\}$. The output monomer $\{c_1, c_2\}$ can be released if, and only if, the input monomers are present.

Note that the designation of a monomer as being a module monomer (or more specifically an output monomer) or input monomer is with respect to a particular reaction. For example, consider the reactions $R_1: A + B \rightarrow C$ and $R_2: C \rightarrow A + D$. In this case, $I_{R_1} \cap M_{R_2} = \{\{a_1, a_2\}\}$.

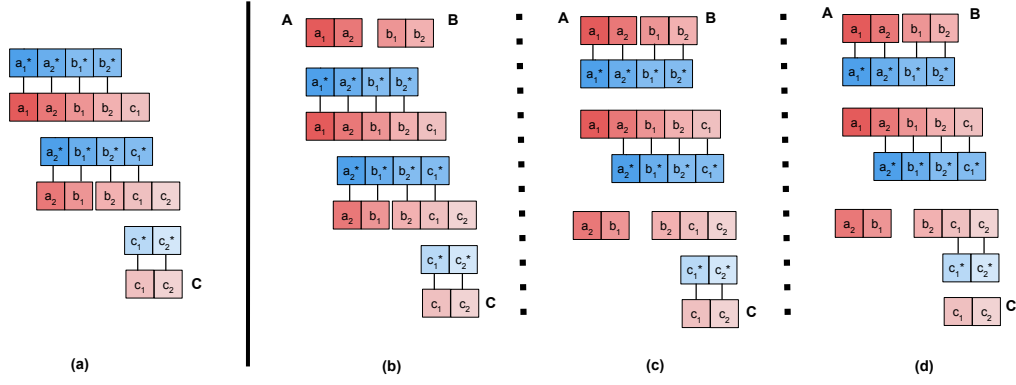


Figure 1 (a) Untriggered modular configuration of the module for the reaction $A + B \longrightarrow C$; (b), (c), (d) three stable configurations when both A and B are present.

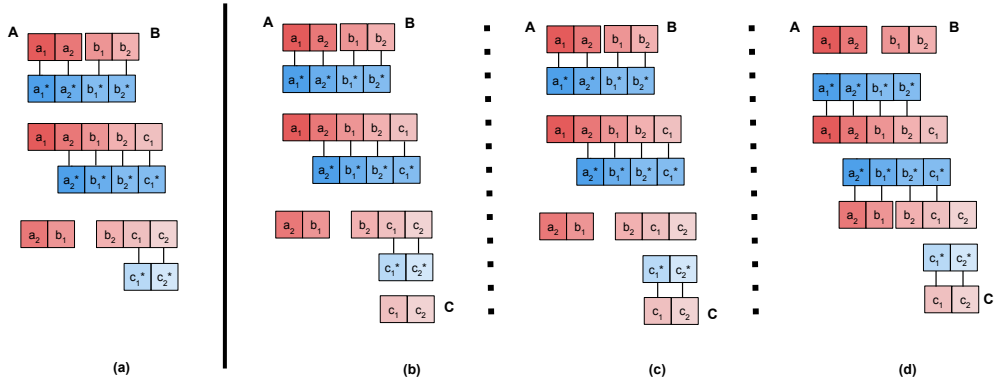


Figure 2 (a) Untriggered modular configuration of the module for $C \longrightarrow A + B$; (b), (c), (d) three stable configurations when C is present.

For a given list of reactions, we formally define the overall TBN network by adding multiplicities to modules and input monomers.

► **Definition 2** (Signal Propagation Network). *Let $\mathcal{R} = (R_1, R_2, \dots, R_n)$ be a list of reactions. Let $I = \bigcup_i I_{R_i}$ denote the set of input monomer types corresponding to the reactions in $\mathcal{R}$. We define the TBN whose set of monomers consists of:*

1. m_i copies of each module monomer in M_{R_i} , for every $i \in [n]$;
2. input monomers with multiplicities specified by the vector $k_I \in \mathbb{N}^{|I|}$.

We denote this TBN by $\text{SPN}(\mathcal{R}, ((m_i, M_i)_{i \in [n]}), k_I)$, where SPN stands for Signal Propagation Network.

4 Characterization of the Stable Configurations of a Signal Propagation Network

We determine the entropy of stable configurations of an arbitrary SPN and fully characterize the stable configurations of SPNs whose underlying reaction sets satisfy certain conditions. Observe that, in an SPN , the multiplicity of each unstarred site is at least as large as the multiplicity of its corresponding starred site. Therefore, SPNs are star-limiting.

Moreover, since each monomer in an SPN consists either entirely of starred sites (a *star monomer*) or entirely of unstarred sites (an *unstar monomer*), it follows that in a self-saturated polymer, every starred site of a star monomer must form a bond with an unstarred site of an unstar monomer. As any saturated configuration consists of self-saturated polymers, a star monomer cannot be free in such a configuration and the entropy of a saturated configuration is upper bounded by the total number of unstar monomers in the network. A polymer containing d unstar monomers decreases this upper bound by $d - 1$. We say that such a polymer performs $d - 1$ unstar merge operations. Then, the entropy of a saturated configuration is equal to the value obtained by subtracting the total number of unstar merge operations across all polymers in the configuration from the total number of unstar monomers in the network.

This observation enables us to derive a lower bound on the entropy of any stable configuration. We denote the multiset of unstar monomers contained in the SPN as U_{SPN} and that of star monomers as S_{SPN} . The cardinalities of these multisets are denoted as $|U_{\text{SPN}}|$ and $|S_{\text{SPN}}|$, respectively.

► **Lemma 3.** *If α is a stable configuration of $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$, then $S(\alpha) \geq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.*

Proof. The proof is by constructing a saturated configuration α' that has entropy of $S(\alpha') = |U_{\text{SPN}}| - \sum_{i=1}^n m_i$. Consider the configuration α' that consists of m_i -many untriggered modular configurations for each reaction R_i , as in Figure 1 and Figure 2 Part (a), and free input monomers, i.e. polymers by themselves. Note that each polymer in an untriggered modular configuration is self-saturated. Hence, α' is saturated.

Furthermore, each untriggered modular configuration has exactly one polymer which consists of two unstar monomers, while all other polymers include exactly one unstar monomer. Each such polymer decreases the maximum possible entropy $|U_{\text{SPN}}|$ by 1. As there are m_i -many of them in α' , the result follows. ◀

Next, we show that the quantity $|U_{\text{SPN}}| - \sum_{i=1}^n m_i$ provides an upper bound for the entropy of a stable configuration of $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$. To do so, we rely on the notion of a Hilbert basis. Following the characterization of unsplittable polymers in TBNs

established in [9], we refer to the collection of unsplittable polymers as the Hilbert basis of the construction. Intuitively, these polymers serve as the fundamental building blocks of the system, since every self-saturated polymer can be expressed as a nonnegative integer combination of unsplittable polymers. In a stable configuration of any TBN – and therefore in an SPN – each polymer must be unsplittable. If a polymer were splittable, the entropy of the configuration could be increased by 1, without harming saturation, by splitting it. We focus on configurations consisting solely of unsplittable polymers because they include all stable configurations. Consequently, finding an upper bound on the entropy of every unsplittable configuration also yields an upper bound on the entropy of all stable configurations.

At this point, there are two obstacles to naively computing the Hilbert basis of an SPN: (1) Since the set of reactions of an SPN can vary arbitrarily from one instance to another, we would need to compute the Hilbert basis separately for each network. This prevents us from deriving general upper bounds in terms of the parameters of an SPN. (2) We would also need to compute the Hilbert bases of large networks with complex binding structures – for example, networks composed of many modules – despite the fact that computing a Hilbert basis is already a computationally hard problem.

Instead, we introduce a novel method based on a reduction map at the monomer level that takes an SPN and produces a system with a fixed number of monomer types, regardless of the modules included in the original SPN. The induced mapping at the configuration level preserves saturation. We then argue about upper bounds on the entropy of a saturated configuration for an arbitrary SPN by analyzing this reduced system.

We define a mapping f , which we call the *reduction map*, that takes monomers from the SPN as input and outputs a monomer in the following way:

1. For $m \in U_{\text{SPN}}$, $f(m) = \{x, x, \dots, x\}$ where $|\{x, x, \dots, x\}| = \text{number of 1-unstarred sites in } m$.
2. For $m \in S_{\text{SPN}}$, $f(m) = \{x^*, x^*, \dots, x^*\}$ where $|\{x^*, x^*, \dots, x^*\}| = \text{number of 1-starred sites in } m$.

The binding site x included in the reduced network is different from every other binding site in the SPN. Note that the reduction map outputs the same monomer for two monomers in SPN as long as the number of 1-unstarred (starred) sites in these monomers is the same, regardless of which reaction's set of module/input monomers these are from.

As an example, for the reaction $R : A + B \rightarrow C$, the output monomers are shown below:

- $f(\{a_1, a_2, b_1, b_2, c_1\}) = \{x, x, x\}$
- $f(\{a_1^*, a_2^*, b_1^*, b_2^*\}) = f(\{a_2^*, b_1^*, b_2^*, c_1^*\}) = \{x^*, x^*\}$
- $f(\{a_2, b_1\}) = f(\{b_2, c_1, c_2\}) = f(\{c_1, c_2\}) = f(\{a_1, a_2\}) = f(\{b_1, b_2\}) = \{x\}$
- $f(\{c_1^*, c_2^*\}) = \{x^*\}$

The image of the monomers of an arbitrary SPN under f consists of exactly four monomers: $\{\{x\}, \{x, x, x\}, \{x^*\}, \{x^*, x^*\}\}$, where the multiplicities of these monomers in the output network are determined by the multiplicities of module monomers and input monomers of the SPN. We refer to these monomers as U1, U3, S1, S2, respectively, based on the number of unstarred or starred sites included in them.

Furthermore, the map f at the monomer level immediately defines mappings for polymers and configurations of the underlying SPN:

1. for a polymer P , $f_{\text{poly}}(P) = \{f(m) | m \in P\}$
2. for a configuration α , $f_{\text{conf}}(\alpha) = \{f_{\text{poly}}(P) | P \in \alpha\}$

10:8 Modularity of Signal Propagation Networks in the TBN Model

Note that f abstracts away binding site types by mapping every 1-unstarred site to a generic unstarred binding site x , and every 1-starred site to a generic starred site x^* , independently of the original binding site type. Since SPN contains at least as many 1-unstarred sites as 1-starred sites for each binding site type, the reduced network is also star-limiting. Consequently, the notion of self-saturation is preserved in the reduced network.

Moreover, if a polymer in SPN is self-saturated, then every 1-starred site in the polymer must be bound to the corresponding 1-unstarred site of the same binding site type. Therefore, the image of a self-saturated polymer under f_{poly} is also a self-saturated polymer in the reduced network. It follows that every saturated configuration of SPN is mapped to a saturated configuration of the reduced network under f_{conf} .

Furthermore, since each monomer in SPN has an image under f , every polymer has a corresponding image under f_{poly} , i.e. there is no polymer of the SPN without an image. Consequently, for any configuration α of SPN , we have $S(\alpha) = S(f_{\text{conf}}(\alpha))$. Combining this with the fact that saturation is preserved under f_{conf} , we obtain the following lemma.

► **Lemma 4.** *An upper bound on the entropy of all saturated configurations of the reduced network is also an upper bound on the entropy of all saturated configurations of SPN .*

To find an upper bound on the entropy of a saturated configuration of the reduced network, we will first compute the unsplittable polymers of the network, i.e., its Hilbert basis. As the number of monomer types in the reduced network is only four, regardless of the types of reactions in the underlying SPN , this is a fairly easy computation and can even be done by hand in a few minutes. Alternatively, the software tools `TBNExplorer2` [13] or `stable_tbn` [8] can compute the Hilbert basis directly from the monomers of the reduced network. The eleven unsplittable polymers of the reduced network are shown in Figure 3.

Next, by arguing about the upper bound on the entropy of an unsplittable configuration of the reduced network corresponding to the parameters of the SPN , we will show that $|U_{\text{SPN}}| - \sum_{i=1}^n m_i$ also acts as an upper bound on the entropy of a saturated configuration of an SPN .

► **Lemma 5.** *If α is a saturated configuration of an $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$, $S(\alpha) \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.*

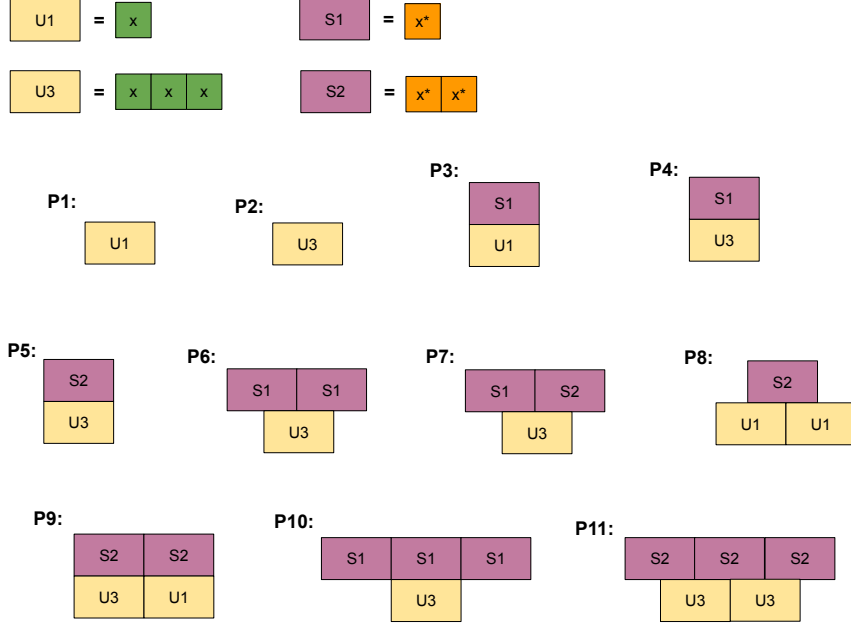
Proof. We show that $\sum_{i=1}^n m_i$ serves as an upper bound on the entropy of any saturated configuration of the reduced network. By Lemma 4, this bound also applies to the entropy of any saturated configuration of SPN . As each module corresponding to a reaction has exactly one monomer mapped to a U3 monomer and exactly two monomers that are each mapped to an S2, the number of U3 monomers in the reduced network is equal to $\sum_{i=1}^n m_i$ and the number of S2 monomers is equal to $2 \sum_{i=1}^n m_i$.

Recall that all stable configurations lie in the set of configurations consisting solely of unsplittable polymers. Therefore, it suffices to consider only such configurations. Take an arbitrary saturated configuration α' of the reduced network which consists only of unsplittable polymers by assigning counts $l_1, l_2, \dots, l_{11}$ to the 11 unsplittable polymers, $P_1, P_2, \dots, P_{11}$ respectively, shown in Figure 3.

The number of S2 monomers is:

$$l_5 + l_7 + l_8 + 2l_9 + 3l_{11} = 2 \sum_{i=1}^n m_i$$

whereas the number of U3 monomers is:



■ **Figure 3** Unsplittable polymers (Hilbert basis) of the reduced network.

$$l_2 + l_4 + l_5 + l_6 + l_7 + l_9 + l_{10} + 2l_{11} = \sum_{i=1}^n m_i$$

If we subtract the second equation from the first equation:

$$l_8 + l_9 + l_{11} - l_2 - l_4 - l_6 - l_{10} = \sum_{i=1}^n m_i \quad (1)$$

which implies

$$l_8 + l_9 + l_{11} \geq \sum_{i=1}^n m_i$$

Note that only polymers P_8 , P_9 , and P_{11} merge two unstar monomers, each of which reduces the entropy by 1 from the maximum possible value, $|U_{\text{SPN}}|$, upon polymer formation. Hence, $S(\alpha') = |U_{\text{SPN}}| - (l_8 + l_9 + l_{11}) \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.

If we take an arbitrary saturated configuration α of the SPN, $S(\alpha) = S(f_{\text{conf}}(\alpha)) \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$ as $f_{\text{conf}}(\alpha)$ is a saturated configuration of the reduced network. ◀

► **Theorem 6.** *If α is a stable configuration of an SPN $(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$, $S(\alpha) = |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.*

Proof. By Lemma 3, $S(\alpha) \geq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$ and by Lemma 5, $S(\alpha) \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$ and hence, $S(\alpha) = |U_{\text{SPN}}| - \sum_{i=1}^n m_i$. ◀

10:10 Modularity of Signal Propagation Networks in the TBN Model

Our technique not only determines the entropy of a stable configuration but also lets us reason about the images of its polymers under f_{poly} .

► **Lemma 7.** *If α is a stable configuration of a given SPN , for each polymer $P \in \alpha$, there exists $1 \leq i \leq 11$ s.t. $f_{\text{poly}}(P) = P_i$ where P_i is one of the polymers in Figure 3.*

Proof. Suppose α is a stable configuration and there exists $P \in \alpha$ such that $f_{\text{poly}}(P) = P'$ where P' is not one of the polymers in Figure 3. Then by the definition of polymer/Hilbert basis, P' is a splittable polymer which implies the existence of a saturated configuration α' of the reduced network such that $S(\alpha) + 1 = S(f_{\text{conf}}(\alpha)) + 1 \leq S(\alpha')$ as α' can be obtained from $f_{\text{conf}}(\alpha)$ with at least one split operation. As shown in the proof of Lemma 5, the entropy of an arbitrary saturated configuration of the reduced network satisfies $S(\alpha') \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.

By using these two facts, we obtain $S(\alpha) < S(\alpha') \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$ and by Theorem 6, α cannot be a stable configuration. Contradiction. ◀

As Lemma 7 restricts the images of polymers of a stable configuration to the unsplittable polymers of the reduced network, we can further eliminate more images:

► **Lemma 8.** *If α is a stable configuration of a given SPN , for each polymer $P \in \alpha$, $f_{\text{poly}}(P) \in \{P_1, P_3, P_5, P_8, P_9\}$.*

Proof. If a monomer is mapped under f to an S1 monomer, it must contain exactly 2 starred sites; if it is mapped to an S2 monomer, it must contain exactly 4 starred sites; and if it is mapped to a U3 monomer, it must contain exactly 5 unstarred sites. Then, if $f_{\text{poly}}(P) = P_7$, P has at least one exposed starred site, which implies that P is not self-saturated. Similarly, if $f_{\text{poly}}(P) = P_{11}$, P has at least two exposed starred sites, which implies that P is not self-saturated. Hence, if there exists $P \in \alpha$ such that $f_{\text{poly}}(P) \in \{P_7, P_{11}\}$, α includes a polymer that is not self-saturated; hence α cannot be a saturated configuration, which implies that α is not a stable configuration.

Secondly, let the count of P_i in $f_{\text{conf}}(\alpha)$, where P_i is one of the polymers in Figure 3, be l_i . Suppose there exists $P \in \alpha$ such that $f_{\text{poly}}(P) \in \{P_2, P_4, P_6, P_{10}\}$. Then, at least for one $i \in \{2, 4, 6, 10\}$, $l_i > 0$. This implies, along with Equation (1) in Lemma 5, $l_8 + l_9 + l_{11} > \sum_{i=1}^n m_i$. Hence, $S(\alpha) = S(f_{\text{conf}}(\alpha)) < |U_{\text{SPN}}| - \sum_{i=1}^n m_i$ which, by Theorem 6, contradicts α being a stable configuration. ◀

Note that Theorem 6 assesses the entropy of a stable configuration and Lemma 8 shows the possible images of polymers in it for an SPN with an arbitrary set of reactions. From now on, we impose the following restrictions on the set of reactions of an SPN :

- Two distinct reactions cannot share more than one species. For example, the pairs of reactions $A + B \rightarrow C$ and $C + A \rightarrow B$, as well as $A + B \rightarrow C$ and $B \rightarrow D + A$, are both forbidden. The only exception is when the reactions are reverses of each other; for instance, $A + B \rightarrow C$ and $C \rightarrow A + B$ are allowed.
- A reaction cannot include the same species more than once, e.g. $A + A \rightarrow C$, $C + A \rightarrow A$, $A \rightarrow A + C$ or $A + A \rightarrow A$ are all forbidden.

The next lemma shows that, with these restrictions, the set of possible image polymers can be further restricted:

► **Lemma 9.** *Let α be a stable configuration of a given SPN . If $\mathcal{R}$ satisfies the restrictions above, $f_{\text{poly}}(P) \in \{P_1, P_3, P_5, P_8\}$ for each polymer $P \in \alpha$.*

Next, we will characterize the actual form of the polymers in a stable configuration of an SPN using the image polymers and the restrictions on the reactions. As a consequence of Lemma 9, only the polymers with image P_8 can include more than two monomers.

► **Lemma 10.** *In a stable configuration of the SPN including the reaction $A + B \rightarrow C$ or $C \rightarrow A + B$, if there is a polymer that includes the monomer $\{a_2^*, b_1^*, b_2^*, c_1^*\}$, it is either $\{\{a_2^*, b_1^*, b_2^*, c_1^*\}, \{a_2, b_1\}, \{b_2, c_1, c_2\}\}$ or $\{\{a_2^*, b_1^*, b_2^*, c_1^*\}, \{a_1, a_2, b_1, b_2, c_1\}\}$.*

► **Lemma 11.** *In a stable configuration of the SPN including the reaction $A + B \rightarrow C$ or $C \rightarrow A + B$, if there is a polymer including $\{a_1^*, a_2^*, b_1^*, b_2^*\}$, it is either $\{\{a_1^*, a_2^*, b_1^*, b_2^*\}, \{a_1, a_2, b_1, b_2, c_1\}\}$ or it includes three monomers: $\{a_1^*, a_2^*, b_1^*, b_2^*\}$ and two unstar monomers, one of which includes binding sites a_1 and a_2 , and the other includes b_1 and b_2 .*

Lemma 9, Lemma 10 and Lemma 11 also imply the form of polymers including the monomer $\{a_1, a_2, b_1, b_2, c_1\}$ and the ones including $\{a_2, b_1\}$.

► **Corollary 12.** *In a stable configuration of an SPN that includes the reaction $A + B \rightarrow C$ or $C \rightarrow A + B$, the module monomer $\{a_1, a_2, b_1, b_2, c_1\}$ is in a polymer either with the monomer $\{a_1^*, a_2^*, b_1^*, b_2^*\}$ or $\{a_2^*, b_1^*, b_2^*, c_1^*\}$.*

Proof. Lemma 9 shows that in a stable configuration, the only polymer that includes $\{a_1, a_2, b_1, b_2, c_1\}$ has image P_5 , i.e. the monomer cannot be free. Lemma 10 and Lemma 11 imply that those polymers can only be of the form $\{\{a_1, a_2, b_1, b_2, c_1\}, \{a_1^*, a_2^*, b_1^*, b_2^*\}\}$ or $\{\{a_1, a_2, b_1, b_2, c_1\}, \{a_2^*, b_1^*, b_2^*, c_1^*\}\}$. ◀

► **Corollary 13.** *In a stable configuration of an SPN that includes a reaction $A + B \rightarrow C$ or $C \rightarrow A + B$, the module monomer $\{a_2, b_1\}$ is either in a polymer with $\{a_2^*, b_1^*, b_2^*, c_1^*\}$ and $\{b_2, c_1, c_2\}$ or free.*

Proof. Lemma 9 implies that $\{a_2, b_1\}$ is either free or in a polymer with image P_8 . By Lemma 10 and Lemma 11, the only polymer with image P_8 containing $\{a_2, b_1\}$ is $\{\{a_2^*, b_1^*, b_2^*, c_1^*\}, \{a_2, b_1\}, \{b_2, c_1, c_2\}\}$. ◀

In light of Lemma 10, Lemma 11, Corollary 12 and Corollary 13, we extend the notion of an untriggered modular configuration. For the reaction $A + B \rightarrow C$, we define a modular configuration to be a *generalized untriggered modular configuration* if the monomer $\{c_1^*, c_2^*\}$ in Figure 1 (a) forms a polymer, which has image P_3 , with an unstar monomer that includes the binding sites c_1 and c_2 , i.e., a monomer of the form $\{c_1, c_2\}$ or $\{x_2, c_1, c_2\}$. On the other hand, for the reaction $C \rightarrow A + B$, we define a modular configuration to be a *generalized untriggered modular configuration* if the monomer $\{a_1^*, a_2^*, b_1^*, b_2^*\}$ in Figure 2 (a) forms a polymer, which has image P_8 , with two unstar monomers, one including binding sites a_1 and a_2 and the other including b_1 and b_2 , and the monomer $\{c_1^*, c_2^*\}$ forms a polymer, which has image P_3 , with an unstar monomer including the binding sites c_1 and c_2 . Observe that the untriggered modular configuration of a reaction is also a generalized untriggered modular configuration for that reaction. We denote a generalized untriggered modular configuration of a reaction R by C_R^{gen} . Next, we will characterize the structure of every stable configuration of an SPN. First, we show the exact number of generalized untriggered modular configurations in a stable configuration as subconfigurations.

► **Lemma 14.** *Let $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$ be given. For each pair of reactions R_i and $R_{i'}$ that are reverses of each other, e.g. $A + B \rightarrow C$ and $C \rightarrow A + B$, the total number of $C_{R_i}^{\text{gen}}$ and $C_{R_{i'}}^{\text{gen}}$ as subconfigurations is exactly $m_i + m_{i'}$ in every stable configuration of the SPN.*

10:12 Modularity of Signal Propagation Networks in the TBN Model

Proof. W.l.o.g $R_i : A + B \longrightarrow C$ and $R_{i'} : C \longrightarrow A + B$ with m_i and $m_{i'}$ copies of module monomers for R_i and $R_{i'}$, possibly $m_i = 0$ or $m_{i'} = 0$. In total, SPN includes $m_i + m_{i'}$ -many copies of each monomer $\{a_1, a_2, b_1, b_2, c_1\}$, $\{a_1^*, a_2^*, b_1^*, b_2^*\}$ and $\{a_2^*, b_1^*, b_2^*, c_1^*\}$.

By Lemma 11, the monomer $\{a_1^*, a_2^*, b_1^*, b_2^*\}$ is either in a polymer with two unstar monomers for which one includes binding sites a_1 and a_2 the other includes b_1 and b_2 (with image P_8) or in a polymer with $\{a_1, a_2, b_1, b_2, c_1\}$ (with image P_5). Let the number of the former in a stable configuration α be $t_{i'}$ and that of the latter be t_i , such that $t_i + t_{i'} = m_i + m_{i'}$. As there are $t_{i'}$ -many $\{a_1, a_2, b_1, b_2, c_1\}$ left to consider, Corollary 12 implies that there are $t_{i'}$ -many polymers of the form $\{\{a_1, a_2, b_1, b_2, c_1\}, \{a_2^*, b_1^*, b_2^*, c_1^*\}\}$. This, with Lemma 10, implies that there are t_i -many polymers of the form $\{\{a_2^*, b_1^*, b_2^*, c_1^*\}, \{a_2, b_1\}, \{b_2, c_1, c_2\}\}$. Furthermore, by Corollary 13, there are exactly $t_{i'}$ -many free monomers of the form $\{a_2, b_1\}$.

Lastly, we consider the polymers including the monomer $\{c_1^*, c_2^*\}$. By Lemma 9, the only image polymer that can include $\{c_1^*, c_2^*\}$ is P_3 . By restrictions, there are only two possible polymers $\{\{c_1, c_2\}, \{c_1^*, c_2^*\}\}$ and $\{\{x_2, c_1, c_2\}, \{c_1^*, c_2^*\}\}$, where x_2 could be of any species other than C , with image P_3 . The total number $m_i + m_{i'}$ of $\{c_1^*, c_2^*\}$ in the modules for R_i and $R_{i'}$ implies that one can always consider t_i -many and $t_{i'}$ -many of these polymers separately. By combining these results, α has exactly t_i -many $C_{R_i}^{gen}$ and $t_{i'}$ -many $C_{R_{i'}}^{gen}$ as subconfigurations. As α is arbitrary, the lemma follows. $\blacktriangleleft$

Note that, in the proof of Lemma 14, for an arbitrary stable configuration, we consider the forms of every polymer that include a star monomer. As starred sites are limiting in the network, any polymer in a stable configuration that does not include a star monomer is necessarily a free unstar monomer. Each free unstar monomer of the form $\{a_2, b_1\}$ is already considered in a generalized untriggered modular configuration. This characterizes the structure of any stable configuration of an SPN:

► **Corollary 15.** *Every stable configuration of an SPN, whose underlying set of reactions satisfies the above restrictions, consists precisely of subconfigurations of the form C_R^{gen} for each reaction R , appearing with multiplicities as specified in Lemma 14, together with free unstarred monomers of the form $\{a_1, a_2\}$ or $\{x_2, a_1, a_2\}$.*

5 Simulating Chemical Reaction Networks with Signal Propagation Networks

► **Definition 16** (Discrete Chemical Reaction Network). *A discrete chemical reaction network (CRN) is a pair $\mathcal{C} = (\Lambda, \mathcal{R})$, where*

- $\Lambda = \{X_1, \dots, X_d\}$ is a finite set of species, and
- $\mathcal{R} \subset \mathbb{Z}^d$ is a finite set of reactions.

Each reaction $r \in \mathcal{R}$ is described as $r = \beta_r - \alpha_r$ where $\alpha_r, \beta_r \in \mathbb{N}^d$ are vectors called the reactant and product stoichiometries, respectively.

A configuration of the system is a vector $x \in \mathbb{N}^d$, where x_i represents the number of molecules of species X_i . A reaction r is enabled at configuration x if $x \geq \alpha_r$ (componentwise). If reaction r occurs, the system moves from configuration x to $x' = x - \alpha_r + \beta_r$. This is written as $x \xrightarrow{r} x'$.

A reaction $r = \beta_r - \alpha_r$ of a CRN can also be represented in stoichiometric form R , e.g., CRN $\mathcal{C} = (\{A, B, C, \dots\}, \{(-1, -1, 1, 0, 0, \dots, 0), \dots\})$ includes a reaction R of the form $A + B \longrightarrow C$. As the reaction modules are reversible in the sense described in Section 3,

we consider reversible discrete CRNs where for each reaction, the reverse reaction is also in the set, i.e., for all $r \in \mathcal{R}$, $-r \in \mathcal{R}$. From now on, we use CRN to refer to discrete reversible CRNs.

There is a natural correspondence between SPNs and CRNs. Every $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$ for which $\mathcal{R}$ includes both the forward and backward directions for each reaction induces a CRN $\mathcal{C} = (\Lambda, \mathcal{R})$, where $\Lambda \leftrightarrow \bigcup_i I_{R_i} = I$, i.e., the input species of the CRN correspond exactly to the input monomer types of the SPN. As $\mathcal{R}$ is reversible, I includes every input and output monomer in the SPN. We refer to the CRN that is constructed from an SPN in this way as the *induced CRN*. Similarly, each CRN corresponds to a family of SPNs that share the same set of reactions – including the associated module and input monomers for each reaction – and differ only in the quantities assigned to those monomers.

Furthermore, a natural mapping takes an SPN configuration and outputs the number of free input monomers – i.e., those that exist as single-monomer polymers, for each input monomer type. We will use a slightly more generalized mapping, consistent with the generalization of untriggered modular configuration, which outputs not only the number of free monomers of the form $\{a_1, a_2\}$ for species A but also those of the form $\{x_2, a_1, a_2\}$. Note that the monomer $\{x_2, a_1, a_2\}$ is included as a module monomer of a reaction of the form $_ + X \rightarrow A$ or $A \rightarrow _ + X$. For a given SPN, we denote this mapping as ϕ_{SPN} . Next, we show that, under mapping ϕ_{SPN} , the behavior of a CRN can be explained by the thermodynamically favorable configurations of a suitable SPN.

The theorem below gives one direction of a simulation desideratum in the sense that any path of a CRN can be mimicked by the corresponding SPN.

► **Theorem 17.** *For every reversible CRN $\mathcal{C} = (\Lambda, \mathcal{R})$, where $\mathcal{R}$ satisfies the restrictions defined in Section 4, every initial configuration $x_{\text{init}} \in \mathbb{N}^{|\Lambda|}$ of the CRN, and every $t \in \mathbb{N}$, there exists an $\text{SPN}(\mathcal{R}, ((m_i, M_i)_{i \in [n]}, (k_I)))$ and a stable configuration α of the SPN such that $\phi_{\text{SPN}}(\alpha) = x_{\text{init}}$, with the following property: for every configuration c of the CRN that is reachable from x_{init} by a path of length at most t , there exists a stable configuration β of the SPN such that $\phi_{\text{SPN}}(\beta) = c$, and β is reachable from α by a height-2 path.*

Proof. For an arbitrary initial configuration x_{init} of CRN $\mathcal{C} = (\Lambda, \mathcal{R})$ and a given $t \in \mathbb{N}$, construct $\text{SPN}(\mathcal{R}, (t, M_i)_{i \in [n]}, x_{\text{init}})$. We consider a configuration α of the SPN where all the input monomers with counts in x_{init} are free polymers and the rest of the configuration consists of t -many untriggered modular configurations for each reaction $R \in \mathcal{R}$. First, note that $\phi_{\text{SPN}}(\alpha) = x_{\text{init}}$. Secondly, as each untriggered modular configuration performs one merge of unstar monomers, which causes a drop of 1 from the maximum possible entropy, $|U_{\text{SPN}}|$, $S(\alpha) = |U_{\text{SPN}}| - tn$. By Theorem 6, α is a stable configuration.

Let c be a CRN configuration s.t. there exists a path $x_{\text{init}} = x_0 \xrightarrow{R_{i_0}} x_1 \xrightarrow{R_{i_1}} x_2 \xrightarrow{R_{i_2}} \dots \xrightarrow{R_{i_{t'-1}}} c = x_{t'-1}$ where $t' \leq t$ and each step corresponds to a transition implied by a reaction $R_{i_k} \in \mathcal{R}$. Consider the transitions in the SPN such that at each step k , a previously unfired reaction module for R_{i_k} is fired. Let the corresponding sequence of SPN configurations be $\alpha \rightarrow \gamma_1 \rightarrow \gamma_2 \rightarrow \dots \rightarrow \gamma_{t'-2} \rightarrow \beta$. As α includes t -many reaction modules for each $R \in \mathcal{R}$ and $t' \leq t$, it is always possible to find an unfired module along the sequence. Hence, $\phi_{\text{SPN}}(\gamma_j) = x_j$ for $j \in \{1, 2, \dots, t' - 1\}$ and $\phi_{\text{SPN}}(\beta) = c$. Note that firing a reaction module does not change the total number of unstar merges in the system, as each reaction module includes exactly one polymer merging two unstar monomers (Figure 1 (b) to (d) and Figure 2 (b) to (d)). Recall that the entropies of two configurations are equal if and only if the total number of unstar merges is the same in the configurations. Hence, $S(\alpha) = S(\gamma_1) = S(\gamma_2) = \dots = S(\gamma_{t'-1}) = S(\beta)$. As firing a reaction results in another

10:14 Modularity of Signal Propagation Networks in the TBN Model

saturated configuration ((Figure 1 (d) and Figure 2 (d)) and α is a stable configuration, $\gamma_1, \gamma_2, \dots, \gamma_{t'-1}$ and β are all stable configurations. Finally, as firing each module necessitates a separate height-2 path, the overall path that takes α to β is also a height-2 path. ◀

Next, we consider the reverse direction and show that the corresponding images, under ϕ_{SPN} , of each pair of stable configurations of an SPN are reachable, with an appropriate definition of reachability, from one another in the induced CRN.

► **Definition 18** (Weak reachability in CRNs). *Let a CRN $\mathcal{C} = (\Lambda, \mathcal{R})$ where $\mathcal{R} = (r_1, r_2, \dots, r_k)$ be given. We consider two configurations c_1 and c_2 of $\mathcal{C}$ and say c_2 is weakly reachable from c_1 if there exists $\gamma_1, \gamma_2, \dots, \gamma_k \in \mathbb{N}$ s.t. $c_2 = c_1 + \sum_{i=1}^k \gamma_i r_i$.*

Our definition of weak reachability corresponds to being in the same stoichiometric compatibility class as defined in the Chemical Reaction Network Theory literature [1].

When the CRN is reversible, c_2 being weakly reachable from c_1 also implies c_1 being weakly reachable from c_2 . Next, we show that the stable configurations of an arbitrary SPN correspond to configurations of the induced CRN that are weakly reachable.

► **Theorem 19.** *For an arbitrary SPN $(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$, where $\mathcal{R}$ satisfies the restrictions in Section 4, let $\mathcal{C} = (\Lambda, \mathcal{R})$ be the induced CRN. For every pair α and β of stable configurations of the SPN, $\phi_{\text{SPN}}(\alpha)$ and $\phi_{\text{SPN}}(\beta)$ are weakly reachable in the CRN.*

Proof. For $0 < i \leq \frac{n}{2}$, let $R_i : X_{i_1} + X_{i_2} \longrightarrow X_{i_3}$ and $R_{i'} : X_{i_3} \longrightarrow X_{i_1} + X_{i_2}$, where $X_{i_1}, X_{i_2}, X_{i_3}$ denote the input species with indices $i_1, i_2, i_3 \in [|\Lambda|]$. Let g_i and $g_{i'}$ be the amount of subconfigurations of $C_{R_i}^{\text{gen}}$ and $C_{R_{i'}}^{\text{gen}}$ in α and let h_i and $h_{i'}$ be those in β . By Lemma 14, $g_i + g_{i'} = h_i + h_{i'} = m_i + m_{i'}$, which implies $g_i - h_i = h_{i'} - g_{i'}$. Note that Lemma 14 also holds when $m_i = 0$ and/or $m_{i'} = 0$ and hence, if $\mathcal{R}$ is not reversible, it can be expanded to include reverse reactions without any copies of module monomers for those reactions.

We will argue based on the counts of the monomers, with image under ϕ_{SPN} when free, in α and β . Note that $\phi_{\text{SPN}}(\alpha)$ and $\phi_{\text{SPN}}(\beta)$ already capture the counts of the free monomers of the form $\{a_1, a_2\}$ and $\{x_2, a_1, a_2\}$ in these configurations. In each $C_{R_i}^{\text{gen}}$, there are exactly two monomers which are mapped to X_{i_3} and no monomers mapped to X_{i_1} or X_{i_2} under ϕ_{SPN} , when free. In each $C_{R_{i'}}^{\text{gen}}$, there are exactly three monomers mapped to X_{i_1} , X_{i_2} and X_{i_3} , respectively, under ϕ_{SPN} , when free. Hence, by the conservation of the number of monomers in α and β :

$$\phi_{\text{SPN}}(\alpha) + \sum_{i=1}^{n/2} (2g_i e_{i_3} + g_{i'} e_{i_1} + g_{i'} e_{i_2} + g_{i'} e_{i_3}) = \phi_{\text{SPN}}(\beta) + \sum_{i=1}^{n/2} (2h_i e_{i_3} + h_{i'} e_{i_1} + h_{i'} e_{i_2} + h_{i'} e_{i_3}) \quad (2)$$

where $e_{i_1}, e_{i_2}, e_{i_3}$ are the i_1 st, i_2 nd and i_3 rd unit vectors of dimension $|\Lambda|$. Using $g_i - h_i = h_{i'} - g_{i'}$ and rearranging the terms in Equation (2) yields

$$\phi_{\text{SPN}}(\alpha) + \sum_{i=1}^{n/2} ((g_i - h_i) e_{i_3} + (g_{i'} - h_{i'}) e_{i_1} + (g_{i'} - h_{i'}) e_{i_2}) = \phi_{\text{SPN}}(\beta)$$

If $g_i - h_i = -(g_{i'} - h_{i'}) \geq 0$, $((g_i - h_i) e_{i_3} + (g_{i'} - h_{i'}) e_{i_1} + (g_{i'} - h_{i'}) e_{i_2}) = (g_i - h_i) r_i$ and if $g_i - h_i = -(g_{i'} - h_{i'}) \leq 0$, $((g_i - h_i) e_{i_3} + (g_{i'} - h_{i'}) e_{i_1} + (g_{i'} - h_{i'}) e_{i_2}) = (g_{i'} - h_{i'}) r_{i'}$ where r_i and $r_{i'}$ are the CRN transitions corresponding to the reactions R_i and $R_{i'}$, respectively. The theorem follows. ◀

6 Two-reactant, two-product reversible signal propagation modules

The reversible signal propagation module studied above has been previously described and experimentally implemented [3, 17]. However, it has the limitation of having only one product (or reactant for the backward module). In Appendix A, we introduce a new reversible signal propagation module with two reactants and two products, potentially enabling a more compact realization of CRNs. Importantly, testifying to the generality of our proof technique, we show that an appropriately chosen reduction mapping can be used to determine the exact entropy of a stable configuration of an SPN with an arbitrary number of reaction modules for this system as well.

The reduction mapping we employ accounts for the total number of 1-unstarred (starred) and 2-unstarred (starred) sites in each monomer. In contrast, if one considers an alternative reduction mapping that tracks only the number of 1-unstarred (starred) sites, which gives the same reduced network if 2-unstarred (starred) sites or 3-unstarred (starred) sites are considered as the construction is symmetric, the resulting lowest upper bound on the entropy of a saturated configuration becomes trivial and thus uninformative. This demonstrates that the choice of reduction mapping for analyzing the entropy of stable configurations in an SPN is non-trivial. For space considerations, full details are deferred to Appendix A and Appendix B.

7 Discussion

We present a general method for analyzing thermodynamically favorable configurations of modular signal propagation networks, with the goal of showing that modules verified in isolation remain correct under composition. Our approach relies on constructing a reduction mapping that transforms the original system into a coarser network, enabling tight upper bounds on the entropy of saturated configurations. We show that this reduction can map networks composed of arbitrarily many reaction modules to a single reduced module.

The reduced system is small enough to make the computation of its Hilbert basis and the characterization of its stable configurations tractable, thereby avoiding the computational hardness of the original problem. Using this framework, we lift entropy bounds and structural properties from the reduced system back to the original network, establishing that modular SPNs can be safely composed. We subsequently show that under certain restrictions on the reaction set, these networks can simulate a class of reversible chemical reaction networks: every bounded-length CRN execution can be realized by a height-2 path between stable TBN configurations, and any two stable TBN configurations map to CRN states in the same stoichiometric compatibility class.

Besides applying our method to the previously described reversible signal propagation module, an additional contribution of this paper is to introduce a novel reversible signal propagation module with two inputs and two outputs. In general, we caution the reader that the choice of reduction mapping is critical. While an appropriate reduction preserves enough structure to yield tight entropy bounds, coarser reductions fail to capture essential combinatorial information and lead only to trivial bounds (as shown in Appendix B).

Several directions remain open. First, it is natural to ask whether the constructions in this paper can be extended to simulate arbitrary reversible CRNs directly without structural restriction, and how the proof technique can be adapted accordingly. Second, adding redundancy in binding-site types and monomer structure, may allow control over the distance to stability of leaked configurations [7]; we conjecture that this can be captured via extensions of our reduction-based analysis. Finally, it would be interesting to characterize all height-2 paths in an SPN and relate them to trajectories in the corresponding CRN.

References

- 1 David F Anderson, Gheorghe Craciun, and Thomas G Kurtz. Product-form stationary distributions for deficiency zero chemical reaction networks. *Bulletin of mathematical biology*, 72(8):1947–1970, 2010.
- 2 Keenan Breik, Cameron Chalk, David Doty, David Haley, and David Soloveichik. Programming substrate-independent kinetic barriers with thermodynamic binding networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 18(1):283–295, 2021. doi:10.1109/TCBB.2019.2959310.
- 3 Keenan Breik, Chris Thachuk, Marijn Heule, and David Soloveichik. Computing properties of stable configurations of thermodynamic binding networks. *Theoretical Computer Science*, 785:17–29, 2019. doi:10.1016/J.TCS.2018.10.027.
- 4 Robert Brijder. Computing with chemical reaction networks: a tutorial. *Natural Computing*, 18(1):119–137, 2019. doi:10.1007/S11047-018-9723-9.
- 5 Cameron Chalk, Jacob Hendricks, Matthew J Patitz, and Michael Sharp. Thermodynamically favorable computation via tile self-assembly. In *International Conference on Unconventional Computation and Natural Computation*, pages 16–31. Springer, 2018. doi:10.1007/978-3-319-92435-9_2.
- 6 David Doty. Theory of algorithmic self-assembly. *Communications of the ACM*, 55(12):78–88, 2012. doi:10.1145/2380656.2380675.
- 7 David Doty, Trent A Rogers, David Soloveichik, Chris Thachuk, and Damien Woods. Thermodynamic binding networks. In *International Conference on DNA-Based Computers*, pages 249–266. Springer, 2017. doi:10.1007/978-3-319-66799-7_16.
- 8 David Haley. StableTBN. https://github.com/drhaley/stable_tbn, 2021. Accessed: 2025-09-03.
- 9 David Haley and David Doty. Computing properties of thermodynamic binding networks: An integer programming approach. In Matthew R. Lakin and Petr Šulc, editors, *27th International Conference on DNA Computing and Molecular Programming (DNA 27)*, volume 205 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:16, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.27.2.
- 10 Austin Luchsinger. Brief announcement: Barrier-1 reachability for thermodynamic binding networks is PSPACE-complete. In *1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2022)*, pages 24–1. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.SAND.2022.24.
- 11 Joshua Petrack, David Soloveichik, and David Doty. Thermodynamically driven signal amplification. In Ho-Lin Chen and Constantine G. Evans, editors, *29th International Conference on DNA Computing and Molecular Programming (DNA 29)*, volume 276 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 8:1–8:22, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.29.8.
- 12 Andrew Phillips and Luca Cardelli. A programming language for composable DNA circuits. *Journal of the Royal Society Interface*, 6(suppl_4):S419–S436, 2009.
- 13 David Soloveichik. TBNEplorer2: A Python library and command-line tools for analyzing thermodynamic binding networks, 2025. URL: <https://github.com/david-soloveichik/TBNEplorer2>.
- 14 David Soloveichik, Matthew Cook, Erik Winfree, and Jehoshua Bruck. Computation with finite stochastic chemical reaction networks. *Natural Computing*, 7(4):615–633, 2008. doi:10.1007/S11047-008-9067-Y.
- 15 David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393–5398, 2010.
- 16 David Soloveichik and Capstone Team. stablegen.net: Free energies and concentrations of domain-level DNA designs. <https://stablegen.net>, 2024. Accessed: 2024-08-30.
- 17 Boya Wang, Cameron Chalk, David Doty, and David Soloveichik. Molecular computation at equilibrium via programmable entropy. *Science Advances*, 12(2):eadx3969, 2026.

- 18 And Kaan Ata Yilmaz, David Soloveichik, and Austin Luchsinger. Thermodynamic binding networks fully capture equilibrium monomer-polymer thermodynamics. In *32nd International Conference on DNA Computing and Molecular Programming (DNA32)*, 2026. doi:10.4230/LIPIcs.DNA.32.9.

A Two-reactant, two-product reversible signal propagation modules

In this section, we introduce a novel reversible signal propagation module. Unlike the system of Section 3, the new construction directly implements a reversible reaction with two reactants and two products such as $A + B \rightarrow C + D$ and thus could realize CRNs more compactly, without requiring a decomposition of such a reaction into $A + B \rightarrow X$ and $X \rightarrow C + D$. Additionally, we are motivated to demonstrate that our proof technique for showing the correctness of such modules is general enough to cover multiple distinct constructions.

The new system consists of reaction modules of the form $R : A + B \rightarrow C + D$. The *untriggered modular configuration* for this reaction module is shown in Figure 4 (a) and the two stable configurations in the presence of input monomers for the module are shown in Figure 4 (b), (c). The correspondence between species and monomers is the following: $\{x_1, x_2, x_3\} \leftrightarrow X$. As in Section 3, the fact that (a)-(c) describe the unique stable configurations of an isolated module can be easily checked via tools such as **StableGen** or **stable_tbn** [3, 9, 16, 8].

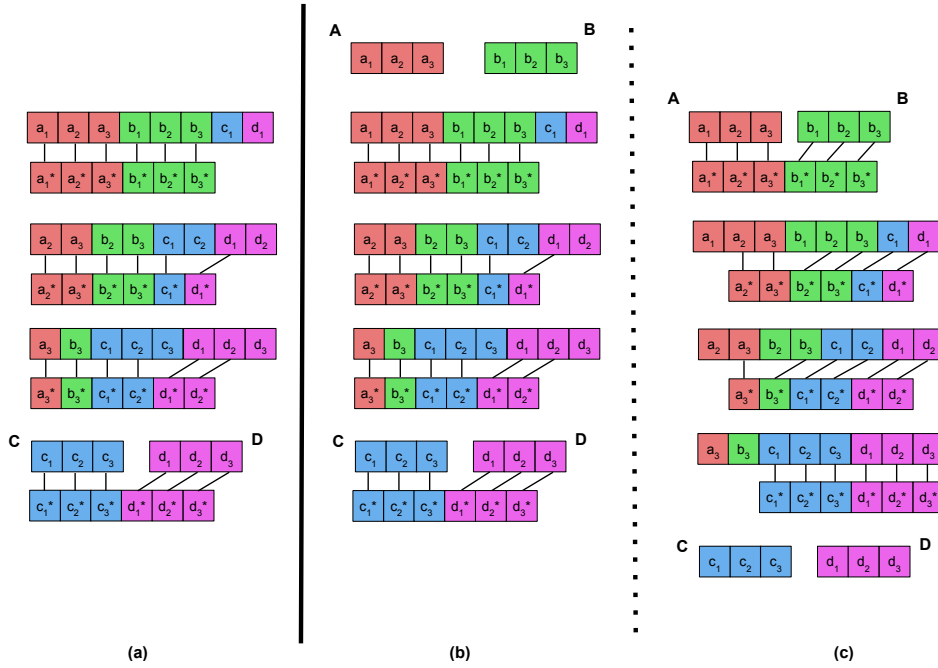
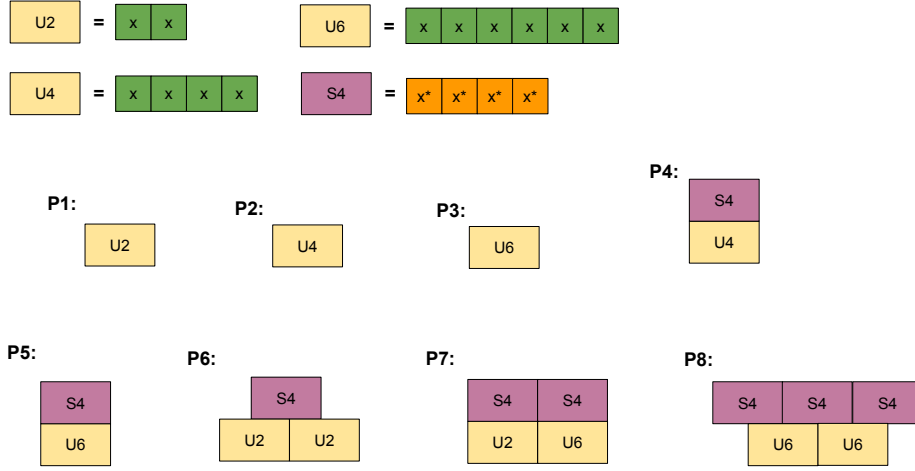


Figure 4 (a) The only stable configuration of the module for the reaction $A + B \rightarrow C + D$; (b), (c) two stable configurations when both A and B are present.

Module monomers, input monomers, output monomers and $\text{SPN}(\mathcal{R}, ((m_i, M_i)_{i \in [n]}), k_I)$, are defined analogously to the first system. The reduction map f that we use for this system outputs a monomer of length equal to the total number of 1-unstarred (starred) and 2-unstarred (starred) sites for unstar (star) monomers. Explicitly,

10:18 Modularity of Signal Propagation Networks in the TBN Model



■ **Figure 5** Unsplittable polymers (Hilbert basis) of the reduced network for the new system.

- $f(\{a_1, a_2, a_3, b_1, b_2, b_3, c_1, d_1\}) = f(\{a_2, a_3, b_2, b_3, c_1, c_2, d_1, d_2\}) = \{x, x, x, x, x, x\}$
- $f(\{a_1^*, a_2^*, a_3^*, b_1^*, b_2^*, b_3^*\}) = f(\{a_2^*, a_3^*, b_2^*, b_3^*, c_1^*, d_1^*\}) = f(\{a_3^*, b_3^*, c_1^*, c_2^*, d_1^*, d_2^*\}) = f(\{c_1^*, c_2^*, c_3^*, d_1^*, d_2^*, d_3^*\}) = \{x^*, x^*, x^*, x^*\}$
- $f(\{a_3, b_3, c_1, c_2, c_3, d_1, d_2, d_3\}) = \{x, x, x, x\}$
- $f(\{a_1, a_2, a_3\}) = f(\{b_1, b_2, b_3\}) = f(\{c_1, c_2, c_3\}) = f(\{d_1, d_2, d_3\}) = \{x, x\}$

f_{poly} and f_{conf} are defined accordingly. As each monomer type has an image under f and f preserves the total number of 1-unstarred (starred) and 2-unstarred (starred) sites of unstar (star) monomers, if a polymer P is self-saturated, $f_{\text{poly}}(P)$ is also self-saturated, which further implies that if a configuration α of the SPN is saturated, then $f_{\text{conf}}(\alpha)$ is also saturated. Moreover, $S(\alpha) = S(f_{\text{conf}}(\alpha))$. Then, we can invoke Lemma 4 and find an upper bound on the entropy of the reduced network by analyzing its unsplittable polymers. The unsplittable polymers (Hilbert basis) of the reduced network are shown in Figure 5.

Next, we will assess the entropy of a stable configuration, with respect to the parameters of the SPN in a way that is the direct analogue of Lemma 3 and Lemma 5.

► **Lemma 20.** *If α is a stable configuration of $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$, then $S(\alpha) \geq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.*

Proof. The proof follows the same lines as the proof of Lemma 3. Consider configuration α' that consists of the untriggered modular configuration of each reaction $R \in \mathcal{R}$ (Figure 4 (a)), with respective multiplicities, as subconfigurations and free input monomers. As each polymer is self-saturated, α' is saturated. Furthermore, as each untriggered modular configuration includes exactly one polymer that merges two unstar monomers, $S(\alpha') = |U_{\text{SPN}}| - \sum_{i=1}^n m_i \leq S(\alpha)$. ◀

► **Lemma 21.** *If α is a saturated configuration of an $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$, $S(\alpha) \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.*

Proof. As in Lemma 5, we will use the unsplittable polymers of the reduced network to show an upper bound on the entropy of a stable configuration of the reduced network. Take an arbitrary unsplittable configuration α' of the reduced network by assigning counts $l_1, l_2, \dots, l_8$ to the 8 unsplittable polymers, $P_1, P_2, \dots, P_8$, respectively, shown in Figure 5. Each module includes two U6, four S4 and one U4 monomer.

The number of U6 monomers is:

$$l_3 + l_5 + l_7 + 2l_8 = 2 \sum_{i=1}^n m_i \quad (3)$$

The number of S4 monomers is:

$$l_4 + l_5 + l_6 + 2l_7 + 3l_8 = 4 \sum_{i=1}^n m_i \quad (4)$$

The number of U4 monomers is:

$$l_2 + l_4 = \sum_{i=1}^n m_i \quad (5)$$

Note that only P_6, P_7 and P_8 merge two unstar monomers, each of which reduces the entropy by 1 from the maximum possible value, $|U_{\text{SPN}}|$, upon polymer formation. Hence, $S(\alpha') = |U_{\text{SPN}}| - (l_6 + l_7 + l_8)$.

If we subtract Equation (3) from Equation (4), we get $l_6 + l_7 + l_8 + l_4 - l_3 = 2 \sum_{i=1}^n m_i$, which implies $l_6 + l_7 + l_8 + l_4 \geq 2 \sum_{i=1}^n m_i$ as $l_3 \geq 0$. By Equation (5), $l_4 \leq \sum_{i=1}^n m_i$, which implies $l_6 + l_7 + l_8 \geq \sum_{i=1}^n m_i$. Hence, $S(\alpha') \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.

Therefore, for any saturated configuration α of SPN , $S(\alpha) = S(f_{\text{conf}}(\alpha)) \leq |U_{\text{SPN}}| - \sum_{i=1}^n m_i$. $\blacktriangleleft$

► **Theorem 22.** *Let α be a stable configuration of $\text{SPN}(\mathcal{R}, (m_i, M_i)_{i \in [n]}, k_I)$, then $S(\alpha) = |U_{\text{SPN}}| - \sum_{i=1}^n m_i$.*

Proof. Direct consequence of Lemma 20 and Lemma 21. $\blacktriangleleft$

B The Choice of Reduction Mapping is Non-Trivial

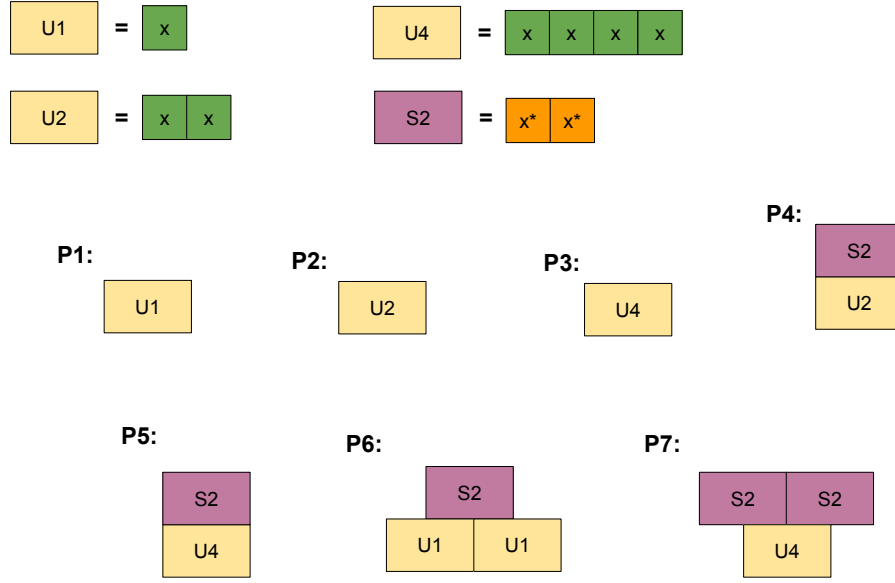
We will show that applying the same method with different reduction mappings might not always be able to assess the entropy of a stable configuration of the SPN . To show this, for the SPN consisting of modules in Figure 4, instead of the reduction mapping that outputs a monomer with length equal to the total number of 1-unstarred (starred) and 2-unstarred (starred) sites for an unstar (star) monomer, we consider f' which outputs a monomer with length equal to the number of 1-unstarred (starred) sites. Explicitly,

- $f'(\{a_1, a_2, a_3, b_1, b_2, b_3, c_1, d_1\}) = \{x, x, x, x\}$
- $f'(\{a_1^*, a_2^*, a_3^*, b_1^*, b_2^*, b_3^*\}) = f'(\{a_2^*, a_3^*, b_2^*, b_3^*, c_1^*, d_1^*\}) = f'(\{a_3^*, b_3^*, c_1^*, c_2^*, d_1^*, d_2^*\}) = f'(\{c_1^*, c_2^*, c_3^*, d_1^*, d_2^*, d_3^*\}) = \{x^*, x^*\}$
- $f'(\{a_3, b_3, c_1, c_2, c_3, d_1, d_2, d_3\}) = f'(\{a_2, a_3, b_2, b_3, c_1, c_2, d_1, d_2\}) = \{x, x\}$
- $f'(\{a_1, a_2, a_3\}) = f'(\{b_1, b_2, b_3\}) = f'(\{c_1, c_2, c_3\}) = f'(\{d_1, d_2, d_3\}) = \{x\}$

The unsplittable polymers (Hilbert basis) of the system are shown in Figure 6. We will show that, by using the reduced network induced by f' , the smallest upper bound on the entropy of a saturated configuration α is $|U_{\text{SPN}}|$:

► **Lemma 23.** *Let α' be a saturated configuration of the reduced network under f' . $S(\alpha') \leq |U_{\text{SPN}}|$ and the bound is tight, i.e., there exists a saturated α' such that $S(\alpha') = |U_{\text{SPN}}|$.*

10:20 Modularity of Signal Propagation Networks in the TBN Model



■ **Figure 6** Unsplittable polymers (Hilbert basis) of the reduced network, under f' , for the new system.

Proof. Take an unsplittable configuration α' of the reduced network by assigning counts $l_1, l_2, \dots, l_7$ to the 7 unsplittable polymers, $P_1, P_2, \dots, P_7$, shown in Figure 6. Each module includes one U4, four S2 and two U2 monomers.

The number of U4 monomers is:

$$l_3 + l_5 + l_7 = \sum_{i=1}^n m_i$$

The number of S2 monomers is:

$$l_4 + l_5 + l_6 + 2l_7 = 4 \sum_{i=1}^n m_i$$

The number of U2 monomers is:

$$l_2 + l_4 = 2 \sum_{i=1}^n m_i$$

Note that the assignment $l_2 = l_3 = l_5 = l_6 = 0$, $l_7 = \sum_{i=1}^n m_i$ and $l_4 = 2 \sum_{i=1}^n m_i$ satisfies all constraints. Consider a saturated configuration α' which consists of $(2 \sum_{i=1}^n m_i)$ -many P_4 and $(\sum_{i=1}^n m_i)$ -many P_7 , with all U1 monomers free, i.e. in polymer P_1 . As α' does not include P_6 , $S(\alpha') = |U_{\text{SPN}}| - l_6 = |U_{\text{SPN}}|$. ◀

C

 Omitted Proofs

Proof of Lemma 9. We will do a case-by-case analysis on the two possible forms of S2 monomers to show that P_9 is not an allowed image under the restrictions. Note that, as two S2 monomers include a total of eight starred sites, the same argument based on self-saturation presented in Lemma 8 can be invoked to show that U1 and U3 monomers that saturate two S2 monomers include three and five unstarred sites, respectively, i.e. an unstar monomer of the form $\{x_1, x_2\}$ cannot be included in a polymer with image P_9 and hence, the polymer has no exposed sites.

For each case, we will argue on the basis of the U1 monomer with three unstarred sites. Hence, all monomers that we consider below are included in the modules of a reaction and of its reverse. As the restrictions, when applicable, are not direction-dependent, without loss of generality, we will show contradictions for the reactions in the forward direction.

Case 1: $S2 = \{x_1^*, x_2^*, y_1^*, y_2^*\}$ and $S2 = \{t_1^*, t_2^*, z_1^*, z_2^*\}$.

The reactions that can include these monomers as their module monomers have the form: $X + Y \longrightarrow _$ and $Z + T \longrightarrow _$. The character ' $_$ ' is a placeholder for an arbitrary species (and for an arbitrary unstarred site in the following):

Case 1.1: If $U1 = \{x_2, _, y_2\}$, there exists a reaction $_ + Y \longrightarrow X$ or $_ + X \longrightarrow Y$ which are not allowed by the restrictions. A symmetric argument can be made for a monomer of the form $\{t_2, _, z_2\}$.

Case 1.2: If $U1 = \{x_2, t_1, t_2\}$, $U3 = \{y_1, y_2, z_1, z_2, x_1\}$. U3 monomer implies the existence of a forbidden reaction $Y + Z \longrightarrow X$ or $Z + Y \longrightarrow X$. The same argument can be made for a monomer of the form $\{x_2, z_1, z_2\}$, $\{y_2, z_1, z_2\}$, $\{y_2, t_1, t_2\}$, $\{z_2, x_1, x_2\}$, $\{z_2, y_1, y_2\}$, $\{t_2, x_1, x_2\}$, $\{t_2, y_1, y_2\}$.

Case 1.3: If $U1 = \{x_2, y_1, t_2\}$, $U3 = \{t_1, y_2, z_1, z_2, x_1\}$. U1 implies either $X = Y$, which is forbidden, or $Y = T$, which would imply U3 monomer being in the form $\{y_1, y_2, z_1, z_2, x_1\}$ and hence, the existence of a reaction of the form $Y + Z \longrightarrow X$ (or $Z + Y \longrightarrow X$). The same argument can be made for $\{x_2, z_1, t_2\}$, $\{x_2, t_1, z_2\}$, $\{x_2, y_1, z_2\}$, $\{y_2, z_1, t_2\}$, $\{y_2, x_1, t_2\}$, $\{y_2, x_1, z_2\}$, $\{y_2, t_1, z_2\}$.

Case 2: $S2 = \{x_1^*, x_2^*, y_1^*, y_2^*\}$ and $S2 = \{t_2^*, z_1^*, z_2^*, u_1^*\}$

The reactions that can include these monomers as their module monomers have the form $X + Y \longrightarrow _$ and $T + Z \longrightarrow U$.

Case 2.1: If U1 monomer is $\{x_2, _, y_2\}$, there exists a reaction $_ + X \longrightarrow Y$ or $_ + Y \longrightarrow X$. The same argument can be made for a monomer of the form $\{t_2, _, z_2\}$.

Case 2.2: If $U1 = \{t_2, x_1, x_2\}$, $U3 = \{y_1, y_2, z_1, z_2, u_1\}$. These forms of U1 and U3 imply two reactions $_ + T \longrightarrow X$ and $Y + Z \longrightarrow U$ (or $Z + Y \longrightarrow U$, which is already forbidden). If $T \neq Y$, the second reaction is forbidden. Else, the first reaction is forbidden. The same argument can be made for $\{t_2, y_1, y_2\}$.

Case 2.3: If $U1 = \{x_2, y_1, t_2\}$. U1 monomer implies either $X = Y$, which is forbidden or $T = Y$, which would imply the existence of a reaction $_ + X \longrightarrow Y$ that is forbidden. The same argument can be made for $\{y_2, x_1, t_2\}$, $\{x_2, z_1, t_2\}$, $\{y_2, z_1, t_2\}$, $\{y_2, u_1, t_2\}$, $\{x_2, u_1, t_2\}$.

Case 2.4 : If $U1 = \{x_2, x_1, z_2\}$, $U3 = \{y_1, y_2, z_1, t_2, u_1\}$. U3 monomer implies that either $Z = T$ or $Z = U$. The same argument can be made for $\{y_2, y_1, z_2\}$.

10:22 Modularity of Signal Propagation Networks in the TBN Model

Case 2.5: If $U1 = \{x_2, y_1, z_2\}$, $U3 = \{x_1, y_2, z_1, t_2, u_1\}$. U1 monomer implies either $X = Y$, which is forbidden, or $Y = Z$. If $Y = Z$, U3 monomer implies either $T = U$, which is forbidden, or $T = X$. $Y = Z$ and $T = X$ implies an existence of a reaction $_ + T \longrightarrow Z$, through the U1 monomer, which is forbidden. The same argument can be made for $U1 = \{y_2, x_1, z_2\}$.

Case 2.6 : If $U1 = \{x_2, z_1, z_2\}$, $U3 = \{x_1, y_1, y_2, t_2, u_1\}$. U3 monomer implies either $T = U$, which is forbidden, or $T = X$. If $T = X$, U1 monomer implies the existence of a reaction either $_ + T \longrightarrow Z$ or $_ + Z \longrightarrow T$, which are both forbidden. The same argument can be made for $\{y_2, z_1, z_2\}, \{x_2, u_1, z_2\}, \{y_2, u_1, z_2\}$.

Case 3: $S2 = \{x_2^*, y_1^*, y_2^*, z_1^*\}$ and $S2 = \{t_2^*, u_1^*, u_2^*, v_1^*\}$ The reactions that can include these monomers as their module monomers have the form $X + Y \longrightarrow Z$ and $T + U \longrightarrow V$

Case 3.1: If $U1 = \{x_2, _, y_2\}$, there exists a reaction of the form $_ + X \longrightarrow Y$ or $_ + Y \longrightarrow X$, which are both forbidden. The same argument is valid for $U1 = \{t_2, _, u_2\}$.

Case 3.2: If $U1 = \{x_2, y_1, t_2\}$, $U3 = \{y_2, z_1, u_1, u_2, v_1\}$. U1 monomer implies either $X = Y$, which is forbidden, or $Y = T$. If $Y = T$, U3 monomer implies either $Z = Y$, which is forbidden or $T = Y = V$, which is also forbidden. The same argument is valid for $\{x_2, u_1, t_2\}$.

Case 3.3: If $U1 = \{x_2, z_1, t_2\}$, $U3 = \{y_1, y_2, u_1, u_2, v_1\}$. U1 monomer implies either $X = Z$, which is forbidden or $Z = T$. If $Z = T$, the reaction $T + U \longrightarrow V$ can be written as $Z + U \longrightarrow V$, which is forbidden with the reaction implied by the U3 monomer, $Y + U \longrightarrow V$ or $U + Y \longrightarrow V$, as $Z \neq Y$. The same argument is valid for $\{x_2, v_1, t_2\}$.

Case 3.4: If $U1 = \{y_2, y_1, u_2\}$, $U3 = \{x_2, z_1, t_2, u_1, v_1\}$. U3 monomer implies that $Z = T$ as $T = U$ and $T = V$ are forbidden. If $X = V$, U3 monomer implies the existence of a reaction $T + V \longrightarrow U$ or $V + T \longrightarrow U$ which are both forbidden. If $X = U$, U1 monomer implies the existence of a reaction, $_ + X \longrightarrow Y$, which is forbidden. The same argument can be made for $\{y_2, u_1, u_2\}$.

Case 3.5: If $U1 = \{y_2, z_1, u_2\}$, $U3 = \{x_2, y_1, t_2, u_1, v_1\}$. U1 monomer implies that either $Y = Z$, which is forbidden, or $Z = U$. U3 monomer implies that $Y = T$ as $T = U$ and $T = V$ are forbidden. $Z = U$ and $Y = T$ imply the existence of a reaction $X + T \longrightarrow U$, which is forbidden. The same argument can be made for $\{y_2, v_1, u_2\}, \{x_2, y_1, u_2\}, \{t_2, u_1, y_2\}$.

Case 3.6: If $U1 = \{x_2, z_1, u_2\}$, $U3 = \{y_1, y_2, t_2, u_1, v_1\}$. U3 monomer implies either $T = U$ or $T = V$, which are both forbidden. The same argument can be made for $\{t_2, v_1, y_2\}$.

Case 3.7: If $U1 = \{x_2, v_1, u_2\}$, $U3 = \{y_1, y_2, z_1, u_1, t_2\}$. U3 monomer implies that either $U = T$, which is forbidden, or $Z = T$. If $Z = T$, U3 monomer implies the existence of a reaction $T + Y \longrightarrow U$ or $Y + T \longrightarrow U$, which are both forbidden. The same argument can be made for $\{t_2, z_1, y_2\}, \{x_2, u_1, u_2\}, \{t_2, y_1, y_2\}$. ◀

Proof of Lemma 10. The monomer $\{a_2^*, b_1^*, b_2^*, c_1^*\}$ is an S2 monomer. By Lemma 9, the monomer is in a polymer either with a U3 monomer (with image P_5) or two U1 monomers (with image P_8). For the former, the self-saturation immediately implies that U3 monomer includes binding sites a_2, b_1, b_2 , and c_1 . Then U3 monomer is necessarily $\{a_1, a_2, b_1, b_2, c_1\}$.

For the latter, we will argue by the partition, of cardinality 2, of starred sites of the monomer $\{a_2^*, b_1^*, b_2^*, c_1^*\}$ that two unstar monomers can use to form bonds. First, if the partition includes a single 2-starred site, i.e. $\{a_2^*\}$ or $\{b_2^*\}$, there does not exist an unstar monomer, with image U1, that can form bonds with the rest of the starred sites. If the partition includes $\{b_1^*\}$, the polymer necessarily includes the unstar monomer $\{a_2, b_2, c_1\}$ which implies $A = C$ or $B = C$, both of which are forbidden. Similarly, if the partition

includes $\{c_1^*\}$, the polymer includes the unstar monomer $\{a_2, b_1, b_2\}$, which would imply the existence of a reaction of the form $_ + A \longrightarrow B$ which is forbidden. Hence, the partition cannot be made up of subsets of one-element and three-elements of the starred binding sites of $\{a_2^*, b_1^*, b_2^*, c_1^*\}$.

As the largest unstar monomer that is mapped to U1 under f has length 3, the only other case we need to consider is when the partition is composed of two two-element subsets of the starred binding sites. There are 3 such partitions of a set with 4 elements. The partition $\{\{a_2^*, b_2^*\}, \{c_1^*, b_1^*\}\}$ is eliminated as there is no unstar monomer that is mapped to U1 and includes two 1-starred sites. The partition $\{\{a_2^*, c_1^*\}, \{b_1^*, b_2^*\}\}$ is also eliminated as $\{a_2^*, c_1^*\}$ implies the existence of a reaction $A + C \longrightarrow _$, if the unstar monomer is $\{a_2, c_1\}$ or, $_ + A \longrightarrow C$, if the unstar monomer is $\{a_2, c_1, c_2\}$, which are both forbidden. The only other case is $\{\{a_2^*, b_1^*\}, \{b_2^*, c_1^*\}\}$. If one of the unstar monomers is of the form $\{a_2, b_1, b_2\}$, there exists a reaction of the form $_ + A \longrightarrow B$ which is forbidden. Hence, one of the unstar monomers that forms bonds with $\{a_2^*, b_1^*\}$ is necessarily $\{a_2, b_1\}$. For the other unstar monomer, which forms bonds with $\{b_2^*, c_1^*\}$, if it is $\{b_2, c_1\}$, there exists a reaction $B + C \longrightarrow _$, which is forbidden. Hence, the second unstar monomer is necessarily $\{b_2, c_1, c_2\}$. ◀

Proof of Lemma 11. We will proceed similarly to Lemma 10. The self-saturation implies that in the polymer with image P_5 , U3 monomer necessarily includes binding sites a_1 , a_2 , b_1 and b_2 . If the fifth unstarred site is x_1 , which is not c_1 , there is a reaction of the form $A + B \longrightarrow X$, which is forbidden.

For the polymer with image P_8 , the partition of the sites of $\{a_1^*, a_2^*, b_1^*, b_2^*\}$ cannot include a single 2-starred site, i.e. $\{a_2^*\}$ or $\{b_2^*\}$ as there does not exist an unstar monomer, with image U1, that can form bonds with the rest of the starred sites. If the partition includes a single 1-starred site $\{a_1^*\}$ (or $\{b_1^*\}$), the other element of the partition implies that the polymer includes the unstar monomer $\{a_2, b_1, b_2\}$ (or $\{b_2, a_1, a_2\}$), which further implies the existence of a reaction $_ + A \longrightarrow B$ (or $_ + B \longrightarrow A$) which is forbidden. Hence, the partition cannot be composed of one-element and three-element subsets of the starred sites and as in Lemma 10, it is enough to consider partitions into two two-element subsets of the starred sites. The partition $\{\{a_1^*, b_1^*\}, \{a_2^*, b_2^*\}\}$ is eliminated as there is no unstar monomer that is mapped to U1 and includes two 1-starred sites. The partition $\{\{a_1^*, b_2^*\}, \{a_2^*, b_1^*\}\}$ is also eliminated as an unstar monomer including unstarred sites a_1 and b_2 implies the existence of a reaction $B + A \longrightarrow _$, if the monomer is $\{b_2, a_1\}$, or $_ + B \longrightarrow A$, if the monomer is $\{b_2, a_1, a_2\}$, which are forbidden. Hence, only the partition $\{\{a_1^*, a_2^*\}, \{b_1^*, b_2^*\}\}$ is left and the result follows. ◀

Universal Spatial Distribution Approximation in Equilibrium Lattice Models with Local Pairwise Interactions

Cameron Chalk 

California Institute of Technology, Pasadena, CA, USA

Erik Winfree 

California Institute of Technology, Pasadena, CA, USA

Abstract

Universality is one of the defining hallmarks of a useful neural network model: it means that the model is expressive enough, in principle, to learn arbitrary target behavior rather than being limited to a narrow class. Inspired by rigorous formal analogies between stochastic neural networks and lattice models in which molecules occupy discrete sites, contribute species-dependent interaction energies with neighbors, and rearrange according to the Boltzmann distribution, we ask whether interactions can be programmed to yield arbitrary spatial distributions. At the level of fine-grained spatial arrangements of species, universality fails: arbitrary spatial microstate distributions cannot be represented, as shown by a counting argument and explicit construction of unrepresentable distributions. Yet, for a natural observable – spatial patterns of molecular labels – universality can be achieved. We show that a grand-canonical lattice model of anisotropic molecules is universal for arbitrary label distributions, and a similar model with isotropic molecules is also universal given a simple symmetry-breaking boundary condition. Our results suggest that equilibrium systems of multivalent molecules undergoing stochastic rearrangement governed by local pairwise interactions can act similarly to probabilistic neural networks: with sufficient microstate detail in hidden dimensions, they can be universal in a meaningful macrostate observable.

2012 ACM Subject Classification Hardware → Neural systems; Applied computing → Physics; Applied computing → Systems biology

Keywords and phrases universality, molecular programming, neural networks, self-assembly

Digital Object Identifier 10.4230/LIPIcs.DNA.32.11

Funding *Cameron Chalk*: NSF CCF/FET 2212546

Erik Winfree: NSF CCF/FET 2212546

Acknowledgements The authors would like to thank David Doty and Damien Woods for tangential yet inspirational discussions.

1 Introduction

Life is not a homogeneous solution of chemicals. Living cells are cities, with roads [37], boundaries [38], factories [31], warehouses [3], scaffolds [11], and gradients [15], all features that change throughout the lifetime of the cell. Control over *where* molecules are helps transform ordinary chemistry into spatially-organized molecular systems capable of replication, evolution, adaptability, computation, and homeostasis. Concretely, control of molecular spatial distribution allows the cell to regulate reaction rates by colocalizing reactants [25], compartmentalize strictly regulated biomolecule populations into membraned or membraneless organelles [4], direct transport of cargo along tracks [21], change its geometry via the cytoskeleton [8], and break symmetries for morphogenetic and developmental purposes [23]. It follows that constructing prescribed molecular spatial distributions is essential for building synthetic systems with lifelike behavior.



© Cameron Chalk and Erik Winfree;

licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 11; pp. 11:1–11:24

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Engineering accomplishments suggest that the combination of programmable attractive forces and random Brownian motion enables molecularly precise assembly. Thermodynamic self-assembly methods such as DNA origami [33] program molecular binding interactions towards a free energy minimum with (ideally) a high yield of the single target structure. Multivalent DNA nanostars with programmable interactions condense into aggregates with short-timescale reconfiguration [6]. However, spatial arrangement in the cell undergoes dynamic reorganization in response to, e.g., environmental changes [30]. Although engineered self-assembly can be dynamically responsive (see, e.g., reconfigurable DNA origami [40] and dynamic nanostar condensates [2]), approaches are currently specialized for particular tasks. What general programming framework might allow the design of molecules capable of complex, adaptable, and reconfigurable spatial arrangements – putting the right pieces in the right places under the right contexts?

Prior work [28, 29, 7] has situated programming of molecular interactions (chemical reactions, binding, diffusion) in an equilibrium setting as analogous to the training and usage of Boltzmann machines [1], a stochastic, statistical mechanics-inspired neural network model that stands as a precursor to modern artificial neural network approaches. When formalized in terms of local pairwise interactions of molecules in a spatial lattice [7], the model and learning framework was called *Boltzmann liquids*, and is summarized in Figure 1. A key element learned from classical Boltzmann machines is that stochastically generated high-dimensional probability distributions can be probed simply by clamping known variables, which results in probabilistic conditioning of the remaining variables. For example, a Boltzmann machine that is trained on the MNIST handwritten digit dataset can both classify input digits and generate digits given a class, and both of these tasks are achieved by conditioning the distribution. Figure 1(a) shows conditioning of a toy probability distribution to give intuition about how conditioning affects a probability distribution. In the Boltzmann liquids framework, a system of pairwise interacting molecules, typically at low molecular counts (e.g., tens to thousands of molecules as is the case in some biological condensates [18]), at equilibrium takes on a probability distribution of fine-grained molecular arrangements, as shown in Figure 1(b), left. If an environmental stimulus is introduced to the system, like a surface with a fixed arrangement (Figure 1(b), middle), or the presence of a polymer with a certain sequence of domains (Figure 1(b), right), the system assumes a conditional distribution. In other words, the molecules can store information via their interaction energies, and thus their distribution, that can be “accessed” or “queried” by stimuli that condition the distribution. Additionally, work on Boltzmann machines inspired a contrastive learning rule for Boltzmann liquids, as shown in Figure 1(c), which finds parameters for the system to match a target distribution. This framework enabled the design and simulation of systems of pairwise interacting molecules which performed Hopfield network-like associative recall and classification of MNIST handwritten digit patterns expressed as volume fractions of the constituent species types. In short, programming a stochastic system to take on a complex, high-dimensional distribution is an effective method of programming, which is a perspective supported by the success of stochastic neural network models like the Boltzmann machine.

In the study of neural networks, machine learning, and statistical modeling, the expressivity of the model used is a key question; single-layer perceptrons, for example, can only classify linearly-separable data [32], and thus multi-layer perceptrons were suggested for more useful classification. The multi-layer perceptron is known to be *universal* [9]. Universality means that a model can effectively represent or approximate any distribution or input-output relation, usually with some number of caveats. A proof of universality of a model provides assurance that the model does not have any fundamental limits inhibiting a classification or

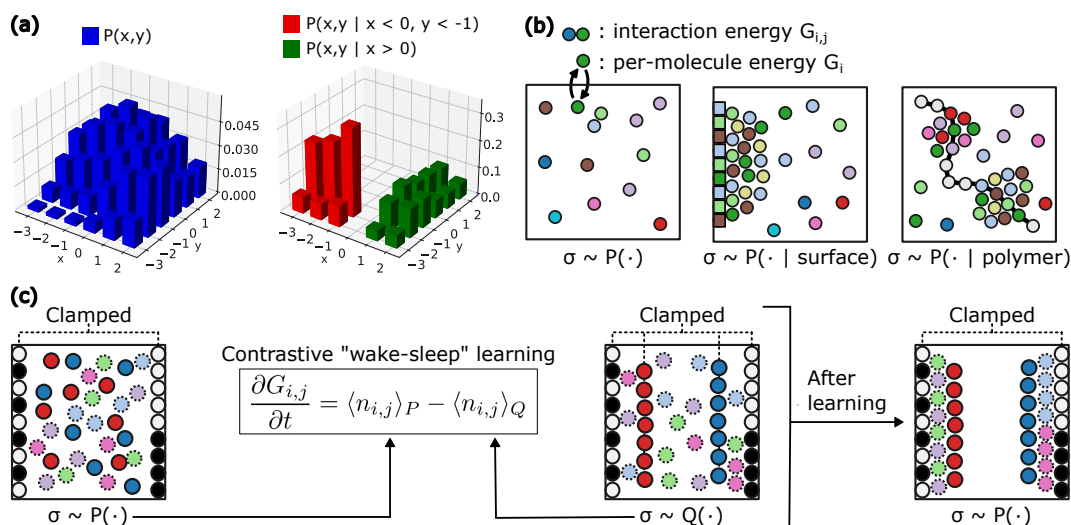
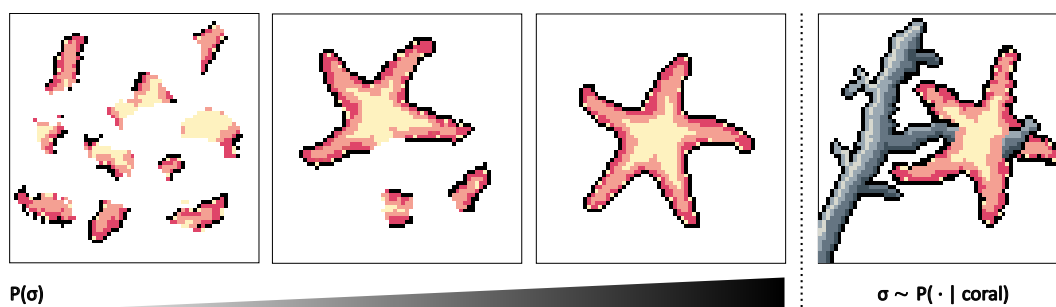


Figure 1 A cartoon summary of Boltzmann liquids. (a) On the left, a two-dimensional probability mass function (PMF). On the right, two different conditional distributions generated from the same PMF. (b) A cartoon representation of an equilibrium pairwise interaction model. Near-enough molecules of species i and j contribute interaction energy $G_{i,j}$, and exchange with a reservoir of molecules occurs dependent on a per-molecule energy (chemical potential) G_i , where more negative $G_{i,j}$ and G_i are more favorable. On the left, a sample configuration σ drawn from the unconditional Boltzmann distribution P , indicating that in P the molecules are dilute with high probability. In the middle, a surface on the left wall is fixed in space, inducing a conditional distribution $P(\cdot \mid \text{surface})$ in which condensation occurs. On the right, the presence of a polymer in the system induces a conditional distribution $P(\cdot \mid \text{polymer})$ in which condensation of different compositions occurs depending on the polymer's sequence. (c) A learning algorithm finds the parameter gradient that makes a distribution more like a target. In this cartoon, there is no molecular exchange with the reservoir, i.e., there is a fixed count of each species. In this example, *clamped* means the molecules are fixed in space; more general notions of clamping to an observable have been studied [7]. On the left, the colored molecules are left unclamped. In the middle, the distribution Q to be learned is sampled – Q is a distribution defined by the current parameters and clamping of a target marginal distribution Q' that specifies where the red and blue molecules should be but does not specify where the “hidden” molecules shown by dotted circles should be. Average statistics $\langle n_{i,j} \rangle$ are collected from each distribution, where $n_{i,j}$ counts the number of $i - j$ interactions. The learning algorithm yields parameters (if possible) which make the system behave more like Q as shown on the right, where the hidden species learn to help organize the red and blue species according to Q' .

generative task, and thus leaves only the remaining questions of required model size, time spent learning (i.e., how much of the data the model sees), and correctness of its learning algorithm. Well-mixed stochastic chemical reaction networks have been shown to be universal for distributions of counts [5], establishing a baseline for viewing molecular systems in terms of probabilistic machine learning theory.

If equilibrium systems are indeed universal for spatial distributions, a consequence is that molecular systems may not require energy to maintain complex spatial patterning. For example, the diversity of acceptable spatial arrangements of an engineered organism could be encoded entirely in its equilibrium distribution, and its response to environmental stimuli could be achieved by conditioning of that distribution (see Figure 2 for an imaginative example). Another consequence of organization as an equilibrium phenomenon is self-healing: the system naturally recovers from perturbations by relaxing to equilibrium. A further consequence is that these aspects of molecular behavior can be viewed as a form



■ **Figure 2** A cartoon of potential outcomes of encoding complex spatial patterning into equilibrium distributions. The Boltzmann distribution encodes the starfish’s structure as shown in the first three panels, with probability increasing from left to right. Connected, five-armed configurations of its constituent molecules are favored (right). A natural consequence is self-healing, where equilibrium relaxation alone is sufficient to restore structure after perturbations like damage to its arms (middle), or even complete disassembly (left). The presence of and interactions with a branch of coral condition the starfish’s Boltzmann distribution, resulting in a high probability of the starfish everting its stomach to feed on the coral.

of “thought” insofar as their governing mathematics are akin to those of neural networks. Of course, energy consumption could and must still be utilized to condition the spatial distribution into particular states, to drive the system through its various configurations, and to metabolically synthesize the relevant molecules in the first place. That said, maintenance of complex structure at equilibrium opposes the idea that equilibrium generically yields disorder. Surprisingly, the disorder intuition is most often taught in terms of lattice gas models close to those studied here. In our case, the key differences which introduce complex spatial order at equilibrium are heterogeneous interaction energies between molecular species, and large species diversity: the inclusion of many species, where an increase in the number of species increases spatial distribution complexity.

In this paper, we introduce several standard lattice models of pairwise molecular interactions at equilibrium, including the model initially studied in the Boltzmann liquids [7] work, and study their universality or lack thereof for representing spatial distributions. Our models are standard generalizations of basic lattice gas models, which include many species types with heterogeneous pairwise interaction energies. Stochastic modeling captures the noise of Brownian motion, and equilibrium assumptions imply the probability of each state of the system is given analytically by the Boltzmann distribution. The question then is whether interaction energies can be tuned so that a target distribution over spatial molecular arrangements can be achieved. Our work first presents two types of impossibility results, arguing that pairwise interactions alone are not sufficient to encode arbitrary distributions over fine-grained molecular arrangements. We then present a notion of “labeling” molecules – one can think of each of many species types being labeled with one of two fluorescent molecules – where we show that any distribution over spatial label configurations can be approximated by programming interaction energies. This universality result for label distributions stands as initial evidence that Boltzmann liquids – molecules rearranging according to pairwise interactions at equilibrium – are not limited to a simple set of distributions, and as such may be programmed effectively for complex computation. As a caveat that will be discussed later, such distributions may use strong energies and thus not behave as a “liquid” (i.e., may not sample many configurations) on reasonable timescales.

2 Lattice Models of Pairwise Interactions

2.1 Model assumptions

Our model assumptions are as follows. First, we consider space to consist of discrete positions. Second, we consider the molecules to be *monodisperse*, meaning all molecules are the same size, the solvent species is treated as any other species, and each molecule occupies one position. It is worth mentioning that the monodispersity assumption can be relaxed by an extension to the model not discussed in this work, and further that in the limit of finer lattice spacing relative to molecule size, the model approximates continuous space. There are two bifurcations to the model assumptions that we consider: the first is isotropy vs. anisotropy. In the isotropic case, we view molecules as rotationally symmetric, meaning species i and j in neighboring positions will always contribute a symmetric interaction energy $G_{i,j}$. We will also consider the anisotropic case, which will explicitly model molecules as rotatable cubes in a cubic lattice, in which case species i and j in neighboring positions contribute an interaction energy that depends on their orientations. The second bifurcation is whether the system is in thermal equilibrium with a heat bath but is otherwise isolated (the canonical ensemble in statistical thermodynamics) vs. in thermal *and chemical* equilibrium with an infinite reservoir wherein molecules exchange with the reservoir (the grand canonical ensemble in statistical thermodynamics). Thus, the reader can keep in mind four models: canonical isotropic, canonical anisotropic, grand-canonical isotropic, and grand-canonical anisotropic. Lastly, we assume that the energy of the system is captured solely by interaction energies $G_{i,j}$ and chemical potentials G_i , and because the system is in equilibrium, the probability mass is given by the Boltzmann distribution, to be formalized shortly.

2.2 Model formalization

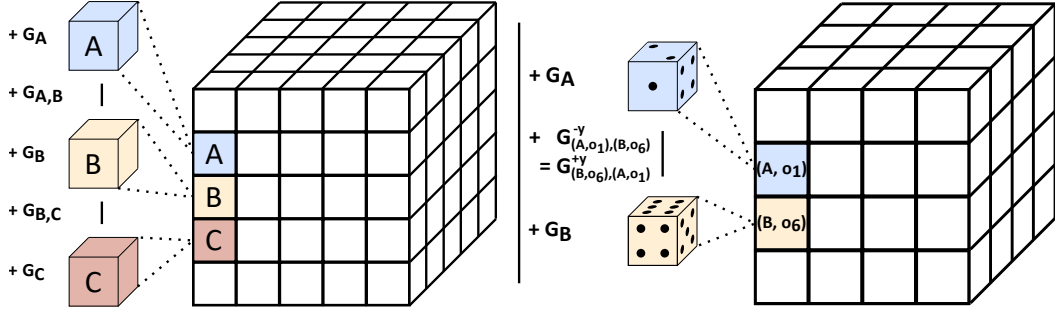
We represent space as a set of positions V . If positions p_1 and p_2 are such that species in those positions induce a pairwise interaction, we say $p_1 \sim p_2$. Equivalently, the space is represented by an undirected graph $G = (V, E)$. Commonly, V is taken as a lattice $V = \mathbb{N}^H \times \mathbb{N}^W \times \mathbb{N}^L$, and edges are restricted to be local, e.g., the *von Neumann* neighbors. Formally, $p_1, p_2 \in \mathbb{Z}^3$ are von Neumann neighbors if $\|p_1 - p_2\|_1 = 1$.

Given a set of chemical *species* M , we consider spatial *configurations* of species in positions, such that a configuration is formally $\sigma : V \rightarrow M$. The solvent is modeled explicitly as one of the species in M . The assumption that the system is in equilibrium implies that the probability mass over configurations is given by the Boltzmann distribution $P(\sigma) \propto \exp(-G(\sigma)/kT)$ for an energy function $G(\sigma)$, where T is the temperature in Kelvin and k is Boltzmann's constant. This energy function is expressed differently in the isotropic vs. anisotropic cases as follows. Figure 3 provides visual examples that may aid the reader in understanding the models and distinguishing the isotropic vs. anisotropic cases.

2.2.1 Isotropic model

Here we use a standard generalization of a lattice gas Ising model, equivalently a Potts model with heterogeneous weights. This model has been used in prior work in the context of liquid-liquid phase separation of more than two species (multicomponent liquids) [12], as well as in the Boltzmann liquids work [7]. Here, the energy of a configuration is given by:

$$G(\sigma) = \frac{1}{2} \sum_{p_1 \in V} \sum_{\substack{p_2 \in V \\ \text{s.t. } p_1 \sim p_2}} G_{\sigma(p_1), \sigma(p_2)} + \sum_{p \in V} G_{\sigma(p)},$$



■ **Figure 3** Two example partial configurations. The configurations are partial in the sense that our models do not allow empty positions: solvents, or equivalently “empty” species are explicit species of the set of species M , and each position must be assigned a species in M . Positions are left empty here for visual clarity. On the left, a configuration σ of the isotropic model and the contributions to the configuration energy $G(\sigma)$ provided by the visualized species and their interactions. On the right, a configuration σ of the anisotropic model. Species can be thought of as rotatable cubes (i.e., as dice), with six distinct faces. A position is assigned a species type (here, A , B) and an orientation (here, o_1 and o_6), where orientations correspond to the 24 discrete rotations of the cube, $SO(3, \mathbb{Z})$. Here, $+y = (0, 1, 0)$ and $-y = (0, -1, 0)$ denote the relative direction in the lattice of the two species which are providing interaction energy. In the anisotropic models, constraints on interaction energies such as $G_{(A,o_1),(B,o_6)}^{-y} = G_{(B,o_6),(A,o_1)}^{+y}$ give the model equivalence to a model which defines $G_{A5,B6}$ as the energy provided when species A and B have their die faces 5 and 6 abutting, as pictured, noting that die face 5 is occluded but is opposite face 2.

where $G_{i,j} = G_{j,i}$ is the energy contribution of a molecule of species $i \in M$ interacting with a molecule of species $j \in M$, and where G_i is the internal energy (equivalently, chemical potential) of species $i \in M$. The convention is more negative $G_{i,j}$ and G_i are more favorable.

In our analysis, it will be useful to write this energy in an alternate form which clarifies that key statistics ($n_{i,j}$ and n_i , defined shortly) of a configuration are sufficient to determine its energy. The Kronecker delta function will be useful, which returns one if its subscripts are equal, otherwise zero: $\delta_{i,j} = 1$ if $i = j$, else $\delta_{i,j} = 0$. A configuration σ induces a value $n_{i,j}$ for all $i, j \in M$, which is the number of neighbor interactions between molecules i and j . We define:

$$\hat{n}_{i,j} = \sum_{p_1 \in V} \sum_{\substack{p_2 \in V \\ \text{s.t. } p_1 \sim p_2}} \delta_{\sigma(p_1),i} \delta_{\sigma(p_2),j} .$$

As written, $\hat{n}_{i,j}$ double counts the case $i = j$, which we correct as:

$$n_{i,j} = \begin{cases} \frac{1}{2} \hat{n}_{i,j} & i = j \\ \hat{n}_{i,j} & i \neq j . \end{cases}$$

Additionally, a configuration σ induces a count of each species in the lattice,

$$n_i = \sum_{p \in V} \delta_{\sigma(p),i} .$$

Finally, we can see that lattice configuration σ has the corresponding energy:

$$G(\sigma) = \sum_{i \leq j \in M} n_{i,j} G_{i,j} + \sum_{i \in M} n_i G_i \quad (1)$$

where each independent energy parameter appears exactly once. This formulation makes clear the idea that any two configurations with equal $n_{i,j}$ and n_i must have equal energy and thus equal probability.

If we assume the system's dynamics obey detailed balance with respect to the energy we have defined, the Boltzmann distribution provides an analytical formula for the probability of a given configuration:

$$P(\sigma) = \frac{1}{Z} e^{-G(\sigma)/kT}, \quad \text{where } Z \text{ is the partition function,} \quad Z = \sum_{\sigma \in \Omega} e^{-G(\sigma)/kT}.$$

Here, Ω is the set of valid configurations, which reflect assumptions about physical constraints. If the system is considered to be in thermal equilibrium with a constant-temperature heat bath (*canonical*), then species do not diffuse from or into the heat bath, so the molecular counts are fixed. If we let $\vec{c} \in \mathbb{N}^M$ be the vector representing the count of each species, the *canonical configuration space* is the set $\Omega = \{\sigma : V \rightarrow M \mid \forall i \in M, n_i(\sigma) = \vec{c}(i)\}$. If instead the system is in thermal and chemical equilibrium with a reservoir (*grand canonical*), species diffuse to and from the reservoir, so the counts of each species type can change, and thus the set of configurations is $\Omega = \{\sigma \mid \sigma : V \rightarrow M\}$.

2.2.2 Anisotropic model

For simplicity, in the anisotropic model we restrict the set of positions to the cubic lattice $V = \mathbb{N}^H \times \mathbb{N}^W \times \mathbb{N}^L$ with von Neumann neighbors. We then effectively consider molecules to be rotatable cubes with asymmetric internal structure (see Figure 3, right). Formally, each species is augmented with an *orientation*: the set of species $M = M_{\text{type}} \times M_{\text{orientation}}$ consists of elements formed by the Cartesian product of a set of *species types* M_{type} and a set of *orientations* $M_{\text{orientation}}$. Since we consider molecules as rotatable cubes, we take $M_{\text{orientation}} = SO(3, \mathbb{Z})$, where $SO(3, \mathbb{Z})$ is the group of orientation-preserving cube rotations. With $i = (a, o_1), j = (b, o_2) \in M$, the interaction energies must be specified with a direction $\hat{u} \in \Delta$, where Δ is a set of *displacements* that must be symmetric in the sense that $-\hat{u} \in \Delta$ if and only if $\hat{u} \in \Delta$. For our von Neumann neighborhood, $\Delta = \{\hat{u} \in \mathbb{Z}^3 \mid \|\hat{u}\|_1 = 1\}$. Then $G_{i,j}^{\hat{u}}$ describes the interaction energy between species a and b when they are in orientations o_1 and o_2 , respectively, and are displaced by $\hat{u}$ in the lattice. Note that the rotation group $SO(3, \mathbb{Z})$ also acts on displacements; then we require that the interaction energies are invariant to rotations, formally:

$$\forall R \in SO(3, \mathbb{Z}), \quad G_{(a, Ro_1), (b, Ro_2)}^{R\hat{u}} = G_{(a, o_1), (b, o_2)}^{\hat{u}}.$$

In addition, we require interaction symmetry as in the isotropic case: $G_{(a, o_1), (b, o_2)}^{\hat{u}} = G_{(b, o_2), (a, o_1)}^{-\hat{u}}$. Finally, we require that the internal energy is independent of orientation, formally $G_{(a, o)} = G_a$ for all $o \in SO(3, \mathbb{Z})$. In the canonical anisotropic model, the counts of each species type in M_{type} are fixed, but orientations can change.

The energy of a configuration is

$$G(\sigma) = \frac{1}{2} \sum_{p \in V} \sum_{\substack{\hat{u} \in \Delta \\ p + \hat{u} \in V}} G_{\sigma(p), \sigma(p + \hat{u})}^{\hat{u}} + \sum_{p \in V} G_{\sigma(p)}.$$

As in the isotropic case, it is useful to define sufficient statistics for the energy. For each $i, j \in M$ and $\hat{u} \in \Delta$, we define:

$$n_{i,j}^{\hat{u}}(\sigma) = \sum_{\substack{p \in V \\ p + \hat{u} \in V}} \delta_{\sigma(p), i} \delta_{\sigma(p + \hat{u}), j},$$

which counts the number of ordered neighboring pairs of type $i \rightarrow j$ in the displacement direction $\hat{u}$. The number of molecules of species type i in the lattice is defined as in the isotropic case: $n_i(\sigma) = \sum_{o \in SO(3, \mathbb{Z})} \sum_{p \in V} \delta_{\sigma(p), (i, o)}$. Noting that $n_{i,j}^{\hat{u}} = n_{j,i}^{-\hat{u}}$, we can write the energy as:

$$G(\sigma) = \frac{1}{2} \sum_{\hat{u} \in \Delta} \sum_{i,j \in M} n_{i,j}^{\hat{u}} G_{i,j}^{\hat{u}} + \sum_{i \in M_{\text{type}}} n_i G_i.$$

2.2.3 A unified expression for energy

To discuss all models simultaneously, it is helpful to write the energy of all models in a standard form. For each model, we define the *parameters* θ and the *sufficient statistics* t . The parameters θ are a vector and are defined as below for each model. Here, $[x_a] \forall a \in A$ denotes the vector obtained by fixing an ordering of the set A and listing one entry x_a for each $a \in A$. In the isotropic canonical model,

$$\theta = [G_{i,j}/kT] \quad \forall i, j \in M \text{ with } i \leq j,$$

in the isotropic grand-canonical model,

$$\theta = [G_{i,j}/kT, G_m/kT] \quad \forall i, j \in M, \forall m \in M \text{ with } i \leq j,$$

in the anisotropic canonical model,

$$\theta = [G_{i,j}^{\hat{u}}/kT] \quad \forall i, j \in M \text{ with } i \leq j, \forall \hat{u} \in \Delta,$$

and in the anisotropic grand-canonical model,

$$\theta = [G_{i,j}^{\hat{u}}/kT, G_m/kT] \quad \forall i, j \in M \text{ with } i \leq j, \forall m \in M_{\text{type}}, \forall \hat{u} \in \Delta.$$

The sufficient statistic vector t is a function $t : \Omega \rightarrow \mathbb{Z}_{\geq 0}^{|\theta|}$ that takes as input a configuration and outputs its statistics $n_{i,j}$, n_i , and/or $n_{i,j}^{\hat{u}}$. We define t so that it aligns with θ (e.g., if $G_{i,j}$ is element k of θ , then $n_{i,j}$ is element k of $t(\sigma)$) and thus write for any model:

$$G(\sigma)/kT = \theta \cdot t(\sigma), \quad P(\sigma) = \frac{1}{Z} \exp(-\theta \cdot t(\sigma)).$$

This formulation makes clear the idea that, for fixed parameters θ , the Boltzmann distribution P depends only on the sufficient statistics t , hence their name; this dependence narrows possible target distributions, as discussed in the next section.

3 Impossibility of Microstate Universality

Here, we formalize and prove the impossibility of the models to represent all microstate distributions, even those that satisfy intuitive symmetries. The most straightforward formulation of universality for distributions is given in terms of the relative entropy, or Kullback-Leibler divergence between target distribution Q and approximate distribution P , where

$$D_{\text{KL}}(Q||P) = \sum_{\sigma} Q(\sigma) \log \frac{Q(\sigma)}{P(\sigma)}.$$

Then:

► **Definition 1.** A model is universal for microstate distributions if for all $\varepsilon > 0$, V , M , and Q over Ω , there exists θ such that $D_{KL}(Q||P) < \varepsilon$.

Of course, this formulation is beyond reach: as written, Q may require different probabilities for configurations that agree on sufficient statistics. For example, in the isotropic models, there is a rotational symmetry in which rotating a configuration yields another configuration with the same sufficient statistics. Similarly, if periodic boundary conditions are used in any model, then there is also a translational symmetry. We thus define the *uniform-within-fibers* constraint on distributions as follows: two configurations with equal sufficient statistics, $t(\sigma_1) = t(\sigma_2)$, must have equal probability. This constraint captures all translational and rotational symmetries of the models, in addition to configurations which share t but are not translations/rotations (see Appendix A.1 for an example). The set of configurations with sufficient statistics equal to t is called a *fiber* over t , formally $F_t = \{\sigma \in \Omega : t(\sigma) = t\}$, so we say a distribution Q over Ω is *fiber uniform* if:

$$\forall t, \forall \sigma_1, \sigma_2 \in F_t, Q(\sigma_1) = Q(\sigma_2).$$

Thus we define:

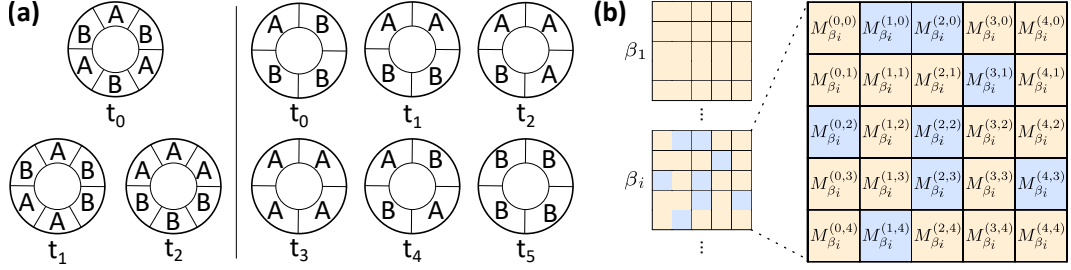
► **Definition 2.** A model is universal for fiber-uniform microstate distributions if for all $\varepsilon > 0$, V , M , and fiber-uniform Q over Ω , there exists θ such that $D_{KL}(Q||P) < \varepsilon$.

First, we provide a simple counting argument that, for nontrivial lattice sizes $|V| = \Theta(|M|^2)$, there are not enough parameters to encode every fiber-uniform target distribution. This counting argument proves that not every fiber-uniform target distribution can be the result of a parameter setting if parameters have bounded precision, and provides intuition that the number of parameters must be allowed to scale independently of the number of target distributions, motivating our problem reformulation in Section 4. Later in this section, we provide concrete, simple fiber-uniform distributions which suffice to prove that approximate microstate universality as defined above is not possible. These counterexamples are constructed by identifying a constraint on distributions that must be satisfied in order for them to be achieved, and thus provide value beyond the counting argument by illustrating what *kind* of microstate distributions are impossible, in addition to removing the assumption of bounded-precision parameters and illustrating impossibility even of approximation.

3.1 Bounded-precision exact microstate universality requires an exponential number of parameters

In the study of *in-silico* probabilistic models, it is standard to assume that parameters have bounded precision (i.e., that each parameter uses a constant number of bits b). If we make the same assumption for our models, with the physical interpretation that interaction energies are not tunable on a true continuum but instead have some underlying discrete characterization, we can apply a standard counting argument to show that the number of parameters required for exact representation of every fiber-uniform distribution should require a number of parameters that scales exponentially in the support of the distribution (the potential configurations).

For any model, define the set of feasible sufficient statistics vectors $\mathcal{T}_\Omega := \{t(\sigma) \mid \sigma \in \Omega\}$. For each probability vector q over $\mathcal{T}_\Omega$, define the following fiber-uniform microstate distribution: $Q_q(\sigma) := \frac{q(t(\sigma))}{|F_{t(\sigma)}|}$. The set of Q_q defined this way is the set of possible fiber-uniform target distributions. In order to argue that there are more targets than our parameter choices allow, it suffices to consider a subset of the total set of targets whose size is easier to count, so we define:



■ **Figure 4** (a) On the left, one representative configuration from each of the fibers t_0, t_1 , and t_2 in a canonical isotropic model with V a ring of six positions as shown. On the right, one representative configuration from each of the six fibers in a grand canonical isotropic model with V a ring of four positions as shown. (b) An example of Theorem 5's construction, in two dimensions for visual clarity. There are two labels, yellow and blue. $\beta_1 \dots \beta_i \dots$ denote the elected label configurations for each Euclidean equivalence class of label configurations. The intended configuration for β_i is shown in the pop-out. Because each species $M_{\beta_i}^{(x,y)}$ is adjacent to its proper neighbors, no energy contribution Λ is made, and the energy of the microstate configuration is given by the energy $-kT \ln Q_K(\beta_i)$ of the leader species $M_{\beta_i}^{(0,0)}$.

$$\mathcal{D}_{\mathcal{T}} = \left\{ q : \mathcal{T}_{\Omega} \rightarrow [0, 1] \mid q(t) = \frac{z_t}{|\mathcal{T}_{\Omega}|}, z_t \in \mathbb{Z}_{\geq 0}, \sum_{t \in \mathcal{T}_{\Omega}} z_t = |\mathcal{T}_{\Omega}| \right\}.$$

Using standard combinatorics (see Appendix A.2), we can count the size of $\mathcal{D}_{\mathcal{T}}$:

$$|\mathcal{D}_{\mathcal{T}}| = \binom{2|\mathcal{T}_{\Omega}| - 1}{|\mathcal{T}_{\Omega}| - 1} \geq 2^{|\mathcal{T}_{\Omega}| - 1}.$$

Now, let $n = |\theta|$, the number of parameters of the model. If each parameter is allowed b bits, the set of all possible parameter choices of the model has size 2^{bn} . For every fiber-uniform distribution to be achieved exactly by some parameter choice would require $2^{bn} \geq 2^{|\mathcal{T}_{\Omega}| - 1}$, therefore $n \geq \frac{|\mathcal{T}_{\Omega}| - 1}{b}$. In Appendix A.3, we prove in each model that with $|V| = \Theta(|M|^2)$ (the lattice size scaling quadratically in the number of species) that $|\mathcal{T}_{\Omega}| \geq 2^{\Omega(n)}$, where Ω on the right-hand side of the inequality refers to asymptotic scaling. Therefore, for fixed b and scaling n , the inequality $n \geq \frac{|\mathcal{T}_{\Omega}| - 1}{b}$ cannot be satisfied.

Next, we provide explicit counterexample distributions that allow proofs against even approximate microstate universality without the assumption of bounded precision.

3.2 Unbounded-precision approximate microstate universality is prohibited by explicit construction of unachievable distributions

Here we show a constraint that distributions must satisfy beyond being fiber uniform, which allows construction of distributions which cannot be approximated, even with unbounded-precision parameters. The *affine log-probabilities* constraint is that the log-probabilities of the Boltzmann distribution are affine ($f(x) = Ax + b$) in form:

$$\log P(\sigma) = -\theta \cdot t(\sigma) - \log Z,$$

since $\log Z$ is constant with respect to σ . Next, we use this constraint to construct microstate distributions in each model variant that are uniform within fibers but are not approximable, inhibiting universality.

Consider in the canonical isotropic model a ring of six positions with left-right edges but no top-bottom edges with three copies of species A and three copies of species B (Figure 4(a), left). There are three *feasible* fibers, meaning values of $t = (n_{A,A}, n_{A,B}, n_{B,B})$ for which $|F_t| > 0$, i.e., sufficient statistics that can arise from valid microstate configurations. These three feasible fibers are illustrated in Figure 4(a), left. Consider the following target distribution Q , described in terms of an assignment of probability to each fiber's microstates:

If $t(\sigma) = t_0 = (0, 6, 0)$, set $Q(\sigma) = 1/32$.

If $t(\sigma) = t_1 = (1, 4, 1)$, set $Q(\sigma) = 1/16$.

If $t(\sigma) = t_2 = (2, 2, 2)$, set $Q(\sigma) = 1/32$.

By this assignment method, this distribution satisfies the uniform-within-fibers constraint. Additionally, the probabilities sum to one: since $|F_{t_0}| = 2$, $|F_{t_1}| = 12$, and $|F_{t_2}| = 6$, we have $2/32 + 12/16 + 6/32 = 1$. But does the distribution satisfy the affine log-probabilities constraint? We show that the answer is no, by showing that it violates the *midpoint relation* of affine functions:

$$f\left(\frac{x_1 + x_2}{2}\right) = \frac{f(x_1) + f(x_2)}{2}.$$

To show this, first note that we can uniquely define the *fiber microstate probability* $u(t) := Q(\sigma)$ such that $\sigma \in F_t$, since Q is uniform within fibers. Note that $t_1 = \frac{t_0 + t_2}{2}$. Towards contradiction, assume Q satisfies the affine log-probabilities constraint, so that it must satisfy the affine midpoint relation:

$$\log u(t_1) = \log u\left(\frac{t_0 + t_2}{2}\right) = \frac{\log u(t_0) + \log u(t_2)}{2},$$

and by exponentiating, we have

$$u(t_1)^2 = u(t_0)u(t_2),$$

but $(1/16)^2 \neq (1/32)^2$, thus a contradiction. Therefore Q , which satisfies the fiber-uniform constraint but does not satisfy the affine log-probabilities constraint, cannot have parameters $\theta = (G_{A,A}, G_{A,B}, G_{B,B})$ that produce it. In Appendix A.4, we provide similar examples in each model variant. Although these counterexamples use lattices with restricted edges for simplicity of presentation, we expect that counterexamples can be constructed on nontrivial lattices in general, e.g., on the typically-considered cubic lattices with von Neumann neighbors.

Next, we utilize the counterexamples to formally argue against universality as per Definition 2:

► **Theorem 3.** *No model variant is universal for fiber-uniform microstate distributions.*

Proof. Towards contradiction, assume at least one model variant is universal for fiber-uniform microstate distributions. Thus, for any fiber-uniform Q , there is an infinite sequence of parameter choices $\theta_1, \theta_2, \dots$ such that the KL divergence between Q and P_{θ_n} gets arbitrarily close to zero as n goes to infinity: $\lim_{n \rightarrow \infty} D_{\text{KL}}(Q||P_{\theta_n}) = 0$. Since P_{θ_n} and Q are discrete distributions, we have $\lim_{n \rightarrow \infty} P_{\theta_n}(\sigma) = Q(\sigma)$, i.e., the probability assigned to σ by P_{θ_n} gets arbitrarily close to the probability assigned by Q (this can be proven using Pinsker's inequality [36]: $\frac{1}{2} \sum_{\sigma} |Q(\sigma) - P_{\theta_n}(\sigma)| \leq \sqrt{\frac{1}{2} D_{\text{KL}}(Q||P_{\theta_n})}$). Let the lattice V , species M , and Q be defined as in the counterexample for the assumed-universal model variant. We write $u_n(t) := P_{\theta_n}(\sigma)$ (and as above, $u(t) := Q(\sigma)$) such that $\sigma \in F_t$. As with $u(t)$,

$u_n(t)$ is well defined, since each P_{θ_n} must be fiber uniform. Since $\lim_{n \rightarrow \infty} P_{\theta_n}(\sigma) = Q(\sigma)$, we have $\lim_{n \rightarrow \infty} u_n(t) = u(t)$. Since P_{θ_n} is the Boltzmann distribution of one of our models, it must satisfy the log-affine constraint, and as above we have $\lim_{n \rightarrow \infty} u_n(t_1)^2 = \lim_{n \rightarrow \infty} u_n(t_0)u_n(t_2)$, so $u(t_1)^2 = u(t_0)u(t_2)$ which, as above, is a contradiction by the counterexample's construction. Therefore no model variant is universal for fiber-uniform microstate distributions. $\blacktriangleleft$

Aside from the parameter scaling issue raised by the counting argument, these counterexamples violating the affine log-probabilities constraint provide an initial investigation into the structure of exactly which microstate distributions can and cannot be represented. In the following section, we provide a macrostate picture in which we achieve universal distribution approximation by circumventing both flavors of impossibility results.

4 A Natural Observable Enables Parameter Scaling and Macrostate Universality

As shown by the counting argument in the previous section, universal distribution approximation requires a way of scaling parameters independently of the domain of the target distribution. Here we describe *macrostates*, which are partitions of the microstates into collections which have the same *observable outcomes*, introducing the possibility for the number of molecular species and thus parameters to be scaled independently of the number of possible observable macrostates.

Formally, we follow [7] in defining macrostates in general. Given a set of configurations Ω , an *observable* is *any* function $a : \Omega \rightarrow A$. We call $\alpha \in A$ an observable outcome. For example, an observable could be $a(\sigma) = n_i(\sigma)$, the count of species i in configuration σ (experimentally, species i could be fluorescently labeled, and thus our observable would be proportional to fluorescence measurement). Given an observable a and an outcome $\alpha \in A$, we define the set of microstates *consistent* with the outcome α as $M_a(\alpha) = \{\sigma \in \Omega \mid a(\sigma) = \alpha\}$. Given parameters θ and thus a Boltzmann distribution P over configurations, the probability of seeing a particular observable outcome α is the marginal probability $P_a(\alpha) = \sum_{\sigma \in M_a(\alpha)} P(\sigma)$, i.e., the sum of corresponding microstates.

Our choice of observable is as follows. We *label* each species with a label from a set of labels K , formally by a labeling function $\mathcal{K} : M \rightarrow K$ that assigns each species $m \in M$ to a label $k \in K$. In the anisotropic case, labels are meant to ignore orientation, so we enforce $\mathcal{K}((a, o)) = \mathcal{K}((a, o'))$ for all $a \in M_{\text{type}}$ and $o, o' \in M_{\text{orientation}}$. This labeling observable captures phenomena such as fluorescently-labeled molecules or molecules with functional domains independent of their interacting domains. A *label configuration* is a function $\beta : V \rightarrow K$ that maps lattice positions to labels. Given a set of species M , a labeling function $\mathcal{K}$, and a configuration σ , we define the label configuration of σ as $\beta_\sigma = \mathcal{K} \circ \sigma$. Given parameters θ , we can write the probability of a label configuration β as:

$$P_{\mathcal{K}}(\beta) = \sum_{\sigma \in \Omega \mid \mathcal{K} \circ \sigma = \beta} P(\sigma),$$

the sum of microstate configurations that map to β by the labeling function $\mathcal{K}$.

In this section, we focus on cubic lattices $V = \{0, \dots, L-1\}^3$ with periodic von Neumann interaction neighborhoods. The purpose of equal edge lengths of the lattice is to simplify the explanation of the requirement that target label distributions must be rotationally invariant, as will be described. The purpose of considering periodicity is to highlight the construction's

requirement that target label distributions are translation invariant, as will be described; we expect the result holds with non-periodic boundary conditions, as well as on other graphs (positions/neighbors).

We define a distribution Q over label configurations as *translationally invariant* as follows. Let a lattice translation $\tau_d : V \rightarrow V$ be defined as $\tau_d(p) = p + d \pmod L$ for $d \in \mathbb{Z}^3$. This induces a label translation $\tau_d\beta : V \rightarrow K$ defined as $\tau_d\beta(p) = \beta(\tau_d^{-1}(p))$. A distribution Q over label configurations is translationally invariant if $Q(\beta) = Q(\tau_d\beta)$ for all $d \in \mathbb{Z}^3$. Any distribution induced by a setting of parameters for our models on a periodic lattice must satisfy translational invariance because translations do not change energy nor probability: $G(\sigma) = G(\tau_d\sigma)$ and thus $P(\sigma) = P(\tau_d\sigma)$, where $\tau_d\sigma(p) = \sigma(\tau_d^{-1}(p))$.

Further, we define a distribution Q over label configurations as *rotationally invariant* as follows. Consider a lattice rotation $R \in SO(3, \mathbb{Z})$. We define the rotation of a label configuration as $R\beta(p) = \beta(R^{-1}p \pmod L)$. Then a distribution Q over label configurations is rotationally invariant if for all $R \in SO(3, \mathbb{Z})$, $Q(\beta) = Q(R\beta)$. This constraint must be satisfied by distributions induced by our model because rotations do not change energy nor probability: in the isotropic model, $G(\sigma) = G(R\sigma)$ and thus $P(\sigma) = P(R\sigma)$, where $R\sigma(p) = \sigma(R^{-1}p \pmod L)$. In the anisotropic model, a lattice rotation also rotates the internal orientation of each molecule: given an anisotropic species (i, o) , define $\hat{R}(i, o) = (i, Ro)$ and allow it to act elementwise on configurations σ , so that $R\sigma(p) = \hat{R}(\sigma(R^{-1}p \pmod L))$, and equivalently if $\sigma(R^{-1}p \pmod L) = (i, o)$ then $R\sigma(p) = (i, Ro)$. Since we constrain the anisotropic interaction energies to be rotationally invariant, rotations do not change energy: $G(\sigma) = G(R\sigma)$ and $P(\sigma) = P(R\sigma)$.

We call a label distribution *Euclidean invariant* if it is both translationally and rotationally invariant. Thus we define universality for label distributions as follows:

► **Definition 4.** *A model is universal for Euclidean-invariant label distributions if for all $\varepsilon > 0$, $V = \{0, \dots, L-1\}^3$, von Neumann neighbors E , a set of labels K , and Euclidean-invariant label distributions Q_K , there exists a set of species M , parameters θ , and labeling function $\mathcal{K}$ such that $D_{KL}(Q_K || P_{\mathcal{K}}) < \varepsilon$, where $P_{\mathcal{K}}$ is the label distribution induced by M , θ , V , E , and $\mathcal{K}$.*

Note that by this definition, the set of species M can scale independently of V and K , so that the number of parameters $|\theta|$ can scale independently of the domain of the target distribution Q_K , circumventing the impossibility via counting argument of Section 3.1. Next, we show by construction that both the anisotropic and isotropic grand-canonical models are universal for label distributions. Our particular definitions prohibit canonical models from universality: since a target distribution fixes the lattice size $|V|$ and canonical configuration spaces consider a set of at most $|V|$ distinct species, scaling the number of labels K increases the number of label configurations beyond the fixed number of parameters. In the discussion, we illustrate an alternative observable that may allow universality in the canonical models.

4.1 Universality in the anisotropic grand-canonical model

Here we provide a construction that shows that the anisotropic grand-canonical model is universal for Euclidean-invariant label distributions. The construction is a “brute-force”, “hard-coded” construction, meaning that each label configuration’s probability is effectively encoded into a single parameter. As a consequence, the number of parameters (and thus unique species) scales with the number of label configurations with nonzero probability in the target distribution. This may be considered analogous to Boltzmann machine universality proofs where the probability of each visible node configuration is effectively encoded in a

single hidden neuron [16]. The role of “lookup-table” constructions such as this one and the potential for next questions related to structured distributions that do not require so many parameters is elaborated on in the Discussion.

► **Theorem 5.** *The anisotropic grand-canonical model is universal for Euclidean-invariant label distributions.*

Proof. Figure 4(b) illustrates the construction in two dimensions, in case it is helpful to the reader. Given a target label distribution Q_K over the set of label configurations $\{\beta_1, \dots\}$, our goal is to construct a set of species M and a labeling function $\mathcal{K}$, and to set the parameters θ such that P_K approximates Q_K . We define the *Euclidean equivalence class* on label configurations as follows. We say $\beta_i \hat{=} \beta_j$ if there exist a lattice rotation $R \in SO(3, \mathbb{Z})$ and lattice translation τ_d for $d \in \mathbb{Z}^3$ such that $\tau_d(R\beta_i) = \beta_j$. Then the Euclidean equivalence class of β_i , denoted $\hat{\beta}_i$, is the equivalence class given the $\hat{=}$ relation, formally $\hat{\beta}_i = \{\beta_j \mid \beta_i \hat{=} \beta_j\}$. This partitions the set of label configurations into a set of equivalence classes. For each equivalence class, we choose one *elected configuration* and denote it $\hat{\beta}_i^*$.

We now construct the set of species M . We first define the *species for a label configuration* β_i . For each elected configuration $\hat{\beta}_i^*$ such that $Q_K(\hat{\beta}_i^*) > 0$, for each position in the lattice $(x, y, z) \in V$, we include a species type $M_{\beta_i}^{(x, y, z)}$. We define the *intended position* $\text{pos}(M_{\beta_i}^{(x, y, z)}) = (x, y, z)$ and the *intended label configuration* $\text{cfg}(M_{\beta_i}^{(x, y, z)}) = \hat{\beta}_i^*$. The labeling function $\mathcal{K}$ is defined so that a species type has a label matching its intended position in its intended label configuration, formally $\mathcal{K}(M_{\beta_i}^{(x, y, z)}) = \hat{\beta}_i^*((x, y, z))$.

We set the interaction energies $G_{i,j}^{\hat{u}}$ as follows. First, we pick a large positive energy Λ , to be quantified later with respect to the target ε . We will consider species of some *default orientation* $R_0 \in SO(3, \mathbb{Z})$. Our interactions will be local von Neumann interactions, so that $G_{i,j}^{\hat{u}} = 0$ for all $\hat{u} \notin \{\hat{u} \mid \|\hat{u}\|_1 = 1\}$. For all $i = (M_{\beta_a}^{p_2}, R_0)$ and $j = (M_{\beta_b}^{p_1}, R_0)$, we set $G_{i,j}^{\hat{u}} = \Lambda$, unless the species types correspond to the same label configuration, formally $a = b$, and their intended positions are adjacent and $\hat{u}$ apart, formally $p_1 - p_2 = \hat{u}$, in which case we set $G_{i,j}^{\hat{u}} = 0$. We call a configuration σ *intended* if all species are next to their six intended von Neumann neighbors and all agree on the intended label configuration, formally $\forall u, v \in V$, $\text{cfg}(\sigma(u)) = \text{cfg}(\sigma(v))$, and there exists a lattice translation and rotation $\tau_d R$ such that $\forall u \in V$, $\text{pos}(\sigma(u)) = \tau_d R(u)$; otherwise we call configurations *unintended*. By construction, the contribution of the parameters $G_{i,j}^{\hat{u}}$ to the energy of an intended configuration is zero and to unintended configurations is at least Λ .

We set the per-molecule energies G_i as follows. For all species types m , if $\text{pos}(m) = (0, 0, 0)$, we set $G_m = -kT \ln Q_K(\text{cfg}(m))$, i.e., we designate the species type $M_{\beta_i}^{(0,0,0)}$ as the *leader of label configuration* β_i , and set its per-molecule energy to $-kT \ln Q_K(\hat{\beta}_i^*)$, with the intention that $M_{\beta_i}^{(0,0,0)}$ is the only energy contribution to its intended configuration. For all other species types m such that $\text{pos}(m) \neq (0, 0, 0)$, we set $G_m = 0$.

With the set of molecule types M and parameters thus defined, we now aim to argue correctness. First note that for each elected configuration $\hat{\beta}_i^*$ with $Q_K(\hat{\beta}_i^*) > 0$, there is an intended configuration σ_i consisting of the species types $M_{\beta_i}^{(x, y, z)}$ each in their intended positions. Because it is an intended configuration, this configuration has zero energy contribution from $G_{i,j}^{\hat{u}}$ parameters, and has one G_i contribution from $G_{M_{\beta_i}^{(0,0,0)}}$, so $G(\sigma_i) = -kT \ln Q_K(\hat{\beta}_i^*)$, and so $P(\sigma_i) = \frac{1}{Z} \exp(-G(\sigma_i)/kT) = \frac{1}{Z} Q_K(\hat{\beta}_i^*)$. For any lattice translation τ_d and lattice rotation R , the label configuration $\beta_j = (\tau_d \circ R)\hat{\beta}_i^*$ matches labels with the microstate configuration $\sigma_j = (\tau_d \circ R)\sigma_i$, and since rotations/translations do not affect energy, $G(\sigma_j) = G(\sigma_i)$, and so $P(\sigma_j) = P(\sigma_i)$.

Finally, let $\mathcal{I}$ be the set of intended configurations and $\mathcal{U}$ be the set of unintended configurations. By construction, for every label configuration β there is exactly one intended configuration σ_β with matching labels, namely the translation/rotation of the elected configuration for the Euclidean equivalence class of β . Further, $\exp(-G(\sigma_\beta)/kT) = Q_K(\beta)$. The total Boltzmann weight of all intended configurations is $\sum_{\sigma \in \mathcal{I}} \exp(-G(\sigma)/kT) = \sum_{\beta} Q_K(\beta) = 1$. By construction, every unintended configuration has at least one $G_{i,j}^u$ contributing Λ to its energy. Note that each G_m is nonnegative since $0 < Q_K(\beta) \leq 1$ so $\ln Q_K(\beta) \leq 0$. Thus each unintended configuration $\sigma \in \mathcal{U}$ has $G(\sigma) \geq \Lambda$. The goal is to set Λ so that unintended configurations contribute negligibly to the partition function Z . Since the set of all configurations Ω is finite and $|\Omega| = |M|^{|V|}$, we have $\sum_{\sigma \in \mathcal{U}} \exp(-G(\sigma)/kT) \leq |\Omega| \exp(-\Lambda/kT)$. Therefore we can bound the partition function:

$$1 \leq Z = \sum_{\sigma \in \mathcal{I}} \exp(-G(\sigma)/kT) + \sum_{\sigma \in \mathcal{U}} \exp(-G(\sigma)/kT) \leq 1 + |\Omega| \exp(-\Lambda/kT).$$

We choose Λ such that $|\Omega| \exp(-\Lambda/kT) < \exp(\varepsilon) - 1$ so that $Z < \exp(\varepsilon)$. For any label configuration β , since there is one intended configuration σ_β with $\mathcal{K} \circ \sigma_\beta = \beta$, we have

$$P_{\mathcal{K}}(\beta) = \sum_{\sigma | \mathcal{K} \circ \sigma = \beta} P(\sigma) \geq P(\sigma_\beta) = \frac{Q_K(\beta)}{Z},$$

thus,

$$D_{\text{KL}}(Q_K || P_{\mathcal{K}}) = \sum_{\beta} Q_K(\beta) \log \frac{Q_K(\beta)}{P_{\mathcal{K}}(\beta)} \leq \sum_{\beta} Q_K(\beta) \log Z = \log Z < \varepsilon. \quad \blacktriangleleft$$

4.2 Universality in the isotropic grand-canonical model with symmetry-breaking boundary conditions

With isotropic molecules, a construction conceptually similar to the anisotropic construction also yields universality for label distributions. However, the isotropic model needs a symmetry-breaking boundary condition to effectively orient the molecules, since anisotropy cannot be used. The global orientation is achieved by propagating orientation from the boundary inward. This boundary condition removes the requirement that the target distribution must be Euclidean invariant. It is worth noting that, as in the picture of environmental clamping in the Boltzmann liquids framework discussed in the Introduction and Figure 1, the boundary condition inducing orientation can be seen as a type of conditioning of the probability distribution. We first formalize the boundary condition. Given a cubic lattice $V^{+2} = \{-1, \dots, L\}^3$, we define the *boundary* $\partial V^{+2} = \{(x, y, z) \in V^{+2} \mid x \in \{-1, L\} \text{ or } y \in \{-1, L\} \text{ or } z \in \{-1, L\}\}$. The *boundary condition* is that for each position in the boundary $p_\partial \in \partial V^{+2}$, we include a species M^{p_∂} that is fixed in that position.

► **Theorem 6.** *The isotropic grand-canonical model with a boundary condition is universal for label distributions.*

The proof can be found in Appendix A.5. The construction and proof are similar to those of Theorem 5, with an additional inductive argument that the boundary condition enforces a global orientation. We expect that the boundary condition can be relaxed to, e.g., a constant-sized corner of the cubic lattice. The boundary condition here is a special case of clamping as discussed in the Boltzmann liquids framework [7]. This result suggests that symmetry-breaking by a fixed boundary condition, which could be introduced by the

environment, can play a role in the complexity of spatial distributions of isotropic molecules internal to the boundary. We note that this is counterintuitive: small examples such as Figure 1(a) give the impression that conditioning a distribution reduces complexity by reducing the state space. However, this result suggests that conditioning reduces the state space in a way that is useful or expressive, in this case by breaking symmetries and likely increasing the class of representable distributions.

5 Discussion

The label-distribution formalism allows the system’s number of parameters to scale independently of the support of the target distribution by including more underlying species per label. However, our universality construction uses a number of species exponential in the lattice volume and number of labels: $|M| = \Theta(|K|^{|V|})$. This is not unexpected: counting arguments like the one in Section 3.1 suggest that the exponential number of parameters is required. When considering all possible distributions, most look algorithmically random [17], in the sense that most have high Kolmogorov complexity: the shortest programs on a universal Turing machine which output the distributions must effectively hard-code them. In other words, most distributions look like uncorrelated noise. Future work, especially work including the use of learning algorithms such as the contrastive learning used in the Boltzmann liquids framework [7], will help identify the types of low-Kolmogorov complexity distributions that can be achieved with a reasonable number of parameters. In life, coding theory arguments suggest that genotype-phenotype maps may naturally favor low-Kolmogorov complexity phenotypes [13], aligning with our expectation that lifelike dynamic spatial distributions may be learnable. Exponential-parameter universality results such as ours are not prescriptions for suggested constructions, but instead suggest that there are no formal barriers to representing arbitrary macrostate distributions and thus invite the study of new universal macrostate pictures as well as the investigation into which spatial distributions are practical for equilibrium systems of programmable pairwise local interactions.

Our universality result for label distributions is conceptually a “hard-coded lookup table”: a disjoint subset of the parameters directly encode each target configuration of the distribution. While lookup-table results are effective for proving the absence of formal restrictions against arbitrary distributions, neural-network literature includes stronger approximation results for structured classes which argue that approximation error can reduce at a controlled rate as more parameters are utilized (e.g., see Section 6 of [27] for a survey of smooth approximation results for multi-layer perceptrons). What quantitative structural properties of spatial distributions might define a “distribution complexity” such that, given a target approximation error, the required number of parameters scales smoothly with that complexity? How might physical constraints such as locality guide the identification of such properties?

The hard-coding of our universality construction calls to mind uniquely-addressed self-assembly, where a two-dimensional square lattice or three-dimensional cubic lattice consisting of unique species for each position is constructed, and shapes are assembled by leaving out subsets of the components [41, 14, 24]. Beyond unique addressing, although kinetic in nature, algorithmic self-assembly [10] has articulated how the number of unique species types required relates to the complexity of a target shape, with theoretical constructions achieving optimal results [34]. Another studied task beyond assembly of shapes is to design a set of self-assembling tiles which assembles a pattern of labels, where species types are labeled as in our labeling observable. The Pattern self-assembly Tile-set Synthesis (PATs) problem [19] aims to synthesize the smallest set of species that assembles a target label pattern. Our

universality construction can be loosely viewed as an extension of this problem to the self-assembly of *distributions* of label patterns, as opposed to deterministic self-assembly of a single label pattern. Future work could establish this connection rigorously and provide synthesis methods for the distribution variant of the PATS problem, or inversely, algorithmic self-assembly theory may inform how spatial distribution complexity relates to the number of parameters required in our lattice models.

The utilization of large positive (repulsive) interactions Λ in our universality results suggests that with any reasonably-defined dynamics, the system will have large energy barriers between its intended configurations. As such, this construction is much more like crystalline self-assembly than liquid-like molecular rearrangement, where we define liquid-like as able to explore its configuration space on short timescales. We posit, along with [7], that restricting to parameters that enable liquid-like behavior is an important desideratum to implement effective neural networks of this kind, as short-timescale reconfiguration corresponds to how quickly the molecules will react to an environmental stimulus, and further, systems with slow rearrangement dynamics do not explore their whole Boltzmann distribution on reasonable timescales and so will be sensitive to initial condition and require another kind of analysis. Thus, a motivated task is to identify a variant of universality (appropriate choice of observable, or limitation on target distributions) that can be achieved while maintaining liquid-like interaction energies (i.e., energies $\sim kT$).

Prior work [22, 35, 26] has identified *equilibrium dimerization networks* as a fruitful direction for programming well-mixed, non-spatial molecular information processing. These networks consist of dimerization reactions like $X_i + X_j \rightleftharpoons D_{i,j}$, where the equilibrium is determined by energies of each X_i and $D_{i,j}$ species, effectively tuned by the pairwise interaction energy between univalent X_i and X_j molecules. Such networks can be expressed in our anisotropic model, since univalence can be expressed by setting $G_{i,j}^u$ appropriately. Further, our model can also be extended to include reactions: when molecules of species A and B are neighbors, they may transition to species C and D . Reactions of this form can induce complex reachability constraints in canonical models, potentially enhancing their distribution representation capability [28, 29]. That our anisotropic lattice model captures well-mixed reaction networks situates it as a prototype or potential candidate for a unified model of molecular information processing.

Although our labeling observable and universality definitions allow positive constructions exclusively for the grand-canonical models, we expect that the canonical models can be universal in distributions defined by other observables. For example, an enticing observable is to consider a subset of lattice positions, e.g. a two-dimensional surface on a three-dimensional lattice, as *visible* observed positions P , and define the rest of the positions as *hidden* positions H . Then, scaling $|H|$ while keeping $|P|$ fixed allows an increased number of species and thus parameters. We anticipate such a formulation may yield universality over microstate distributions of the visible positions in canonical models. In addition, this formulation allows for conditioning the distribution by physically clamping regions of the visible surface P , in contrast to our label observable that lacks a clear clamping mechanism.

Our explicit constructions of impossible distributions shown in Section 3.2 provide an initial exploration into the class of microstate distributions that can be represented by our models. These counterexamples can be seen as “the XOR of Boltzmann liquids”, in analogy to the classical example of XOR as a non-linearly-separable dataset that illuminates the limits of single-layer perceptrons [20]. Future work may extend these counterexamples into a complete characterization of achievable microstate distributions.

Our universality result encourages pushing the limits of complexity in programmed equilibrium spatial distributions. Future work aligned with the Boltzmann liquids' notion of clamping [7] will illustrate how these complex molecular spatial arrangements may respond dynamically to environmental cues as the cues condition their Boltzmann distribution. Ultimately, this work provides theoretical evidence that programming equilibrium distributions of pairwise-interacting molecules may be sufficient to yield a complexity of spatial distributions matching or surpassing those demonstrated by life.

References

- 1 David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985. doi:10.1207/S15516709COG0901_7.
- 2 Siddharth Agarwal, Dino Osmanovic, Mahdi Dizani, Melissa A Klocke, and Elisa Franco. Dynamic control of DNA condensation. *Nature Communications*, 15(1):1915, 2024.
- 3 Paul C. Bethke and Russell L. Jones. Vacuoles and prevacuolar compartments. *Current Opinion in Plant Biology*, 3(6):469–475, 2000.
- 4 Clifford P. Brangwynne, Christian R. Eckmann, David S. Courson, Agnieszka Rybarska, Cathrin Hoege, Javad Gharakhani, Frank Jülicher, and Anthony A. Hyman. Germline P granules are liquid droplets that localize by controlled dissolution/condensation. *Science*, 324(5935):1729–1732, 2009.
- 5 Daniele Cappelletti, Andrés Ortiz-Muñoz, David F Anderson, and Erik Winfree. Stochastic chemical reaction networks for robustly approximating arbitrary probability distributions. *Theoretical Computer Science*, 801:64–95, 2020. doi:10.1016/J.TCS.2019.08.013.
- 6 Aria S Chaderjian, Sam Wilken, and Omar A Saleh. Diverse, distinct, and densely packed DNA nanostar droplets. *Proceedings of the National Academy of Sciences*, 123(7):e2523462123, 2026.
- 7 Cameron Chalk, Salvador Buse, Krishna Shrinivas, Arvind Murugan, and Erik Winfree. Learning and inference in a lattice model of multicomponent condensates. In *30th International Conference on DNA Computing and Molecular Programming (DNA 30)*, pages 5–1. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.DNA.30.5.
- 8 Priyamvada Chugh, Andrew G. Clark, Matthew B. Smith, Davide A. D. Cassani, Kai Dierkes, Anan Ragab, Philippe P. Roux, Guillaume Charras, Guillaume Salbreux, and Ewa K. Paluch. Actin cortex architecture regulates cell surface tension. *Nature Cell Biology*, 19(6):689–697, 2017.
- 9 George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. doi:10.1007/BF02551274.
- 10 David Doty. Theory of algorithmic self-assembly. *Communications of the ACM*, 55(12):78–88, 2012. doi:10.1145/2380656.2380675.
- 11 Daniel A Fletcher and R Dyche Mullins. Cell mechanics and the cytoskeleton. *Nature*, 463(7280):485–492, 2010.
- 12 William M Jacobs and Daan Frenkel. Phase transitions in biological systems with many components. *Biophysical Journal*, 112(4):683–691, 2017.
- 13 Iain G Johnston, Kamaludin Dingle, Sam F Greenbury, Chico Q Camargo, Jonathan PK Doye, Sebastian E Ahnert, and Ard A Louis. Symmetry and simplicity spontaneously emerge from the algorithmic nature of evolution. *Proceedings of the National Academy of Sciences*, 119(11):e2113883119, 2022.
- 14 Yonggang Ke, Luvena L Ong, William M Shih, and Peng Yin. Three-dimensional structures self-assembled from DNA bricks. *Science*, 338(6111):1177–1183, 2012.
- 15 Anna Kicheva and James Briscoe. Control of tissue development by morphogens. *Annual Review of Cell and Developmental Biology*, 39(1):91–121, 2023.

- 16 Nicolas Le Roux and Yoshua Bengio. Representational power of restricted Boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649, 2008. doi:10.1162/NECO.2008.04-07-510.
- 17 Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Texts in Computer Science. Springer, Cham, 4 edition, 2019.
- 18 Andrew S. Lyon, William B. Peeples, and Michael K. Rosen. A framework for understanding the functions of biomolecular condensates across scales. *Nature Reviews Molecular Cell Biology*, 22(3):215–235, 2021.
- 19 Xiaojun Ma and Fabrizio Lombardi. Synthesis of tile sets for DNA self-assembly. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(5):963–967, 2008. doi:10.1109/TCAD.2008.917973.
- 20 Marvin Minsky and Seymour A. Papert. *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 2017.
- 21 Sahil Nagpal, Karthikeyan Swaminathan, Daniel Beaudet, Maud Verdier, Samuel Wang, Christopher L. Berger, Florian Berger, and Adam G. Hendricks. Optogenetic control of kinesin-1, -2, -3 and dynein reveals their specific roles in vesicular transport. *Cell Reports*, 43(8):114649, 2024.
- 22 Maxim P Nikitin. Non-complementary strand commutation as a fundamental alternative for information processing by DNA and gene regulation. *Nature Chemistry*, 15(1):70–82, 2023.
- 23 Shigenori Nonaka, Hidetaka Shiratori, Yukio Saijoh, and Hiroshi Hamada. Determination of left-right patterning of the mouse embryo by artificial nodal flow. *Nature*, 418(6893):96–99, 2002.
- 24 Luvena L Ong, Nikita Hanikel, Omar K Yaghi, Casey Grun, Maximilian T Strauss, Patrick Bron, Josephine Lai-Kee-Him, Florian Schueder, Bei Wang, Pengfei Wang, et al. Programmable self-assembly of three-dimensional nanostructures from 10,000 unique components. *Nature*, 552(7683):72–77, 2017.
- 25 Vidhi Pareek, Hua Tian, Nicholas Winograd, and Stephen J. Benkovic. Metabolomics and mass spectrometry imaging reveal channeled de novo purine synthesis in cells. *Science*, 368(6488):283–290, 2020.
- 26 Jacob Parres-Gold, Matthew Levine, Benjamin Emert, Andrew Stuart, and Michael B Elowitz. Contextual computation by competitive protein dimerization networks. *Cell*, 188(7):1984–2002, 2025.
- 27 Allan Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8:143–195, 1999.
- 28 William Poole, Andrés Ortiz-Muñoz, Abhishek Behera, Nick S. Jones, Thomas E. Ouldridge, Erik Winfree, and Manoj Gopalkrishnan. Chemical Boltzmann machines. In Robert Brijder and Lulu Qian, editors, *DNA Computing and Molecular Programming*, pages 210–231, Cham, 2017. Springer International Publishing. doi:10.1007/978-3-319-66799-7_14.
- 29 William Poole, Thomas Ouldridge, Manoj Gopalkrishnan, and Erik Winfree. Detailed balanced chemical reaction networks as generalized Boltzmann machines. *arXiv preprint arXiv:2205.06313*, 2022. doi:10.48550/arXiv.2205.06313.
- 30 Catherine Rabouille and Simon Alberti. Cell adaptation upon stress: the emerging role of membrane-less compartments. *Current Opinion in Cell Biology*, 47:34–42, 2017.
- 31 Venki Ramakrishnan. Ribosome structure and the mechanism of translation. *Cell*, 108(4):557–572, 2002.
- 32 Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.
- 33 Paul WK Rothmund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297–302, 2006.
- 34 David Soloveichik and Erik Winfree. Complexity of self-assembled shapes. *SIAM Journal on Computing*, 36(6):1544–1569, 2007. doi:10.1137/S0097539704446712.

- 35 Alexei V Tkachenko, Bortolo Matteo Moggetti, and Sergei Maslov. Evolutionary chemical learning in dimerization networks. *arXiv preprint arXiv:2506.14006*, 2025. doi:10.48550/arXiv.2506.14006.
- 36 Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer, New York, 2009. doi:10.1007/B13794.
- 37 Ronald D Vale. The molecular motor toolbox for intracellular transport. *Cell*, 112(4):467–480, 2003.
- 38 Gerrit Van Meer, Dennis R Voelker, and Gerald W Feigenson. Membrane lipids: where they are and how they behave. *Nature Reviews Molecular Cell Biology*, 9(2):112–124, 2008.
- 39 Carl G Wagner. *A first course in enumerative combinatorics*, volume 49. American Mathematical Society, 2020.
- 40 Jianbang Wang, Zhenzhen Li, and Itamar Willner. Dynamic reconfigurable DNA nanostructures, networks and materials. *Angewandte Chemie International Edition*, 62(18):e202215332, 2023.
- 41 Bryan Wei, Mingjie Dai, and Peng Yin. Complex shapes self-assembled from single-stranded DNA tiles. *Nature*, 485(7400):623–626, 2012.

A

 Appendix

A.1 Configurations in the same fiber which are not translations/rotations

B	B	A	B	A	A	B	B
B	A	B	A	A	B	A	B
A	B	A	B	B	A	B	A
B	A	A	A	B	B	A	A

■ **Figure 5** Two configurations with the same sufficient statistics which are not translations/rotations of each other. The neighborhood is von Neumann with periodic boundaries.

Here we provide an example of two configurations, shown in Figure 5, with the same sufficient statistics vector $t = (n_A, n_B, n_{A,A}, n_{A,B}, n_{B,B}) = (8, 8, 4, 12, 4)$, which are not rotations or translations. A short proof that they are not rotations or translations is to note that in the configuration in Figure 5 right, no B has four B neighbors, whereas the configuration in Figure 5 left, has a B with four B neighbors.

A.2 Counting the size of probability vector set $\mathcal{D}_{\mathcal{T}}$

Recall

$$\mathcal{D}_{\mathcal{T}} = \left\{ q : \mathcal{T}_{\Omega} \rightarrow [0, 1] \mid q(t) = \frac{z_t}{|\mathcal{T}_{\Omega}|}, z_t \in \mathbb{Z}_{\geq 0}, \sum_{t \in \mathcal{T}_{\Omega}} z_t = |\mathcal{T}_{\Omega}| \right\}.$$

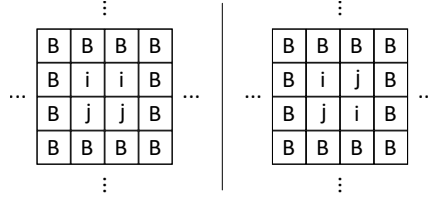
Let $N = |\mathcal{T}_{\Omega}|$ and enumerate $\mathcal{T}_{\Omega} = \{t_1, \dots, t_N\}$. Consider the function:

$$\Phi : q \rightarrow (z_1, \dots, z_N), \quad z_i = Nq(t_i).$$

Φ is a bijection between $\mathcal{D}_{\mathcal{T}}$ and the set of *weak compositions of N into N parts*:

$$\left\{ (z_1, \dots, z_N) \in \mathbb{Z}_{\geq 0}^N \mid \sum_{i=1}^N z_i = N \right\}.$$

The set of weak compositions of N into N parts can be shown to have size $\binom{2N-1}{N-1}$ via stars-and-bars counting (see [39]).



■ **Figure 6** Two subconfigurations of a subset of positions of a lattice, with $i, j \in M$. Assuming von Neumann neighbors, the internal 2×2 positions in the left subconfiguration contribute $(n_{i,i}, n_{i,j}, n_{j,j}) = (1, 2, 1)$, while the internal 2×2 positions on the right contribute $(n_{i,i}, n_{i,j}, n_{j,j}) = (0, 4, 0)$.

A.3 The set of feasible sufficient statistics vectors $\mathcal{T}_\Omega$ scales exponentially in the number of parameters in the regime $|V| = \Theta(|M|^2)$

Here, for the sake of the counting argument of Section 3.1, we argue that $\mathcal{T}_\Omega := \{t(\sigma) \mid \sigma \in \Omega\}$, the set of sufficient statistics feasible for a given configuration space Ω , scales exponentially with the number of parameters $n = |\theta|$ in a construction where $|V| = \Theta(|M|^2)$. To do so, we consider a subset of configurations, where each configuration has a fixed count of species (and thus is a subset of both canonical and grand-canonical configuration spaces). Each configuration consists of 2×2 blocks of species, separated by a “buffer” species B that keeps the blocks separated (see Figure 6).

Assume the lattice is large enough to contain $\binom{|M|}{2}$ disjoint 2×2 blocks, one for each unordered pair of species $i, j \in M$, separated by buffer B . As shown in Figure 6, each i, j block can have one of at least two distinct arrangements, which give distinct contributions to $t(\sigma)$: $(n_{i,i}, n_{i,j}, n_{j,j}) = (1, 2, 1)$ or $(n_{i,i}, n_{i,j}, n_{j,j}) = (0, 4, 0)$. The gadgets are disjoint, so each choice can be made independently. Therefore $|\mathcal{T}_\Omega| \geq 2^{\binom{|M|}{2}}$.

In the isotropic canonical model, the number of parameters is $\frac{|M|(|M|+1)}{2}$, and in the isotropic grand-canonical model, the number of parameters is $|M| + \frac{|M|(|M|+1)}{2}$. In the anisotropic models with fixed, finite neighborhood and orientations sets, there is a constant multiplicative factor to the number of parameters. Thus, in each model variant, $n = \Theta(|M|^2)$. Since $\binom{|M|}{2} = \Theta(|M|^2)$, we have $|\mathcal{T}_\Omega| \geq 2^{\Omega(n)}$, i.e., the number of feasible sufficient statistic vectors t scales exponentially in the number of parameters, if $|V|$ is allowed to scale as $|M|^2$.

A.4 Microstate universality counterexamples

A.4.1 Isotropic grand-canonical model

Consider a ring of four positions with left-right edges and no top-bottom edges (see Figure 4(a), right). The species are A and B , so that $\theta = (G_A, G_B, G_{A,A}, G_{A,B}, G_{B,B})$ and $t(\sigma) = (n_A, n_B, n_{A,A}, n_{A,B}, n_{B,B})$. There are six feasible t values as shown in Figure 4(a), right, and the counterexample distribution is defined as follows:

$$\begin{aligned} \text{If } t(\sigma) = t_0 = (1, 3, 0, 2, 2), \text{ set } Q(\sigma) &= \frac{1}{64}. \\ \text{If } t(\sigma) = t_1 = (2, 2, 1, 2, 1), \text{ set } Q(\sigma) &= \frac{1}{32}. \\ \text{If } t(\sigma) = t_2 = (3, 1, 2, 2, 0), \text{ set } Q(\sigma) &= \frac{1}{64}. \\ \text{If } t(\sigma) = t_3 = (4, 0, 4, 0, 0), \text{ set } Q(\sigma) &= \frac{1}{8}. \end{aligned}$$

If $t(\sigma) = t_4 = (2, 2, 0, 4, 0)$, set $Q(\sigma) = \frac{1}{4}$.

If $t(\sigma) = t_5 = (0, 4, 0, 0, 4)$, set $Q(\sigma) = \frac{1}{8}$.

Note that $t_1 = \frac{t_0+t_2}{2}$ and that the probabilities sum to 1. Towards contradiction, assume Q satisfies the affine log-probabilities constraint. As in the isotropic canonical model example in the main body, this means Q satisfies $u(t_1)^2 = u(t_0)u(t_2)$, however, $(\frac{1}{32})^2 \neq (\frac{1}{64})^2$, thus a contradiction. Therefore there does not exist θ for the isotropic grand canonical model such that the Boltzmann distribution is Q .

A.4.2 Anisotropic canonical model

The approach is to embed the isotropic canonical counterexample (Section 3.2) into the anisotropic case. Let V be a ring of 6 positions as in the isotropic canonical counterexample (Figure 4(a), left). The set of species types is $M_{\text{type}} = \{A, B\}$. We denote a standard orientation for each type: $A_0 = (A, o_1), B_0 = (B, o_1)$. We fix the canonical counts to have three of A and three of B . With direction $+\hat{x} = (1, 0, 0)$, we consider the subset of statistics $t^{+\hat{x}} = (n_{A_0, A_0}^{+\hat{x}}, n_{A_0, B_0}^{+\hat{x}}, n_{B_0, A_0}^{+\hat{x}}, n_{B_0, B_0}^{+\hat{x}})$. Note that the statistics $n_{X, Y}^{-\hat{x}} = n_{Y, X}^{+\hat{x}}$. Since we consider V as a one-dimensional ring and the following specifications of $t^{+\hat{x}}$ are for configurations in which all species are in their standard orientations A_0, B_0 , each $t^{+\hat{x}}$ specifies a full fiber since the $-\hat{y}, +\hat{y}, -\hat{z}, +\hat{z}$ directions are unavailable for their respective statistics (e.g., $n_{A_0, B_0}^{-\hat{y}} = 0$) and non-standard oriented species have zero for their statistics (e.g., $n_{(A, o_2), (B, o_2)}^{+\hat{x}} = 0$). The $A_0 B_0 A_0 B_0 A_0 B_0$ configurations have

$$t_0^{+\hat{x}} = (0, 3, 3, 0),$$

the $A_0 A_0 B_0 B_0 A_0 B_0$ configurations have

$$t_1^{+\hat{x}} = (1, 2, 2, 1),$$

and the $A_0 A_0 A_0 B_0 B_0 B_0$ configurations have

$$t_2^{+\hat{x}} = (2, 1, 1, 2).$$

These satisfy $t_1^{+\hat{x}} = \frac{t_0^{+\hat{x}} + t_2^{+\hat{x}}}{2}$.

We now define our fiber-uniform target distribution Q by assigning a microstate probability $u(t)$ to each fiber F_t . We set $u(t_0^{+\hat{x}}) = u(t_2^{+\hat{x}}) = p$, and $u(t_1^{+\hat{x}}) = 2p$ such that $p > 0$. Probabilities for each other fiber are set to arbitrary positive real numbers such that the sum of probabilities is one. As in Section 3.2, assume Q satisfies the affine log-probabilities constraint, then we have $u(t_1^{+\hat{x}})^2 = u(t_0^{+\hat{x}})u(t_2^{+\hat{x}})$, therefore $(2p)^2 = p^2$, which gives a contradiction since $p > 0$.

A.4.3 Anisotropic grand-canonical model

Here, we embed the isotropic grand-canonical counterexample distribution (Appendix A.4.1) in the anisotropic case. Let V be a ring of four positions, as in the isotropic grand-canonical counterexample (Figure 4(a), right). The set of species types is $M_{\text{type}} = \{A, B\}$. We denote a standard orientation for each type: $A_0 = (A, o_1), B_0 = (B, o_1)$. With direction $+\hat{x} = (1, 0, 0)$, we consider the subset of statistics $t^{+\hat{x}} = (n_A, n_B, n_{A_0, A_0}^{+\hat{x}}, n_{A_0, B_0}^{+\hat{x}}, n_{B_0, A_0}^{+\hat{x}}, n_{B_0, B_0}^{+\hat{x}})$. Note that the statistics $n_{X, Y}^{-\hat{x}} = n_{Y, X}^{+\hat{x}}$. Since we consider V as a one-dimensional ring and the

following specifications of $t^{+\hat{x}}$ are for configurations in which all species are in their standard orientations A_0, B_0 , each $t^{+\hat{x}}$ specifies a full fiber since the $-\hat{y}, +\hat{y}, -\hat{z}, +\hat{z}$ directions are unavailable for their respective statistics (e.g., $n_{A_0, B_0}^{-\hat{y}} = 0$) and non-standard oriented species have zero for their statistics (e.g., $n_{(A, o_2), (B, o_2)}^{+\hat{x}} = 0$). The $A_0 B_0 B_0 B_0$ configurations have

$$t_0^{+\hat{x}} = (1, 3, 0, 1, 1, 2),$$

the $A_0 A_0 B_0 B_0$ configurations have

$$t_1^{+\hat{x}} = (2, 2, 1, 1, 1, 1),$$

and the $A_0 A_0 A_0 B_0$ configurations have

$$t_2^{+\hat{x}} = (3, 1, 2, 1, 1, 0).$$

These satisfy $t_1^{+\hat{x}} = \frac{t_0^{+\hat{x}} + t_2^{+\hat{x}}}{2}$.

We now define our fiber-uniform target distribution Q by assigning a microstate probability $u(t)$ to each fiber F_t . We set $u(t_0^{+\hat{x}}) = u(t_2^{+\hat{x}}) = p$, and $u(t_1^{+\hat{x}}) = 2p$ such that $p > 0$. Probabilities for each other fiber are set to arbitrary positive real numbers such that the sum of probabilities is one. As in Section 3.2, assume Q satisfies the affine log-probabilities constraint, then we have $u(t_1^{+\hat{x}})^2 = u(t_0^{+\hat{x}})u(t_2^{+\hat{x}})$, therefore $(2p)^2 = p^2$, which gives a contradiction since $p > 0$.

A.5 Proof of Theorem 6

Proof. The construction is similar to the proof of Theorem 5, except with the addition of interactions with the boundary species. Note the boundary condition as defined just before this theorem's statement. For each label configuration β_i such that $Q_K(\beta_i) > 0$, we include species $M_{\beta_i}^{(x,y,z)}$ for $(x, y, z) \in V$ and call $\text{pos}(M_{\beta_i}^{(x,y,z)}) = (x, y, z)$ the *intended position* and $\text{cfg}(M_{\beta_i}^{(x,y,z)}) = \beta_i$ the *intended label configuration*. We set $G_{i,j} = 0$ if $\text{pos}(i)$ and $\text{pos}(j)$ are von Neumann neighbors and $\text{cfg}(i) = \text{cfg}(j)$, and otherwise set $G_{i,j} = \Lambda$. Additionally, for all β_i , we set $G_{i,j} = 0$ for $i = M_{\beta_i}^{p_{\hat{u}}}$ and $j = M_{\beta_i}^p$ for $p_{\hat{u}}$ on the boundary and p such that p and $p_{\hat{u}}$ are von Neumann neighbors. We call a (microstate) configuration σ *intended* if all species are next to their six intended von Neumann neighbors and all agree on intended label configuration, formally for all $u \in V^{+2}$, $\text{pos}(\sigma(u)) = u$ and for all $u, v \in V$, $\text{cfg}(\sigma(u)) = \text{cfg}(\sigma(v))$, otherwise we call configurations *unintended*. For each β_i , for species $M_{\beta_i}^{(0,0,0)}$, we set $G_{M_{\beta_i}^{(0,0,0)}} = -kT \ln Q_K(\beta_i)$. For all other species i , we set $G_i = 0$.

The key is to show that any unintended configuration σ must have $G(\sigma) \geq \Lambda$. Let $S_k = \{(x, y, z) \mid x \in \{k-1, L-k\} \vee y \in \{k-1, L-k\} \vee z \in \{k-1, L-k\}\}$, i.e., S_k contains the k th hollow cube of points moving inwards from the boundary S_0 . We define $H(k)$ as the proposition: if there is no Λ contribution from interactions between species in $S_0 \cup \dots \cup S_k$, then all these species agree on label configuration and are in their intended positions. $H(0)$ is always true since the boundary is fixed and provides no Λ contribution. We induct on $H(k)$. Assuming $H(k-1)$, if there is no Λ contribution in $S' = S_0 \cup \dots \cup S_{k-1}$, we know all species in S_{k-1} agree on label configuration and are in their intended positions. Consider the contrapositive of the inductive hypothesis $H(k)$: if some species in S_k disagrees on label configuration or is not in its intended position, then there is a Λ contribution. The contrapositive is true since disagreement on label configuration between a species in S_k and one of its von Neumann neighbors in S_{k-1} contributes a Λ term, and similarly a species in S_k

which is not in its intended position contributes a Λ term with its von Neumann neighbors in S_{k-1} which are in their intended positions. Therefore $H(k)$ is true for all k , and so the contrapositive “if some species do not agree on label configuration or are not in their intended positions, then there is a Λ contribution from interactions between species in $S_0 \cup \dots \cup S_k$ ” is true. This implies that every unintended configuration σ has $G(\sigma) \geq \Lambda$.

Finally, we set Λ in order to achieve KL divergence less than ε . Let $\mathcal{I}$ be the set of intended configurations and $\mathcal{U}$ be the set of unintended configurations. For each label configuration β , there is exactly one intended configuration σ_β with $\mathcal{K} \circ \sigma_\beta = \beta$, namely the configuration in which each interior position $p \in V$ contains the species M_β^p , while the boundary positions contain their fixed boundary species. By construction, every interaction term in this intended configuration contributes zero, except for the single leader species $M_\beta^{(0,0,0)}$, whose per-molecule energy is $G_{M_\beta^{(0,0,0)}} = -kT \ln Q_K(\beta)$. Therefore $G(\sigma_\beta) = -kT \ln Q_K(\beta)$ so $\exp(-G(\sigma_\beta)/kT) = Q_K(\beta)$. Summing over intended configurations gives

$$\sum_{\sigma \in \mathcal{I}} \exp(-G(\sigma)/kT) = \sum_{\beta} Q_K(\beta) = 1.$$

By the inductive proof above, every unintended configuration has at least one Λ penalty, and all other per-molecule energy contributions are nonnegative, since $Q_K(\beta) \leq 1$ implies $-kT \ln Q_K(\beta) \geq 0$. Therefore every $\sigma \in \mathcal{U}$ satisfies $G(\sigma) \geq \Lambda$. Since the configuration space is finite, we have $\sum_{\sigma \in \mathcal{U}} \exp(-G(\sigma)/kT) \leq |\Omega| \exp(-\Lambda/kT)$. We choose Λ so that $|\Omega| \exp(-\Lambda/kT) < \exp(\varepsilon) - 1$. Then $Z < \exp(\varepsilon)$. For any label configuration β , $P_K(\beta) = \sum_{\sigma | \mathcal{K} \circ \sigma = \beta} P(\sigma) \geq P(\sigma_\beta) = \frac{Q_K(\beta)}{Z}$. Finally,

$$D_{\text{KL}}(Q_K || P_K) = \sum_{\beta} Q_K(\beta) \log \frac{Q_K(\beta)}{P_K(\beta)} \leq \sum_{\beta} Q_K(\beta) \log Z = \log Z < \varepsilon. \quad \blacktriangleleft$$

Scalable Enumeration of Pareto-Optimal Polymers for Computing Equilibrium Concentrations

Archit Patil ✉ 

University of Texas at Austin, TX, USA

Minki Hhan ✉ 

Korea Advanced Institute of Science and Technology, Daejeon, South Korea

David Soloveichik ✉ 

University of Texas at Austin, TX, USA

Abstract

Predicting equilibrium concentrations of molecular complexes is essential for verifying the behavior of engineered DNA systems. However, a finite set of monomer types can in principle generate infinitely many complexes. We study this candidate-enumeration problem in a geometry-free, domain-level abstraction called a domain-monomer system, generalizing Thermodynamic Binding Networks (TBNs) to the unsaturated setting where not every possible bond need be formed. We define *Pareto-suboptimal* polymers as those that can be split into non-interacting parts, and show that restricting attention to *Pareto-optimal* polymers is thermodynamically justified: no Pareto-suboptimal polymer appears in any minimum free-energy configuration, and the total equilibrium concentration of such polymers is small. We prove that there are finitely many Pareto-optimal polymers and exactly characterize them via a Hilbert basis computation, extending prior work from the saturated TBN model. To scale this approach to large systems, we develop a framework that restricts the number of different monomer types that a single polymer contains, and uses combinatorial covering designs to reduce the number of Hilbert basis computations required. We benchmark the method on several families of DNA molecular programming systems, demonstrating order-of-magnitude speedups over direct computation while recovering nearly all equilibrium-relevant polymers.

2012 ACM Subject Classification Computer systems organization → Molecular computing; Theory of computation → Design and analysis of algorithms; Applied computing → Computational biology

Keywords and phrases Molecular computation, Hilbert Basis, Thermodynamic Binding Network, covering design, equilibrium concentration

Digital Object Identifier 10.4230/LIPIcs.DNA.32.12

Supplementary Material *Software (Github Repo)*: <https://github.com/architrahul/Pareto-polymer-enumerator>; archived at `swh:1:dir:b5917c1073ebd7f44f1d3c4b5420671254195cee`
Text (Supplementary Text): https://github.com/architrahul/Pareto-polymer-enumerator/blob/main/Supplementary_Text.pdf

Funding Schmidt Sciences Polymath Award to D.S., DOE grant DE-SC0024248, NSF SemiSynBio III: GOALI grant 2227578.

Acknowledgements We thank And Kaan Ata Yilmaz for valuable discussions and insights.

1 Introduction

Understanding and engineering complex molecular systems and behaviors requires both kinetic and thermodynamic modeling. In the context of dynamic DNA nanotechnology, NUPACK [10, 12] is widely used to analyze the minimum free energy configurations of complexes, and their concentrations at thermodynamic equilibrium. However, since NUPACK operates at the individual base-pair level, it is infeasible to use for large systems (many strands and/or large complex sizes). At the same time, molecular programming researchers have



© Archit Patil, Minki Hhan, and David Soloveichik;
licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 12; pp. 12:1–12:24

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

been developing systems of ever larger complexity. For example, recent work demonstrating DNA neural networks and logic circuits involved 700 distinct complexes and over 200 strands in a single test tube [7, 20]. Such complexity necessitates a higher level of abstraction.

Most engineered DNA systems are organized around domains, contiguous sequence regions that behave as logical binding units. Domain-level abstractions make different geometric commitments: some models enforce rigid geometry (e.g., tile assembly), while others ignore geometry and retain only which domain types can bind. In this paper we study the latter, geometry-free setting. Our formalism is a *domain-monomer system*, closely related to Thermodynamic Binding Networks (TBNs) [11, 4, 3, 15, 22], in which a monomer is a multiset of abstract domains, complementary domains can bind, and a polymer is a multiset of monomers. Unlike the saturated TBN setting which is meant to capture strong bonding, we do not require every possible bond to be formed. The solution might have polymers with unbound complementary domains, corresponding to a more realistic model that permits bonds of varying affinities. The central thermodynamic question is to determine the equilibrium distribution of polymer concentrations: which polymers form, and in what amounts? In engineered systems, some polymers are intended products and verifying correct behavior requires checking their equilibrium concentrations (e.g., that certain polymers are present in specific ratios), while others are undesired leakage.

In typical systems there is a relatively small set of polymers that dominate the equilibrium, with the rest present in negligible concentrations (a finite set of monomer types can generate infinitely many polymers). If we could a priori eliminate the vast majority of safe-to-neglect polymers, then equilibrium optimization itself would be efficient (tools such as COFFEE [24] can handle hundreds of thousands of candidate polymers). But constructing a sensible finite candidate set of polymers in a principled way appears nontrivial as it must be defined independently of the unknown equilibrium solution itself. We address this difficulty with a natural structural notion called *Pareto-optimality*. Informally, a polymer is Pareto-suboptimal if it can be split into two non-complementary polymers. More precisely, a polymer is Pareto-suboptimal if it is possible to reconfigure the monomers of the polymer into two or more separate polymers without decreasing overall bonding. Such a split preserves the total bond energy (enthalpy) while producing more separate complexes (increasing entropy), so Pareto-suboptimal polymers are thermodynamically disfavored. We formally justify the focus on Pareto-optimal polymers by showing that no Pareto-suboptimal polymer appears in a minimum free-energy configuration in a discrete model, and further that in the continuous equilibrium model their total concentration is bounded in terms of the total concentration of Pareto-optimal polymers. This makes Pareto-optimal polymers a principled candidate set for equilibrium analysis.

Prior work of Haley and Doty [15] connected Hilbert bases to polymer enumeration in the saturated TBN model, which takes the thermodynamic limit of strong bonds. Our goal is to extend this algebraic perspective to the more general unsaturated setting of possibly weak bonds, allowing different bond strengths corresponding to hybridization energies of different domains. Further, we want to make the analysis practical on large systems where direct Hilbert basis computation is far too expensive.

Our contributions are as follows.

- We bound the total concentration of Pareto-suboptimal polymers in geometry-free domain-monomer systems, justifying their exclusion.
- We show that the full set P^* of Pareto-optimal polymers is finite and can be characterized exactly via Hilbert basis enumeration. This extends the Hilbert-basis connection from the saturated TBN, discrete setting [15].

- We develop a scalable framework for enumerating the subset $P_t^* \subseteq P^*$ of Pareto-optimal polymers that use at most t monomer types (*support-boundedness restriction*). We first express P_t^* using Hilbert bases of smaller subsystems, and reduce the number of required subsystem computations by using covering designs. The resulting method returns a superset $\hat{P}_{k,t}$ satisfying $P_t^* \subseteq \hat{P}_{k,t} \subseteq P^*$. Importantly, the parameter t is interpretable: rather than imposing an opaque algebraic or numerical cutoff, it bounds the number of distinct monomer types in a polymer, a quantity often apparent from the design specification of the intended products and plausible leakage pathways.
- We describe implementation techniques, including a probe-and-prune heuristic for choosing the block size k , and benchmark the method on several families of molecular systems from the recent literature [22, 23, 21]. For example, on a 7-module linear cascade, the covering-design strategy with $(t, k) = (5, 25)$ reduces polymer enumeration time from >1000 s for direct full Hilbert basis computation to 24 s, without missing any equilibrium-relevant polymer in the corresponding leakage analysis.

Our results provide one theoretical and algorithmic answer to the candidate-enumeration problem for equilibrium analysis in large geometry-free molecular systems: Pareto-optimal polymers give a well-justified reduced search space characterized by a Hilbert basis enumeration, and covering-design-based support restriction makes computation feasible in practice.

2 Preliminaries

2.1 Model

Let $\mathbb{Z}, \mathbb{N}, \mathbb{R}$, and $\mathbb{R}_{\geq 0}$ denote the sets of integers, nonnegative integers, real numbers, and nonnegative real numbers, respectively. For a finite set A , we define $\mathbb{N}^A, \mathbb{Z}^A, \mathbb{R}^A$, and $\mathbb{R}_{\geq 0}^A$ as the sets of $|A|$ -dimensional vectors over $\mathbb{N}, \mathbb{Z}, \mathbb{R}$, and $\mathbb{R}_{\geq 0}$, respectively. The vector $\mathbf{x} \in \mathbb{N}^A$ (or $\mathbb{Z}^A, \mathbb{R}^A, \mathbb{R}_{\geq 0}^A$) is represented by $(\mathbf{x}(a))_{a \in A}$, or sometimes $(\mathbf{x}_a)_{a \in A}$ if there is no confusion. A multiset $\mathbf{x}$ over A is an unordered collection of elements allowing duplicates, which is naturally identified by a vector $\mathbf{x}$ in $\mathbb{N}^A$. Similarly, we define $\mathbb{Z}^{A \times B}$ as the set of $|A| \times |B|$ -matrices over $\mathbb{Z}$ for finite sets A, B . The matrix-vector multiplication $M\mathbf{x}$ for $M \in \mathbb{Z}^{A \times B}$ and $\mathbf{x} \in \mathbb{Z}^B$ (or $\mathbb{N}^B, \mathbb{R}^B, \mathbb{R}_{\geq 0}^B$) is defined in a natural way.

We use the following three-level abstraction motivated by DNA nanotechnology and the TBN model [11, 4]. This model captures how monomers (indivisible molecules) are described by domains, and how they combine to form polymers (complexes).

- **Definition 1.** A domain-monomer system is a pair (Σ, M) where
- $\Sigma = D \cup D^*$ is a set of domains described by D and D^* , which are the disjoint and same-sized sets of primal and complementary domains, respectively. For each $a \in D$, there exists a corresponding $a^* \in D^*$ that can form a bond with a , i.e., $D^* = \{a^* : a \in D\}$. We say a and a^* are complementary to each other, and call (a, a^*) the domain pair.
 - M is a finite set of monomers, each of which is a finite multiset of domains over Σ .

Each monomer $m \in M$ is encoded as a vector $\mathbf{a}_m \in \mathbb{Z}^D$, where the entry corresponding to the domain pair $\sigma = (a, a^*)$ is $+1$ for each occurrence of a in m and -1 for a^* . This gives the monomer matrix $A \in \mathbb{Z}^{D \times M}$, whose columns $\mathbf{a}_m$ correspond to each $m \in M$.

Given a domain-monomer system (Σ, M) , a polymer is defined as a finite, nonzero multiset of monomers. We represent each polymer as a vector $\mathbf{p} \in \mathbb{N}^M \setminus \{\mathbf{0}\}$ where $\mathbf{p}(m)$ (or sometimes $\mathbf{p}_m$) denotes the number of copies of monomer m in polymer $\mathbf{p}$. In other words, $P = \mathbb{N}^M \setminus \{\mathbf{0}\}$ defines the set of all possible polymers. The net domain vector of $\mathbf{p}$ is $A\mathbf{p} \in \mathbb{Z}^D$, whose entry for domain a gives the net count of a minus a^* across all monomers in the polymer. We call an equation $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$ a split (of $\mathbf{p}$).

Intrapolymer bonding is assumed throughout: complementary domains within a single polymer are taken to be already bound. Under this assumption, the free domains of a polymer $\mathbf{p}$ are precisely those whose corresponding entry in $A\mathbf{p}$ is nonzero. Note that as in the TBN model, there is no underlying notion of geometry which can prevent two complementary domains within a polymer from binding.

2.2 Computing Polymer Concentrations at Equilibrium

The *concentration* $[\mathbf{p}]$ of polymer $\mathbf{p} \in P$ specifies the amount of polymer $\mathbf{p}$ in the system. In our theoretical results, we use dimensionless mole fractions [10], capturing the amount of the polymer relative to the solvent molecules (water). In normal dilute solutions, $\sum_{\mathbf{p} \in P} [\mathbf{p}] \ll 1$. For each $m \in M$, the polymer concentration induces the total monomer concentration

$$[m] = \sum_{\mathbf{p} \in P} \mathbf{p}_m [\mathbf{p}]. \quad (1)$$

Equilibrium analysis aims to determine the concentrations of polymers at equilibrium, given an initial supply of monomers (or polymers). NUPACK’s `complex_concentrations` function [12]¹ and COFFEE [24] can compute equilibrium concentrations given the initial monomer concentrations and a candidate set of polymers. However, the main computational bottleneck is typically not the equilibrium computation itself, but the construction of an appropriate candidate set of polymers. For a given domain-monomer system, the set of possible polymers is generally infinite and cannot be enumerated exhaustively.

In this paper, we introduce the principled restriction to polymers that are Pareto-optimal. As we show next, Pareto-suboptimal polymers can be excluded on thermodynamic grounds. This reduces the search space from an intractable infinite family to a finite set of candidates, after which additional enumeration strategies can be applied.

For complex systems, even restricting to Pareto-optimal polymers is computationally intractable. This motivates our subsequent methodology of limiting the “size” of the enumerated Pareto-optimal polymers, where size can be defined in two ways: (1) the number of different types of monomers, or (2) the number of types of domain pairs in the polymer.

► **Remark 2.** A domain-monomer system corresponds exactly to a Thermodynamic Binding Network (TBN) [11, 4] under two relaxed assumptions. First, saturation is not assumed: the net domain vector $A\mathbf{x}$ of a polymer $\mathbf{x}$ need not be nonnegative, meaning unstarred domains may remain unbound. Second, no assumption is made that enthalpy dominates entropy. The present work analyzes TBNs from this more general perspective, without either idealization.

3 Pareto-Optimal Polymers

3.1 Defining Pareto-Optimal Polymers

Throughout this paper, we fix the domain-monomer system (Σ, M) unless specified otherwise. In this section, we introduce reasoning that allows us to exclude certain polymers from consideration, based on the observation that they do not occur in minimum free energy configurations. This leads to a *finite* search space instead of the infinite set of all polymers.

► **Definition 3** (Complementary polymers). *Two polymers $\mathbf{p}, \mathbf{q} \in P$ are complementary if there exists a domain such that the entries of $A\mathbf{p}$ and $A\mathbf{q}$ corresponding to that domain are both nonzero and have opposite sign. In other words, $(A\mathbf{p})_a \cdot (A\mathbf{q})_a < 0$ for some $a \in D$.*

¹ Equivalently the `concentrations` command line tool for older NUPACK 3.

Equivalently, $\mathbf{p}$ and $\mathbf{q}$ are complementary if and only if the combined polymer $\mathbf{p} + \mathbf{q}$ contains at least one intrapolymer bond between a domain of $\mathbf{p}$ and a domain of $\mathbf{q}$, so that splitting $\mathbf{p} + \mathbf{q}$ back into $\mathbf{p}$ and $\mathbf{q}$ necessarily breaks at least one bond.

► **Definition 4** (Pareto-(sub)optimal polymer). *A polymer $\mathbf{p} \in P$ is Pareto-suboptimal if there exists a split $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$ with $\mathbf{p}_1, \mathbf{p}_2 \in P$ such that $\mathbf{p}_1$ and $\mathbf{p}_2$ are not complementary. If $\mathbf{p}$ is not Pareto-suboptimal, it is Pareto-optimal. The set of all Pareto-optimal polymers is denoted P^* .*

Figure 1a illustrates a Pareto-suboptimal polymer: its monomers can be regrouped into two non-complementary polymers without breaking any bond. We stress that the model does not track bonds between specific domains. A polymer is represented only by its monomer multiset $\mathbf{p}$; the explicit bonds drawn in the figure are an informal visualization rather than part of the model. From the model's perspective, the reconfiguration changes only how the monomers are partitioned into separate polymers, leaving the bonding unchanged. Note that the geometry-free nature of the model is essential here. Geometric constraints could permit $\mathbf{p}$ but not its decomposition into $\mathbf{p}_1 + \mathbf{p}_2$.

3.2 Thermodynamic Justification

In this section we justify the exclusion of Pareto-suboptimal polymers on thermodynamic grounds, arguing that they are in a certain sense negligible. We begin with the discrete model, showing that no Pareto-suboptimal polymer appears in any minimum free energy configuration. This provides strong motivation for the exclusion, but the gap between discrete minimum free energy configurations and continuous equilibrium concentrations calls for a quantitative bound in the continuous regime. Accordingly, in the second part of this section we derive a general bound on the total concentration of suboptimal polymers in terms of the total concentration of optimal ones.

3.2.1 Discrete thermodynamics

Towards justifying the exclusion of Pareto-suboptimal polymers, in this section we use a simple discrete model to show that any minimum free energy configuration contains only Pareto-optimal ones. Given an initial composition, i.e., a multiset of monomers, a *configuration* is a multiset of polymers built from this composition. The *free energy* of a configuration C is

$$G(C) = a \cdot H(C) - b \cdot S(C)$$

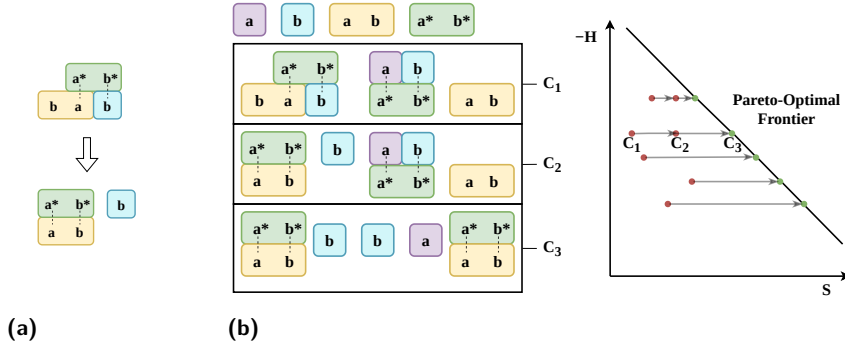
where $H(C) < 0$ is the total enthalpy (sum of bond energies across all bonds in C), $S(C)$ is the entropy (number of separate polymer units), and $a, b > 0$ are system-dependent weights. The probability of observing a configuration C is proportional to its Boltzmann weight:

$$\mathbb{P}[C] \propto e^{-G(C)/kT}$$

► **Lemma 5.** *No Pareto-suboptimal polymer appears in any minimum free energy configuration.*

The proof is deferred to Appendix A.2.

Figure 1b shows an example configuration C_1 where the first polymer is Pareto-suboptimal. Splitting this polymer yields configuration C_2 , which strictly improves on the free energy. Note that a configuration may have no Pareto-suboptimal polymer and still not lie on the Pareto-optimal frontier. For example, the second and third polymers in C_2 can be reconfigured into three polymers, which again strictly improves on the free energy (configuration C_3).



■ **Figure 1** Pareto-suboptimal polymers and the Pareto-optimal frontier. **(a)** Visualization of a Pareto-suboptimal polymer (top) whose monomers can be regrouped into two non-complementary polymers with the same number of bonds (bottom). **(b)** The Pareto-optimal frontier in the $(S, -H)$ plane. Minimum free energy configurations lie on the frontier. Configurations containing Pareto-suboptimal polymers lie strictly below the frontier; each such configuration is dominated by a configuration with the same enthalpy and strictly higher entropy (rightward arrows). While suboptimal polymers necessarily make a configuration suboptimal, the implication does not go the other way. For example, C_2 does not contain a Pareto-suboptimal polymer, but reconfiguring its monomers yields the configuration C_3 with one more polymer but the same bonds.

3.2.2 Continuous thermodynamics

Toward further justifying the exclusion of Pareto-suboptimal polymers, we now turn to the continuous concentrations regime. We show that the total concentration of Pareto-suboptimal polymers is much smaller than the total concentration of Pareto-optimal polymers at equilibrium.

The equilibrium in the continuous model corresponds to the minimizer of the pseudo-Helmholtz free energy

$$g = \sum_{\mathbf{p} \in P} [\mathbf{p}] (\log[\mathbf{p}] + G_{\mathbf{p}} - 1),$$

where $G_{\mathbf{p}} = -\log \Omega_{\mathbf{p}}$ is the formation energy of the polymer $\mathbf{p} \in P$, subject to the monomer conservation constraints Equation (1).

At equilibrium, for any (nontrivial) split $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$ where $\mathbf{p}_1$ and $\mathbf{p}_2$ are not complementary, the following holds for the dimensionless mole fractions in Equation (1):

$$[\mathbf{p}] = [\mathbf{p}_1] \cdot [\mathbf{p}_2]. \quad (2)$$

This equality follows from the first-order optimality conditions of the pseudo-Helmholtz free energy minimization, which yield $[\mathbf{p}] = \Omega_{\mathbf{p}} \prod_{m \in M} z_m^{\mathbf{p}(m)}$ for some $z_m > 0$. The factorization over a non-complementary split then holds because the formation energy is additive across non-interacting parts. This is a standard result in the polymer equilibrium literature [10].

► **Theorem 6.** *Let $[\cdot]$ denote equilibrium concentrations, and define $C_{\text{opt}} := \sum_{\mathbf{p} \in P^*} [\mathbf{p}]$, and $C_{\text{sub}} := \sum_{\mathbf{p} \in P \setminus P^*} [\mathbf{p}]$. If $C_{\text{opt}} < 1$, then*

$$C_{\text{sub}} \leq \frac{C_{\text{opt}}^2}{1 - C_{\text{opt}}}.$$

In particular, in the dilute regime when $C_{\text{opt}} \ll 1$, we have $C_{\text{sub}} = O(C_{\text{opt}}^2)$, so the total concentration of Pareto-suboptimal polymers is negligible compared with the total concentration of Pareto-optimal polymers. We defer the proof to Appendix A.1.

4 Enumeration of Pareto-Optimal Polymers via Hilbert Basis Computation

This section shows that the Pareto-optimal polymers of a domain-monomer system can be identified by a Hilbert basis of a set of neutral polymers having zero net domain vectors in an extended system.

Our Hilbert-basis characterization generalizes the saturated-TBN framework of Haley and Doty [15]. Specifically, while the original TBN model [11, 4] used by the prior work focuses on configurations with maximum binding (called saturated configurations), our domain-monomer system relaxes this assumption and allows unpaired complementary domains in different polymers to coexist in a configuration. In order to make this generalization, we utilize an additional augmented neutralization construction described below.

Given a domain-monomer system (Σ, M) with monomer matrix $A \in \mathbb{Z}^{|D| \times |M|}$, we define an *augmented (domain-monomer) system* by introducing two *unit monomers* per domain pair. More precisely, the augmented system (Σ, M') is a domain-monomer system defined as follows: It inherits the original domain Σ , and M' is defined as a superset of M by

$$M' = M \cup \{u_a = \{a\}, u_{a^*} = \{a^*\} : (a, a^*) \text{ is a domain pair}\}.$$

In words, for each domain pair (a, a^*) , the unit monomer u_a contains only a , and the unit monomer u_{a^*} contains only a^* . Both are added as monomers in the augmented system. Their columns in the augmented monomer matrix are $+1$ and -1 , respectively, in the coordinates corresponding to (a, a^*) and 0 elsewhere. Let M' denote the augmented monomer set and $A' \in \mathbb{Z}^{D \times M'}$ the corresponding augmented monomer matrix. It is worth noting that for $\mathbf{w} = (\mathbf{w}_M, \mathbf{w}_\Sigma) \in \mathbb{Z}^M \times \mathbb{Z}^\Sigma = \mathbb{Z}^{M'}$, it holds that for $a \in D$

$$(A'\mathbf{w})_a = (A\mathbf{w}_M)_a + (\mathbf{w}_\Sigma)_{u_a} - (\mathbf{w}_\Sigma)_{u_{a^*}}. \quad (3)$$

For simplicity, we assume that the original monomer set M does not contain any unit monomer u_a or u_{a^*} so that $|M'| = |M| + |\Sigma|$, which significantly simplifies the details of this section. However, our results continue to hold without this assumption with minor modification of the argument.²

A polymer $\mathbf{x} \in \mathbb{Z}^M$ can be *neutralized* by attaching the unit monomers to unbound domains. More precisely, we define the neutralizing map $\nu : \mathbb{Z}^M \rightarrow \mathbb{Z}^{M'}$ as follows.

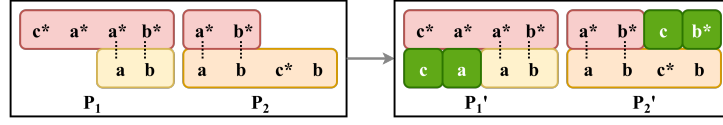
$$(\nu(\mathbf{x}))_m = \begin{cases} \mathbf{x}_m & \text{for } m \in M \\ \max(-(A\mathbf{x})_a, 0) & \text{for } m = u_a \\ \max((A\mathbf{x})_a, 0) & \text{for } m = u_{a^*} \end{cases}$$

In words, for each pair (a, a^*) , we add c copies of u_{a^*} if $\mathbf{x}$ has $c > 0$ unbound domain a , and add c copies of u_a if $\mathbf{x}$ has $c > 0$ unbound a^* . Figure 2 visualizes the neutralizing map.

We give the algebraic interpretation. For each $a \in D$, let $e_a \in \mathbb{Z}^D$ denote the unit vector with a 1 in the coordinate corresponding to a and 0 in all other coordinates. Observe that the neutralizing map indeed makes the net domain vector zero:

$$A' \cdot \nu(\mathbf{x}) = A\mathbf{x} + \left(\sum_{a \in D} \max(-(A\mathbf{x})_a, 0) \cdot e_a \right) + \left(\sum_{a \in D} \max((A\mathbf{x})_a, 0) \cdot (-e_a) \right) = \mathbf{0}.$$

² For example, we can add “dummy” domains to the singleton monomers in the original system to enforce the assumption.



■ **Figure 2** The polymers in the first configuration have been neutralized by unit monomers so that each vector representation has zero net domain vector.

This follows from Equation (3) coordinate-wise for each $a \in D$. This means that all the domains are bound, justifying the name of the neutralizing map ν . In general, we define the set of *neutral* polymers³ in the augmented system:

$$\mathcal{C}(A') = \{ \mathbf{z} \in \mathbb{N}^{M'} \mid A' \mathbf{z} = \mathbf{0} \}.$$

Each $\mathbf{z} \in \mathcal{C}(A')$ has a zero net domain vector. The set of neutral polymers has a clean algebraic representation, allowing us to relate the Pareto-optimal polymers in the system with the “indecomposable” elements, i.e., the Hilbert basis elements as we will see later.

Given a neutral polymer in the augmented system, we can recover a polymer in the original system using a projection map $\pi : \mathbb{Z}^{M'} = \mathbb{Z}^M \times \mathbb{Z}^\Sigma \rightarrow \mathbb{Z}^M$ that discards the unit monomer coordinates. In particular, $\pi(\nu(\mathbf{x})) = \mathbf{x}$ holds for all polymers $\mathbf{x} \in \mathbb{Z}^M$.

It is clear that $(\nu(\mathbf{x}))_{u_a} \cdot (\nu(\mathbf{x}))_{u_{a^*}} = 0$ for all $a \in D$. We call a polymer $\mathbf{w} \in \mathbb{Z}^{M'}$ *dichotomous* if it satisfies this property: $(\mathbf{w})_{u_a} \cdot (\mathbf{w})_{u_{a^*}} = 0$ for all $a \in D$. The following lemma is not hard to show. Intuitively, among all neutral augmented polymers that project to a fixed original polymer $\mathbf{x}$, there is exactly one with no redundant canceling pair of unit monomers.

► **Lemma 7.** $\nu(\mathbf{x})$ is the unique dichotomous neutral polymer in $\pi^{-1}(\mathbf{x}) \cap \mathcal{C}(A')$.

4.1 Hilbert Basis

We briefly recall the definitions and basic properties of Hilbert bases that we need. First, for an integer matrix $B \in \mathbb{Z}^{m \times n}$, the set $C = \{ \mathbf{x} \in \mathbb{R}^n : B\mathbf{x} \geq \mathbf{0} \}$ defined by a system of inequalities is called a (rational polyhedral) cone. If there exists a $\mathbf{c} \in \mathbb{R}^n$ such that $\mathbf{c} \cdot \mathbf{x} > 0$ for all $\mathbf{x} \in C \setminus \{ \mathbf{0} \}$, we say C is a pointed cone.

The following definition of the Hilbert basis will be convenient for this paper.

► **Definition 8** (Hilbert basis). The Hilbert basis $H(C)$ of a pointed cone C is a subset of $C \cap \mathbb{Z}^n \setminus \{ \mathbf{0} \}$ defined as follows:

$$H(C) = \{ \mathbf{h} \in C \cap \mathbb{Z}^n \setminus \{ \mathbf{0} \} \mid \mathbf{h} \text{ is not the sum of two vectors in } C \cap \mathbb{Z}^n \setminus \{ \mathbf{0} \} \}.$$

It is known that the Hilbert basis of any pointed (rational polyhedral) cone is finite. There are several equivalent definitions of the Hilbert basis. Notably, it is the smallest set that generates $C \cap \mathbb{Z}^n$ by nonnegative integer combinations; this characterization was previously used to enumerate the minimal set of reactions or polymers [2, 15].⁴ The computation of the Hilbert basis has been extensively studied in the literature [6, 16, 18], and implementations are publicly available [1, 5].

³ In other words, the saturated polymers. We stress that we do not assume that all polymers are saturated; we consider this set only in the augmented system for deriving the computational tools.

⁴ Technically, our definition coincides with this usual definition only for the *pointed* cone.

4.2 Pareto-Optimal Polymers as Hilbert Basis

The set of neutral polymers $\mathcal{C}(A')$ is an intersection of $\mathbb{Z}^{M'}$ and a cone

$$\mathcal{C}_{\mathbb{R}}(A') = \{\mathbf{x} \in \mathbb{R}^{M'} \mid A'\mathbf{x} \geq \mathbf{0}, -A'\mathbf{x} \geq \mathbf{0}, I\mathbf{x} = \mathbf{x} \geq \mathbf{0}\} \quad (4)$$

which is pointed because $\mathbf{1} \cdot \mathbf{x} > 0$ holds for all nonzero $\mathbf{x} \in \mathcal{C}_{\mathbb{R}}(A')$. We denote $H = H(\mathcal{C}_{\mathbb{R}}(A'))$ the Hilbert basis of $\mathcal{C}_{\mathbb{R}}(A')$, which is a subset of $\mathcal{C}(A') = \mathcal{C}_{\mathbb{R}}(A') \cap \mathbb{Z}^{M'}$ by definition.

Recall P^* is the set of Pareto-optimal polymers in the original system (Σ, M) . The following theorem states that the Pareto-optimal polymers are characterized by the projection of Hilbert basis elements $\pi(H) := \{\pi(\mathbf{h}) \mid \mathbf{h} \in H\}$.

► **Theorem 9.** $P^* = \pi(H) \setminus \{\mathbf{0}\}$.

The proof is deferred to Appendix A.3. A few implications are in order. First, this shows that P^* is finite because of the finiteness of the Hilbert basis. Furthermore, we can, in principle, compute the set P^* using the well-established Hilbert basis algorithms. We discuss the limitations of using this theorem directly in Appendix B.

5 Scalable Enumeration with Bounded Support

We present scalable algorithms to compute a meaningful subset of Pareto-optimal polymers. To scale the enumeration of Pareto-optimal polymers, we consider the subset of polymers with a bounded number of monomer types. We define the *support* of polymer $\mathbf{p} \in \mathbb{N}^M$ by

$$\text{supp}(\mathbf{p}) = \{m \in M \mid \mathbf{p}_m > 0\},$$

and for a user-chosen parameter t , the *t-bounded Pareto-optimal set* by

$$P_t^* = \{\mathbf{x} \in P^* \mid |\text{supp}(\mathbf{x})| \leq t\}.$$

We also consider a similarly restricted subset of P^* based on the number of domain pairs instead of monomers. This introduces the domain-support mode, which proved ineffective in our numerical experiments. One reason is that many polymers have a larger number of domain pairs than distinct monomers. In other words, the domain mode may be advantageous when the number of domain pairs is smaller than the number of monomers, or when many monomers share the same domain pairs. We outline this approach in Appendix C.

In Section 5.1, we show how to compute the t -bounded Pareto-optimal set through the computations of smaller Hilbert bases. However, the number of Hilbert bases to compute is large, making the naive algorithm less appealing. Instead, we introduce our main algorithm in Section 5.2, which optimizes the overall time complexity by reducing the number of Hilbert basis computations using covering designs. In numerical experiments of Section 7, we will see that the equilibrium concentration analysis with the subset P_t^* is almost as good as the one with the full set P^* for many applications.

5.1 Enumeration with All t -sized Supports

We introduce a few pieces of mathematical notation that will streamline the arguments to follow. Let $S \subseteq M$ be a subset of monomers with $|S| = t$. The set (Σ, S) gives a natural domain-monomer (sub)system. Define an *embedding* $\iota_S : \mathbb{N}^S \rightarrow \mathbb{N}^M$ that appends 0 on each entry corresponding to $M \setminus S$. After embedding, the set of polymers becomes

$$\text{Supp}(S) = \{\mathbf{p} \in \mathbb{N}^M : \text{supp}(\mathbf{p}) \subseteq S\}.$$

12:10 Scalable Enumeration of Pareto-Optimal Polymers

It is not hard to see that if $\mathbf{p} \in \text{Supp}(S)$ admits a split $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$ with nonzero $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{N}^M$, then $\mathbf{p}_1, \mathbf{p}_2 \in \text{Supp}(S)$ as well, since the non-negativity of entries in $\mathbf{p}$ forces $\text{supp}(\mathbf{p}_1), \text{supp}(\mathbf{p}_2) \subseteq \text{supp}(\mathbf{p}) \subseteq S$.

Define $P_S^* = P^* \cap \text{Supp}(S)$ as the set of Pareto-optimal polymers whose support is contained in S . The following shows the connection between the Pareto-optimal polymers in the full system (Σ, M) and subsystem (Σ, S) . Intuitively, Pareto-optimality of the polymers supported only on S can be checked entirely inside the subsystem (Σ, S) , as any split of such a polymer is automatically also supported on S . We defer the formal proof to Appendix A.4.

► **Lemma 10.** *P_S^* is the set of Pareto-optimal polymers of the system (Σ, S) after embedding.*

In particular, this lemma implies the following inclusion:

$$P_S^* \subseteq P_T^* \text{ if } S \subseteq T. \quad (5)$$

Given this, we compute a subset of Pareto-optimal polymers, namely P_S^* , by computing a much smaller Hilbert basis. Formally, we define the set of neutral polymers restricted to S by

$$c^S(A') = \{ \mathbf{x} \in \mathbb{N}^{M'} \mid A' \mathbf{x} = \mathbf{0}, \text{supp}(\pi(\mathbf{x})) \subseteq S \},$$

and the corresponding pointed cone by

$$c_{\mathbb{R}}^S(A') = \{ \mathbf{x} \in \mathbb{R}^{M'} \mid A' \mathbf{x} = \mathbf{0}, \mathbf{x} \geq \mathbf{0}, \mathbf{x}_m = 0 \text{ for } m \in M \setminus S \} \quad (6)$$

following Equation (4). We denote $H_S = H(c_{\mathbb{R}}^S(A'))$ the Hilbert basis of $c_{\mathbb{R}}^S(A')$. Applying Theorem 9 on the system (Σ, S) , we have the following Hilbert basis characterization.

► **Lemma 11.** *$P_S^* = \pi(H_S) \setminus \{\mathbf{0}\}$. In particular, $P_t^* = \bigcup_{\substack{S \subseteq M \\ |S|=t}} (\pi(H_S) \setminus \{\mathbf{0}\})$.*

The proof is deferred to Appendix A.4. This lemma says that each support-restricted Pareto-optimal set can be computed by applying the Hilbert-basis characterization to the subsystem on S , and the t -bounded set is obtained by taking the union over these.

For $|S| \ll |M|$, computing a single Hilbert basis H_S can be done much more efficiently than computing the full Hilbert basis H . The effective ambient space of the pointed cone in Equation (6) is much smaller than that of the original cone in Equation (4), since we can ignore the always-zero original-monomer coordinates for $m \in M \setminus S$. More precisely, the dimensions are $|S| + |\Sigma|$ and $|M| + |\Sigma|$, respectively. Some domains may not appear in the monomers in S , so the dimension can be further reduced. Let D_S be the set of domains that appear in the monomers in S . Let $M'_S = S \cup \{u_a, u_{a^*} : a \in D_S\}$. We also reduce the number of equations defining the cone by considering $A'_S \in \mathbb{Z}^{D_S \times M'_S}$, obtained by removing irrelevant rows and columns. This significantly reduces the number of input constraints in computing H_S .

By Lemma 11, we can compute the t -bounded Pareto-optimal set P_t^* by multiple smaller Hilbert bases H_S for $|S| = t$. This naive subset enumeration algorithm requires one invocation of Hilbert basis computation per t -subset of M , giving a total of $\binom{n}{t}$ invocations. The factor $\binom{n}{t}$ grows rapidly with n and t , making naive enumeration infeasible for moderate n and t . In the next subsection, we explore a better strategy for selecting which subsets to run Hilbert basis computation on in order to reduce the total time.

5.2 Enumeration with Covering Design

The starting point of this section is Equation (5). This inclusion implies that if we compute a Hilbert basis H_T for some T , then we learn *all* H_S for all $S \subset T$. Instead of computing all H_S for $|S| = t$, we compute a small number of H_T with $|T| = k$ for some $k > t$. To minimize the complexity, we want to minimize the number of different T 's.

The *minimum covering design problem* addresses this problem. This problem asks for a collection $\mathcal{B}$ of k -element subsets (blocks) of M of minimum cardinality such that every t -element subset of M is contained in at least one block. More formally:

► **Definition 12** (Covering design). *Let M be a finite set of size n . A (n, k, t) -covering design $\mathcal{D}$ is a collection of k -element subsets of M such that for every t -element subset $T \subseteq M$, there exists a block $B \in \mathcal{D}$ with $T \subseteq B$. The covering number $c(n, k, t)$ denotes the minimum cardinality of any such collection.*

Let $\mathcal{D}$ be a $(|M|, k, t)$ -covering design. Given $\mathcal{D}$, the *covering design strategy* runs the Hilbert basis algorithm on each block $B \in \mathcal{D}$ and computes

$$\hat{P}_{k,t} = \bigcup_{B \in \mathcal{D}} (\pi(H_B) \setminus \{\mathbf{0}\}).$$

The following theorem shows that the covering design strategy computes (a superset of) the t -bounded Pareto-optimal set. The proof can be found in Appendix A.4.

► **Theorem 13.** $P_t^* \subseteq \hat{P}_{k,t} \subseteq P^*$.

In the covering-design strategy, using a covering design $\mathcal{D}$ requires $|\mathcal{D}|$ Hilbert basis computations. The Schönheim lower bound [19] gives

$$c(n, k, t) \geq \left\lceil \frac{n}{k} \left\lceil \frac{n-1}{k-1} \cdots \left\lceil \frac{n-t+1}{k-t+1} \right\rceil \cdots \right\rceil \right\rceil.$$

In practice, covering designs are vastly smaller than the family of all k -subsets, whose cardinality is $\binom{n}{k}$. This reduction makes the covering-design strategy substantially more attractive. However, computing the minimal covering design or computing $c(n, k, t)$ is known to be hard [9]. We use precomputed near-optimal covering designs, or greedy constructions if these are unavailable. Further details are described in the next section.

6 Implementation

All code is available at github.com/architrahul/Pareto-polymer-enumerator. We leverage the modern Hilbert basis algorithm `Normaliz` [5].

For a fixed size of the monomer set n and Pareto-optimal polymer support-bound t , the total runtime of the covering design strategy for a given k is determined by the number of `Normaliz` invocations $|\mathcal{D}_{n,k,t}|$ and the per-invocation cost, which depends on the specific sub-system. The user may specify a particular value of k directly, or use a built-in adaptive probing heuristic to select it. The probing heuristic exploits the fact that the average runtime of a `Normaliz` call for a fixed block size k can be estimated from a relatively small number of sampled blocks. Starting with small k , our tool runs `Normaliz` on a sample of blocks and uses the resulting average per-block runtime to extrapolate the total runtime, stopping when the runtime appears to be the minimum. Since the probing step uses only a small number of sampled blocks, its runtime is negligible compared with the full enumeration time of the experiment.

The details of the construction of covering designs and the block-size selection heuristic are described in Section 1 of the supplementary material.

7 Benchmarking

In this section, we present benchmarks of our new algorithm by applying the covering design strategy on several families of domain-monomer systems drawn from the thermodynamic binding network and DNA molecular programming literature [22, 23, 21]. Our analysis shows that the approach described in Section 6 can analyze large systems that could not be handled by previous TBN analysis methods or by the naive approach from Theorem 9.

We design the experiments to address two key aspects of the covering design approach, namely *scalability* and *equilibrium accuracy*. Our experiments show that, for many applications, the covering design strategy with an appropriate value of t and our heuristic choice of k quickly predicts the equilibrium concentration of the system. For example, for the AND gate linear cascade system [22] with 7 modules, the covering design approach with $(t, k) = (5, 25)$ computes the set $\hat{P}_{25,5}$ in 23.79s while the full Pareto-optimal set is computed in 1105.05s (Figure 3), achieving a $46\times$ speedup. The concentration analysis with COFFEE shows that the covering design approach does not miss any relevant polymer, as shown in Figure 5b.

In our experiments, all Hilbert basis computations were performed using `Normaliz` 3.11.1. The Hilbert basis computations dominate the overall time complexity; the other steps, including the heuristic choice of k , took relatively little time. Equilibrium concentrations were computed with COFFEE [24]. For the AND gate linear cascade experiments, we use two concentration regimes adapted from Figure S6 of the supplementary materials of Wang et al. [22]. In both regimes, the upstream inputs $X_2, \dots, X_{m+1}$ are supplied at 10 nM. Regime A sets all non-input monomers to 100 nM. Regime B uses the same concentrations as Regime A, but additionally supplies 50 nM of all monomers except the output of the final module. For each regime, we consider both a correct-output condition, where X_1 is supplied at 10 nM, and a leakage condition, where $X_1 = 0$.⁵ Each complementary domain bond is assigned energy either -20 kcal/mol or -10 kcal/mol.⁶ The value -20 kcal/mol represents a strong-binding regime comparable to the saturated regime used by Wang et al. [22], while -10 kcal/mol gives a weaker-binding system in which unsaturated polymers are more pronounced.⁷ All experiments were conducted on a MacBook Pro (Apple M3 Max, 16 cores, 48 GB RAM).

7.1 TBN Families

To perform numerical experiments and benchmarking, we focused on several families of TBNs. By “family,” we mean a collection of related TBNs in which the same motif is repeated arbitrarily many times. This parametrization lets us scale these systems to arbitrarily large sizes, increasing the difficulty of the problem.

Linear cascade of AND-gate modules

The AND-gate module was introduced in prior works [11, 4] and recently realized experimentally as a reversible signal propagation module [22]. The linear cascade system [22] (refer to Figure S3 of its supplementary materials for a diagram) consists of m sequentially

⁵ The theoretical model uses dimensionless mole fractions, but the COFFEE input concentrations are specified in molar units. In the dilute-solution regime considered here, mole fractions and molarities differ by a (temperature-dependent) constant conversion factor. COFFEE converts between molarity and mole fractions internally.

⁶ The total energy of a polymer is computed by summing this bond-energy contribution over all complementary domain bonds present in the polymer.

⁷ Although we used uniform bond energies across all domains, our framework does not require this; different domains can have different bond energies.

chained AND-gate modules. Each module is a collection of monomers that are normally bonded with each other in a particular way and, in the presence of two input polymers, can rearrange to release an output polymer. The output of each module serves as one of the two inputs to the next. Ideally, a module’s output forms only when both of its inputs are present, implementing an AND-gate computation. However, unintended reaction pathways cause outputs to form even in the absence of the required inputs, or produce other spurious polymers, a phenomenon known as *leakage*. This system is referred to as the AND gate linear cascade system in this paper.

Binary tree of AND-gate modules

The binary tree system [22] arranges AND-gate modules in a complete binary tree of depth d . Each module operates identically to those in the linear cascade, but here both inputs to every non-leaf module are outputs of two child modules, rather than a mix of primary inputs and upstream outputs. The number of monomers grows exponentially in d . This system is referred to as the AND gate binary tree system in this paper.

Scaffolded DNA system

Sterin et al. [21] introduced “scaffolded DNA computation,” which obtains the desired output at thermodynamic equilibrium. This system differs from the AND module motif in that large polymers are expected: the system consists of a large “scaffold” with short monomers bound to it, their number proportional to the length of the scaffold.

2-reactant, 2-product reversible signal propagation module (2-2 modules) system

This module from [23] behaves similarly to the AND-gate module in the linear cascade, with the difference that each module takes two input monomers and produces two output monomers. For the benchmarking system, both outputs of a module are used as inputs to the next module in the cascade. It will be referred to as the 2-2 linear cascade in this paper.

7.2 Runtime Analysis

Figures 3 and 4 show the (logarithmic) runtime of our algorithm for the AND gate linear cascade, binary tree system, and 2-2 linear cascade. We vary the system sizes by scaling the depth or the number of modules. In each system, we run the algorithm for various t , with the best k chosen by probing (described in section 1 of supplementary text).

Across all benchmark families, the covering design strategy provides substantial speedups of several orders of magnitude over full-system Hilbert basis computation (FULL HB), which is executed whenever feasible. The speedup is more drastic for larger systems, while for smaller systems, the covering design strategy with large t shows slower performance. The Hilbert basis computation dominates the overall runtime for all experiments.

7.3 Equilibrium Analysis Accuracy

We compute the equilibrium concentrations using COFFEE [24], given the set of polymers computed using either the covering design approach ($\hat{P}_{k,t}$) or the full Pareto-optimal set (P^*). For the systems considered in this section, the numerical experiments below show that $t = 5$ recovers almost all relevant polymers in equilibrium.

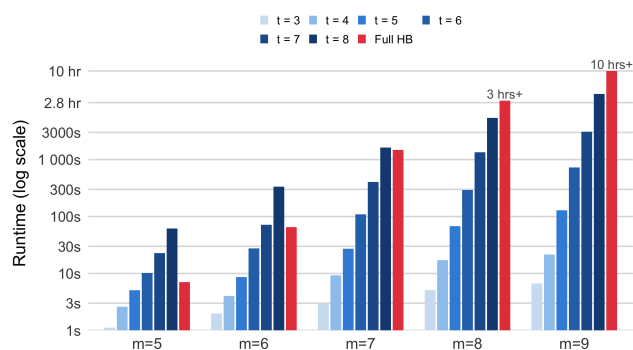


Figure 3 Pareto-optimal polymer set computation time for AND gate linear cascade systems with $m = 5-9$ modules. Within each group, bars indicate covering design $t = 3-8$ and full Hilbert basis computation (FULL HB). Bars labeled *hrs+* denote terminated runs, truncated at the cutoff.

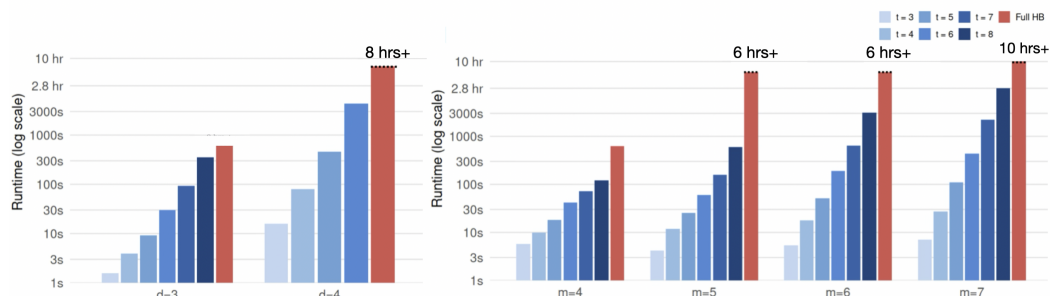


Figure 4 Pareto-optimal polymer set computation time for AND gate binary tree (left) and 2-2 linear cascade (right) TBN systems. Binary tree: depth $d = 3, 4$, with $t = 7, 8$ and full HB truncated for $d = 4$. 2-2 linear cascade system: $m = 4-7$, full HB truncated at 6 hours for $m = 5, 6$ and 10 hours for $m = 7$.

We focus on cases where computing the full Pareto-optimal set P^* is feasible to allow comparison with results on $\hat{P}_{k,t}$. In particular, we consider the AND gate linear cascade with 7 modules, the AND gate binary tree with depth 3, and the 2-2 linear cascade with 4 modules. The COFFEE analysis with the reduced candidates $\hat{P}_{k,t}$ for $t = 4, \dots, 7$ recovered all significant⁸ polymers from the full set, showing that the support-bounded restriction does not distort the analysis for these systems.

We also consider a seven-module AND gate linear cascade under the Regime B leakage condition with bond energy $E = -20$ kcal/mol. Here, the effect of the support bound t becomes visible, as shown in Figure 5. For $t = 3$, the reduced candidate set misses some equilibrium-relevant polymers and produces significant relative concentration errors for some recovered polymers. In contrast, for $t = 5$, all equilibrium-relevant polymers are recovered, and the relative concentration errors are comparatively very low.⁹

⁸ Polymers were categorized as significant if their equilibrium concentration exceeded 0.1 nM and insignificant otherwise.

⁹ Note that even for $t = 3$, only the lower-concentration polymers are missed and the overall equilibrium analysis is still fairly accurate.

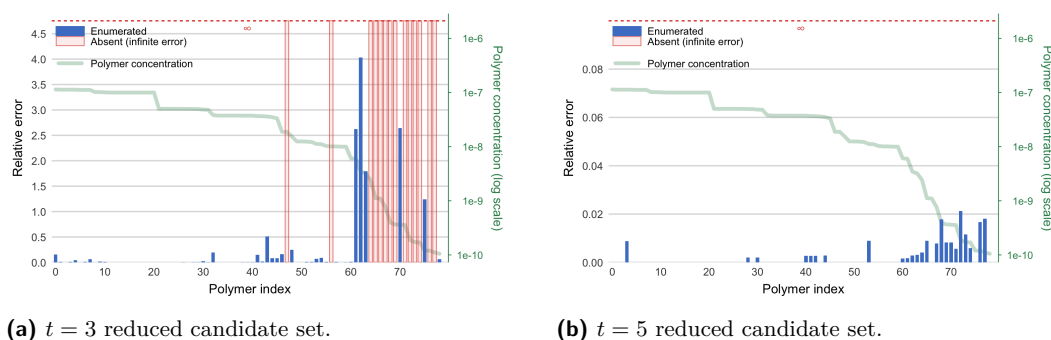


Figure 5 Relative concentration error for equilibrium-relevant polymers in the seven-module AND gate linear cascade under the Regime B leakage condition with bond energy $E = -20$ kcal/mol. Red lines indicate polymers absent from the reduced candidate set, corresponding to infinite relative error; blue lines indicate recovered polymers; the green line shows the polymer concentrations.

7.4 Leakage Analysis Applications

Beyond equilibrium analysis, the reduced candidate sets can be used for downstream applications as well. One important application is leakage analysis. In engineered TBN systems, leakage refers to the formation of an unintended product through spurious reaction pathways. We focus on the output leakage for the AND gate linear cascade family here, which is the final output concentration when any one of the inputs is removed.¹⁰ The output leakage analysis of this system with more modules is out of reach for the previous analytic tools.¹¹

We consider two leakage experiments. The first experiment compares how much leakage occurs depending on where the input is removed. We fix a seven-module AND gate linear cascade and remove one input at a time, using Regime B (introduced earlier in Section 7) with bond energy $E = -10$ kcal/mol. For each removed input, the equilibrium concentration of the final output polymer is computed using COFFEE on the candidate set produced by our covering-design enumeration pipeline. As shown in Section 2 of the supplementary materials, removing an input from the first module gives the largest leakage, and the leakage decreases as the removed input moves later in the cascade. This suggests that leakage generated earlier in the cascade can propagate through subsequent modules, whereas leakage introduced closer to the end has fewer downstream opportunities to amplify.

The second experiment studies how leakage changes as the cascade becomes longer. We remove an input to the first module and vary the number of modules in the linear cascade. For each setting, we compare the leakage output concentration against the correct output concentration obtained when all inputs are present.

We run this experiment under both concentration regimes (A and B) and both bond-energy values (-10 and -20 kcal/mol). Regime A gives the direct leakage setting, but the final output concentration reaches its plateau quickly. Regime B suppresses leakage and makes the dependence on cascade length more visible. Similarly, $E = -20$ kcal/mol mirrors the strong-binding regime used in Wang et al. [22] (the model used in [22], Section S4.2, restricts to saturated polymers only), while $E = -10$ kcal/mol tests a weaker-binding regime where unsaturated polymers play a larger role. Thus, the four cases in Figure 6 compare how leakage scales across both concentration and binding regimes.

¹⁰The mechanism of an AND gate linear-cascade module is illustrated in Figure 7; a full four-module example is given in Appendix D.1.

¹¹We note that the numerical analysis of [22] (Section S4.2) restricts to the fully bound (saturated) regime, and even with that simplification only extends up to 10 modules.

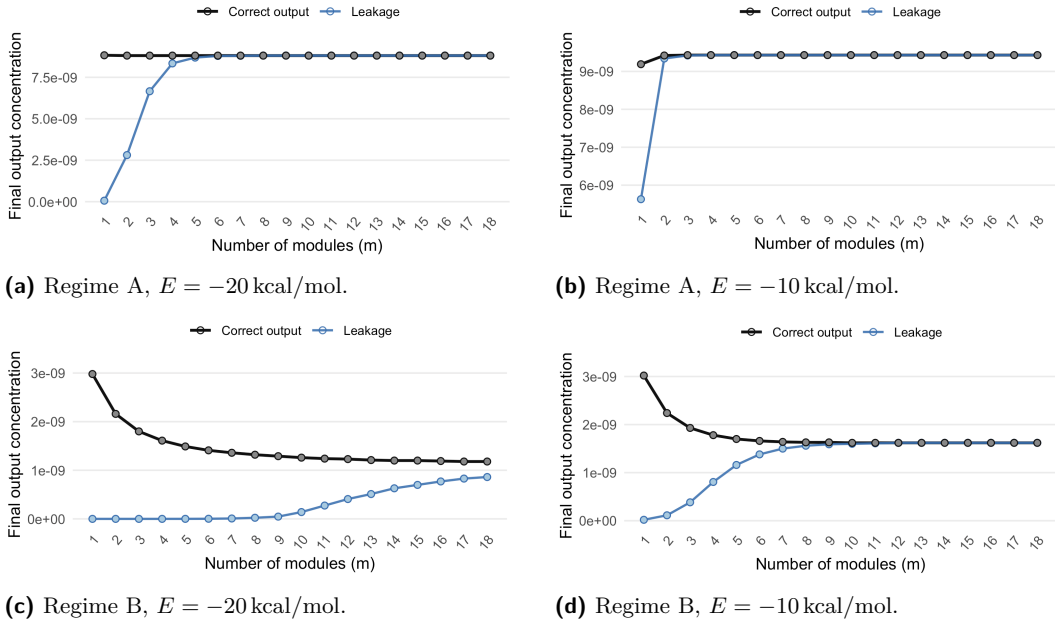


Figure 6 Leakage versus correct-output concentration for AND gate linear cascade systems across cascade length. The four panels show the two concentration regimes, A and B, crossed with bond energies $E = -20$ kcal/mol and $E = -10$ kcal/mol. For leakage, an input to the first module is removed; for correct output, all inputs are present. All curves are computed with COFFEE using $\hat{P}_{k,t}$, with $t = 5$ when feasible and $t = 4$ otherwise.

As shown in Figure 6, the final output concentration increases with cascade length. Regime A reaches a plateau quickly, whereas Regime B has comparatively less leakage. Weakening the bond energy from -20 kcal/mol to -10 kcal/mol increases leakage as expected.

Although the runtime still grows roughly exponentially with the system size, the covering-design approach remains practical for systems beyond the range of full Hilbert basis enumeration. When larger systems become intractable, the support parameter t can be decreased to further reduce the enumeration cost. This sacrifices some accuracy in the equilibrium analysis because fewer candidate polymers are included, but the earlier experiments show that small values of t can still recover the dominant equilibrium behavior in the tested systems. The main additional limitation is the size of the covering design: the dynamic-programming construction used for large parameters does not guarantee a minimum-size covering design, and may not scale optimally for very large $|M|$. Nevertheless, in our experiments, AND gate linear cascade systems with up to eighteen modules were analyzed efficiently enough to support leakage studies that would be infeasible with direct full Hilbert basis computation.

8 Conclusion

We introduced a framework for restricting polymer enumeration in geometry-free domain-monomer systems to the set of *Pareto-optimal polymers* for the purpose of equilibrium analysis. The Hilbert basis analysis provides a principled way to reduce the infinitely many possible polymers to finitely many that dominate the equilibrium. Further, because our method permits restricting the number of monomer types in allowed polymers, our algorithm can scale to larger systems for which enumerating all Pareto-optimal polymers would be impractical. Benchmarking our analysis on AND gate linear cascades, binary trees, and 2-2 linear cascade families from [22, 23, 21] shows the efficiency and accuracy of the algorithm for appropriate parameters.

The scalability of our algorithm is based on focusing on polymers with at most t distinct monomer types. This gives users a practical knob for trading completeness against runtime, tied to a design-level quantity rather than to an internal feature of the Hilbert-basis computation. Our experiments show that in many applications, modest values of t suffice to accurately predict the equilibrium behavior. However, when working with systems in which very large polymers are expected (as with the scaffolded DNA computer [21]), this support-bounded analysis is less effective.

Despite the greater efficiency afforded by our algorithm, our approach inevitably runs into limits for large t . One area for optimization is pre-computation of covering designs for larger system sizes. Recall that when a precomputed covering design is unavailable [13], we use the dynamic-programming construction of Gordon et al. [14]. As described in Section 1 of the supplementary material, this construction does not necessarily produce an optimal (smallest possible) covering design. Nonetheless, even with optimizations, the Hilbert basis algorithm becomes quickly impractical as the system size grows. The broader open question is whether, beyond the Pareto-optimality criterion introduced here, there are other structural properties – perhaps exploiting additional knowledge of a given system – that could further reduce the candidate set while preserving all equilibrium-relevant polymers.

References

- 1 4ti2 Team. 4ti2—a software package for algebraic, geometric and combinatorial problems on linear spaces. URL: <https://4ti2.github.io>.
- 2 Hamidreza Akef, Minki Hhan, and David Soloveichik. Computing and bounding equilibrium concentrations in athermic chemical systems. In *31st International Conference on DNA Computing and Molecular Programming (DNA 31)*, pages 10–1. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.DNA.31.10.
- 3 Keenan Breik, Cameron Chalk, David Doty, David Haley, and David Soloveichik. Programming substrate-independent kinetic barriers with thermodynamic binding networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 18(1):283–295, 2021. doi:10.1109/TCBB.2019.2959310.
- 4 Keenan Breik, Chris Thachuk, Marijn Heule, and David Soloveichik. Computing properties of stable configurations of thermodynamic binding networks. *Theoretical Computer Science*, 785:17–29, 2019. doi:10.1016/J.TCS.2018.10.027.
- 5 W. Bruns, B. Ichim, C. Söger, and U. von der Ohe. Normaliz: Algorithms for rational cones and affine monoids. <https://www.normaliz.uni-osnabrueck.de>.
- 6 Winfried Bruns and Bogdan Ichim. Normaliz: Algorithms for affine monoids and rational cones. *Journal of Algebra*, 324(5):1098–1113, 2010.
- 7 Kevin M Cherry and Lulu Qian. Supervised learning in DNA neural networks. *Nature*, 645(8081):639–647, 2025.
- 8 Evelyne Contejean and Hervé Devie. An efficient incremental algorithm for solving systems of linear diophantine equations. *Information and Computation*, 113(1):143–172, 1994. doi:10.1006/INCO.1994.1067.
- 9 Pilu Crescenzi, Federico Montecalvo, and Gianluca Rossi. Optimal covering designs: Complexity results and new bounds. *Discrete Applied Mathematics*, 144(3):281–290, 2004. doi:10.1016/J.DAM.2003.11.006.
- 10 Robert M Dirks, Justin S Bois, Joseph M Schaeffer, Erik Winfree, and Niles A Pierce. Thermodynamic analysis of interacting nucleic acid strands. *SIAM Review*, 49(1):65–88, 2007. doi:10.1137/060651100.
- 11 David Doty, Trent A Rogers, David Soloveichik, Chris Thachuk, and Damien Woods. Thermodynamic binding networks. In *International Conference on DNA-Based Computers*, pages 249–266. Springer, 2017. doi:10.1007/978-3-319-66799-7_16.

- 12 Mark E Fornace, Jining Huang, Cody T Newman, Avinash Nanjundiah, Nicholas J Porubsky, Marshall B Pierce, and Niles A Pierce. NUPACK: Analysis and design of nucleic acid structures, devices, and systems. *ChemRxiv*, 2025.
- 13 Daniel Gordon. La jolla coverings repository, April 2026. doi:10.5281/zenodo.19735294.
- 14 Daniel M Gordon, Oren Patashnik, and Greg Kuperberg. New constructions for covering designs. *Journal of Combinatorial Designs*, 3(4):269–284, 1995.
- 15 David Haley and David Doty. Computing properties of thermodynamic binding networks: An integer programming approach. In Matthew R. Lakin and Petr Šulc, editors, *27th International Conference on DNA Computing and Molecular Programming (DNA 27)*, volume 205 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:16, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.27.2.
- 16 Raymond Hemmecke. On the computation of hilbert bases of cones. In *Mathematical Software: Proceedings of the First International Congress of Mathematical Software*, page 307. World Scientific, 2002.
- 17 Miki Hermann, Laurent Juban, and Phokion G Kolaitis. On the complexity of counting the hilbert basis of a linear diophantine system. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning*, pages 13–32. Springer, 1999. doi:10.1007/3-540-48242-3_2.
- 18 Loic Pottier. The euclidean algorithm in dimension n. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation*, pages 40–42, 1996. doi:10.1145/236869.236894.
- 19 Johanan Schonheim. On coverings. *Pacific Journal of Mathematics*, 14(4):1405–1411, 1964.
- 20 Tianqi Song and Lulu Qian. Heat-rechargeable computation in DNA logic circuits and neural networks. *Nature*, 646(8084):315–322, 2025.
- 21 Tristan Stérin, Abeer Eshra, Janet Adio, Constantine Glen Evans, and Damien Woods. A thermodynamically favoured molecular computer: Robust, fast, renewable, scalable. *bioRxiv*, 2025.
- 22 Boya Wang, Cameron Chalk, David Doty, and David Soloveichik. Molecular computation at equilibrium via programmable entropy. *Science Advances*, 12:eadx3969, 2026. doi:10.1126/sciadv.adx3969.
- 23 And Kaan Ata Yilmaz and David Soloveichik. Modularity of signal propagation networks in the thermodynamic binding network model. In *32nd International Conference on DNA Computing and Molecular Programming (DNA32)*, 2026.
- 24 Fu-Yao Yu, Sahil Shah, Yash Mittal, Paul Bessler, Aamir Mohsin, Jeffrey Geng, Arnav Vats, and David Soloveichik. COFFEE: A high-performance approach to convex optimization for thermodynamic equilibrium computations. In *2025 Systems and Information Engineering Design Symposium (SIEDS)*, pages 161–166. IEEE, 2025.

A

 Deferred Proofs

A.1 Proof of Theorem 6

Proof. Let $\mathbf{p} \in P \setminus P^*$ be a Pareto-suboptimal polymer. By definition, there exists a (nontrivial) split $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$ with $\mathbf{p}_1$ and $\mathbf{p}_2$ not complementary. If either piece is itself suboptimal, split it again, continuing until all remaining polymers are Pareto-optimal. This process terminates because each split strictly decreases the number of monomers in each piece. Eventually this yields a decomposition of the form:

$$\mathbf{p} = \boldsymbol{\mu}_1 + \cdots + \boldsymbol{\mu}_k, \quad k \geq 2, \quad \boldsymbol{\mu}_i \in P^*,$$

where every split along the way is non-complementary. So we have the equilibrium relationship:¹²

$$[\mathbf{p}] = \prod_{i=1}^k [\mu_i]. \quad (7)$$

Fix one such decomposition for each $\mathbf{p} \in P \setminus P^*$ (although a suboptimal polymer may admit several decompositions). Note that, after fixing the decompositions, the map $\mathbf{p} \mapsto (\mu_1, \dots, \mu_k)$ is an injective map from the set of Pareto-suboptimal polymers to the set of tuples of Pareto-optimal polymers. Substituting (7) gives

$$C_{\text{sub}} = \sum_{\mathbf{p} \in P \setminus P^*} [\mathbf{p}] = \sum_{\mathbf{p} \in P \setminus P^*} \prod_{i=1}^{k(\mathbf{p})} [\mu_i(\mathbf{p})].$$

Each suboptimal polymer $\mathbf{p}$ contributes exactly one term to this sum, corresponding to its chosen tuple $(\mu_1, \dots, \mu_k)$ with $k \geq 2$. Since every such tuple is in particular an ordered tuple of elements of P^* , and all terms are non-negative, we may bound the sum by ranging over all ordered tuples of $k \geq 2$ elements of P^* .¹³

$$C_{\text{sub}} \leq \sum_{k=2}^{\infty} \sum_{\mu_1, \dots, \mu_k \in P^*} \prod_{i=1}^k [\mu_i].$$

The inner sum factorizes as a product of independent sums:

$$\sum_{\mu_1, \dots, \mu_k \in P^*} \prod_{i=1}^k [\mu_i] = \left(\sum_{\mu \in P^*} [\mu] \right)^k = C_{\text{opt}}^k.$$

Summing the resulting geometric series for $C_{\text{opt}} < 1$ completes the proof:

$$C_{\text{sub}} \leq \sum_{k=2}^{\infty} C_{\text{opt}}^k = \frac{C_{\text{opt}}^2}{1 - C_{\text{opt}}}. \quad \blacktriangleleft$$

A.2 Proof for Thermodynamic Justification

Proof of Lemma 5. Let C be a configuration containing a Pareto-suboptimal polymer $\mathbf{x}$. Then there exists a split $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2$ with nonzero $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{N}^M$ such that $\mathbf{x}_1$ and $\mathbf{x}_2$ are not complementary. Let C' be the configuration obtained from C by replacing one copy of $\mathbf{x}$ with one copy each of $\mathbf{x}_1$ and $\mathbf{x}_2$. Since no bond is broken in this split, $H(C') = H(C)$. Since the number of separate polymer units increases by one, $S(C') = S(C) + 1$. Therefore

$$G(C') = a \cdot H(C') - b \cdot S(C') = G(C) - b < G(C),$$

so C' has strictly lower free energy than C . Since this holds for any configuration containing $\mathbf{x}$, no such configuration can be a minimum free energy configuration. $\blacktriangleleft$

¹²For example, consider the split $\mathbf{p} = \mu_1 + \mathbf{p}_2 = \mu_1 + (\mu_2 + \mu_3)$ where μ_1, μ_2, μ_3 are Pareto-optimal. The two intermediate splits $\mathbf{p} = \mu_1 + \mathbf{p}_2$ and $\mathbf{p}_2 = \mu_2 + \mu_3$ are non-complementary, so we have $[\mathbf{p}_2] = [\mu_2] \cdot [\mu_3]$ and $[\mathbf{p}] = [\mu_1] \cdot [\mathbf{p}_2]$ by Equation (2), which gives $[\mathbf{p}] = [\mu_1][\mu_2][\mu_3]$ as desired.

¹³For example, suppose $P^* = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ and the suboptimal polymers are $\mathbf{a} + \mathbf{b}$, $\mathbf{a} + \mathbf{c}$, and $\mathbf{a} + \mathbf{b} + \mathbf{c}$ for simplicity. Then $C_{\text{sub}} = [\mathbf{a}][\mathbf{b}] + [\mathbf{a}][\mathbf{c}] + [\mathbf{a}][\mathbf{b}][\mathbf{c}]$, while the bound sums $([\mathbf{a}] + [\mathbf{b}] + [\mathbf{c}])^k$ over $k \geq 2$. The $k = 2$ term alone already exceeds the first two contributions, and the $k = 3$ term covers the third. Each power includes spurious terms (e.g. $[\mathbf{a}]^2$ or $[\mathbf{b}]^3$) that do not correspond to any suboptimal polymer, so the bound is loose, but all extra terms are non-negative.

A.3 Proof of Theorem 9

We observe the following lemma first.

► **Lemma 14.** *If $\mathbf{x} = \pi(\mathbf{h}) \in \pi(H) \setminus \{\mathbf{0}\}$, then $\mathbf{h} = \nu(\mathbf{x})$.*

Proof. Choose $\mathbf{h} \in H$ such that $\mathbf{x} = \pi(\mathbf{h})$. Observe that $\mathbf{e}_{u_a} + \mathbf{e}_{u_{a^*}} \in \mathcal{C}(A')$ for all $a \in D$. We will show that $\mathbf{h}$ is dichotomous, so it is the unique dichotomous neutral lift $\nu(\mathbf{x})$. Suppose that $(\mathbf{h})_{u_a} \cdot (\mathbf{h})_{u_{a^*}} \neq 0$ holds for some $a \in D$. Then both of them are positive integers because of $\mathbf{h} \geq \mathbf{0}$, which implies that $\mathbf{h} - \mathbf{e}_{u_a} - \mathbf{e}_{u_{a^*}} \in \mathcal{C}(A')$. This means that

$$\mathbf{h} = (\mathbf{e}_{u_a} + \mathbf{e}_{u_{a^*}}) + (\mathbf{h} - \mathbf{e}_{u_a} - \mathbf{e}_{u_{a^*}})$$

holds, which contradicts the definition of the Hilbert basis. This shows that $\mathbf{h}$ must be dichotomous. By the uniqueness in Lemma 7, $\mathbf{h}$ must be $\nu(\mathbf{x})$, so $\nu(\mathbf{x}) = \mathbf{h} \in H$. ◀

Proof of Theorem 9. We prove the two inclusions separately.

First inclusion $\pi(H) \setminus \{\mathbf{0}\} \subseteq P^*$. First, choose $\mathbf{x} \in \pi(H) \setminus \{\mathbf{0}\}$. Toward contradiction, assume that $\mathbf{x} \notin P^*$. This means that $\mathbf{x}$ is Pareto-suboptimal, so that there exists a non-complementary split $\mathbf{x} = \mathbf{y} + \mathbf{z}$ with nonzero $\mathbf{y}, \mathbf{z}$ such that $(A\mathbf{y})_a \cdot (A\mathbf{z})_a \geq 0$ for all $a \in D$. In other words, for each a , $(A\mathbf{y})_a, (A\mathbf{z})_a$, as well as $(A\mathbf{x})_a$ are all simultaneously nonnegative (i.e., ≥ 0) or simultaneously nonpositive (≤ 0).

Because all have the same sign, the neutralization $\nu(\mathbf{x}), \nu(\mathbf{y}), \nu(\mathbf{z})$ all add only one of u_a or u_{a^*} for each a . From this, we have $\nu(\mathbf{x}) = \nu(\mathbf{y}) + \nu(\mathbf{z})$. However, this is a contradiction because Lemma 14 shows that $\nu(\mathbf{x}) \in H$, which must not be written as the sum of two nonzero $\nu(\mathbf{y}), \nu(\mathbf{z}) \in \mathcal{C}(A')$. This shows that $\pi(H) \setminus \{\mathbf{0}\} \subseteq P^*$.

Second inclusion $P^* \subseteq \pi(H) \setminus \{\mathbf{0}\}$. Next, choose $\mathbf{x} \in P^*$, which is nonzero by definition, and assume that $\mathbf{x} \notin \pi(H) \setminus \{\mathbf{0}\}$. This shows that $\nu(\mathbf{x}) \notin H$, otherwise $\mathbf{x} = \pi(\nu(\mathbf{x})) \in \pi(H) \setminus \{\mathbf{0}\}$.

By the definition of the Hilbert basis, there are two nonzero $\mathbf{v}, \mathbf{w} \in \mathcal{C}(A')$ such that $\nu(\mathbf{x}) = \mathbf{v} + \mathbf{w}$, which gives a split $\mathbf{x} = \pi(\mathbf{v}) + \pi(\mathbf{w})$. We will show that this is a non-complementary split so that $\mathbf{x}$ is Pareto-suboptimal.

Given $\mathbf{v}, \mathbf{w} \geq \mathbf{0}$ and the dichotomous property of $\nu(\mathbf{x})$, the split polymers $\mathbf{v}, \mathbf{w}$ must also be dichotomous. Furthermore, for each $a \in D$, one of the two unit-monomer coordinates is zero in all three vectors:

$$(\nu(\mathbf{x}))_{u_a} = 0 \text{ and } (\mathbf{v})_{u_a} = (\mathbf{w})_{u_a} = 0, \text{ or } (\nu(\mathbf{x}))_{u_{a^*}} = 0 \text{ and } (\mathbf{v})_{u_{a^*}} = (\mathbf{w})_{u_{a^*}} = 0. \quad (8)$$

Suppose that $\pi(\mathbf{v}) = \mathbf{0}$ so that $\mathbf{x} = \pi(\mathbf{w})$. The polymer $\mathbf{w}$ is a dichotomous neutral element in $\pi^{-1}(\mathbf{x}) = \pi^{-1}(\pi(\mathbf{w}))$, which must be $\nu(\mathbf{x})$ by Lemma 7, leading to a contradiction: $\mathbf{v} = \nu(\mathbf{x}) - \mathbf{w} = \mathbf{0}$. Therefore $\pi(\mathbf{v}) \neq \mathbf{0}$ and similarly $\pi(\mathbf{w}) \neq \mathbf{0}$.

It remains to show that $\pi(\mathbf{v})$ and $\pi(\mathbf{w})$ are non-complementary. By Equation (3),

$$(A\pi(\mathbf{w}))_a + (\mathbf{w})_{u_a} - (\mathbf{w})_{u_{a^*}} = (A'\mathbf{w})_a = 0$$

holds for all $a \in D$, where the last equality is due to $\mathbf{w} \in \mathcal{C}(A')$. By Equation (8), one of the following must be true:

$$(A\pi(\mathbf{w}))_a + (\mathbf{w})_{u_a} = (A\pi(\mathbf{v}))_a + (\mathbf{v})_{u_a} = 0 \text{ or } (A\pi(\mathbf{w}))_a - (\mathbf{w})_{u_{a^*}} = (A\pi(\mathbf{v}))_a - (\mathbf{v})_{u_{a^*}} = 0.$$

In any case, $(A\pi(\mathbf{w}))_a$ and $(A\pi(\mathbf{v}))_a$ are both nonnegative or nonpositive because of $\mathbf{v}, \mathbf{w} \geq \vec{0}$, so that $(A\pi(\mathbf{w}))_a \cdot (A\pi(\mathbf{v}))_a \geq 0$ for all $a \in D$. This proves that $\pi(\mathbf{v})$ and $\pi(\mathbf{w})$ are non-complementary. Since $\pi(\mathbf{v}), \pi(\mathbf{w}) \neq \mathbf{0}$ and $\mathbf{x} = \pi(\mathbf{v}) + \pi(\mathbf{w})$, this is a non-complementary split of $\mathbf{x}$, contradicting $\mathbf{x} \in P^*$. Therefore $\mathbf{x} \in \pi(H) \setminus \{\mathbf{0}\}$.

Combining the two inclusions, we conclude the proof. $\blacktriangleleft$

A.4 Proof for Covering Design Strategy

Proof of Lemma 10. First, if $\mathbf{p} \in P_S^* \subset P^*$, then any split $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$ for nonzero $\mathbf{p}_1, \mathbf{p}_2 \in \text{Supp}(S) \subset \mathbb{N}^M$ must be complementary. Therefore, $\mathbf{p}$ is a Pareto-optimal polymer in (Σ, S) after embedding.

On the other hand, if $\mathbf{p}$ is a Pareto-optimal polymer in (Σ, S) after embedding, it is clear that $\mathbf{p} \in \text{Supp}(S)$. Also, any split of it $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$ for nonzero $\mathbf{p}_1, \mathbf{p}_2 \in \text{Supp}(S)$ must be complementary. We showed that all splits must be in this form, thus $\mathbf{p}$ is in P^* . This proves the equivalence. $\blacktriangleleft$

Proof of Lemma 11. For the first statement, observe that the entries in $M \setminus S$ are always 0 and can be removed. After removal, this is exactly Theorem 9 on the system (Σ, S) .

For the second statement, any $\mathbf{p} \in P_t^*$ has $|\text{supp}(\mathbf{p})| \leq t$, so $\text{supp}(\mathbf{p})$ is contained in some subset $S \subseteq M$ with $|S| = t$. Hence $\mathbf{p} \in P_S^* = \pi(H_S) \setminus \{\mathbf{0}\}$. Conversely, if $\mathbf{p} \in \pi(H_S) \setminus \{\mathbf{0}\}$ for some $S \subseteq M$ with $|S| = t$, then $\mathbf{p} \in P_S^* \subseteq P^*$ and $|\text{supp}(\mathbf{p})| \leq |S| = t$, so $\mathbf{p} \in P_t^*$. $\blacktriangleleft$

Proof of Theorem 13. The inclusion $\subseteq P^*$ holds because $\pi(H_B) \setminus \{\mathbf{0}\} = P_B^* = P^* \cap \text{Supp}(B) \subset P^*$.

For the first inclusion, it suffices to show that $P_S^* \subseteq \hat{P}_{k,t}$ for all $|S| = t$ because of Lemma 11. By the definition of the covering design and Equation (5), there exists $B \in \mathcal{D}$ such that $S \subseteq B$ so that $P_S^* \subseteq P_B^*$. This concludes the proof. $\blacktriangleleft$

B Scalability Limitations

Theorem 9 shows that the set of Pareto-optimal polymers P^* is finite and computable. We can employ a modern implementation, such as `Normaliz`, to compute the Hilbert basis H , which gives the set of Pareto-optimal polymers for a relatively small domain-monomer system. However, this approach quickly becomes impractical as the system grows. We give some explanation of this phenomenon in the next subsection.

A natural way to remedy this inefficiency is to terminate the algorithm early, expecting to obtain a part of the full Hilbert basis. Unfortunately, the state-of-the-art implementations of Hilbert basis algorithms usually enumerate a superset of the Hilbert basis first and then iteratively reduce it to the Hilbert basis. This algorithmic aspect makes it difficult to introduce a meaningful stopping criterion, as we are not aware of any other way to understand the intermediate superset except for the reduction to the Hilbert basis.

We have made several attempts to employ scalable algorithms, which turned out to be inefficient. We explain two notable approaches below.

► **Remark 15 (Alternative representation).** We can use the cone $\{\mathbf{x} \in \mathbb{N}^{|M|+|D|} \mid A'\mathbf{x} \geq \mathbf{0}\}$ and its Hilbert basis by only adding the complementary unit monomers $\{u_{a^*}\}_{a \in D}$. In this setting, the net domain vector has nonnegative entries instead of being a zero vector. This formulation and computation ultimately gives the same result as in this paper, but modern implementations such as `Normaliz` are more native to the cone $A'\mathbf{x} = \mathbf{0}$, as we present in the main body.

B.1 Hilbert Basis Hardness

The size of the Hilbert basis rapidly increases along several parameters, and counting the number of Hilbert basis elements is known to be NP-hard (strictly speaking, #P-hard [17]). In fact, the Hilbert basis of a pointed cone defined by a tiny matrix can be arbitrarily large: the Hilbert basis of the pointed cone $\mathcal{C}_{\mathbb{R}}(A_k) = \{\mathbf{x} \in \mathbb{R}^3 \mid A_k \mathbf{x} = 0, \mathbf{x} \geq 0\}$ for $A_k = \begin{bmatrix} 1 & 1 & -k \end{bmatrix}$ is

$$H_k = \{(i, k-i, 1) \mid i = 0, \dots, k\}.$$

In this example, the size of H_k increases exponentially for the bit-length of describing A_k , i.e., the input size.

B.2 Failed Approaches

This subsection introduces two slow approaches, highlighting that it is not clear how to obtain a practical, scalable algorithm by either (i) employing an older scalable Hilbert basis algorithm or (ii) directly exploiting properties of the Pareto-optimal polymers. This motivates the main approach in Section 5, employing both modern Hilbert basis algorithms and properties of the Pareto-optimal polymers.

The first approach is based on an old Hilbert basis algorithm [8]. This algorithm inspects candidate basis elements by increasing the target 1-norm (i.e., the sum of absolute values of entries). This is exceedingly slow, especially because of the enormous number of small-norm polymers. The algorithm reduces the search size using some geometric ideas, but our implementation failed to terminate even for small examples.

Another approach is to use the following nontrivial fact for Pareto-optimal polymers.

► **Lemma 16.** *For every Pareto-optimal polymer $\mathbf{p} \in P^*$ that is not a monomer, there exist nonzero $\mathbf{p}_1, \mathbf{p}_2 \in P^*$ such that $\mathbf{p} = \mathbf{p}_1 + \mathbf{p}_2$.*

We will not prove this lemma here because it is not relevant to the main body of this paper. Given this lemma, we can recursively enumerate P^* by repeatedly merging pairs $\mathbf{p}, \mathbf{q}$ of previously enumerated Pareto-optimal polymers and checking whether the resulting $\mathbf{p} + \mathbf{q}$ is Pareto-optimal. Our implementation (with some optimizations) shows that this approach is also much slower than the simple Hilbert basis algorithm using Theorem 9.

C Domain Support Mode

This section introduces an alternative way to make the computation of the Pareto-optimal polymers scalable, based on the domain supports instead of the monomer supports. In the *domain mode*, subset enumeration is applied over domain pairs rather than monomer types. This may be advantageous when the number of domains is relatively smaller than the number of monomers, or when many monomers share the same domain pairs, as in cascade and binary-tree TBNs, in which case the relevant covering design size $c(|D|, k, t)$ may be significantly smaller than $c(|M|, k, t)$. Further details are provided in Section 4 of the supplementary material.

D TBN System Examples

D.1 Linear Cascade 4 Module AND Gate System

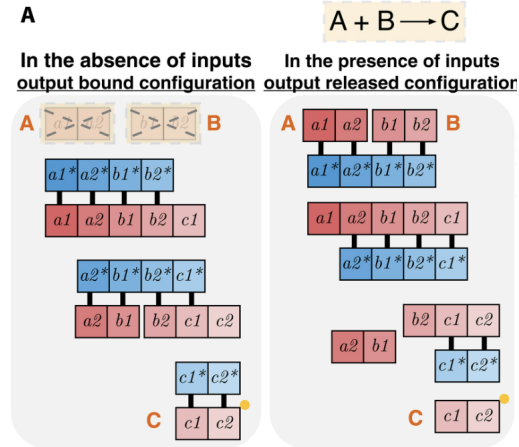


Figure 7 Mechanism of a single AND-gate module in the linear cascade system. In the presence of both input polymers, the module can rearrange to release the output polymer, which then serves as an input to the next module in the cascade. Figure adapted from Fig. 3 of [22].

```
inputs: x1_1 x1_2
        x2_1 x2_2
        x3_1 x3_2
        x4_1 x4_2
        x5_1 x5_2

# Module 1: x1 + x2 -> y1
x1_1 x1_2 x2_1 x2_2 y1_1
x1_1* x1_2* x2_1* x2_2*
x1_2 x2_1
x2_2 y1_1 y1_2
x1_2* x2_1* x2_2* y1_1*
y1_1* y1_2*
outputy1: y1_1 y1_2

# Module 2: y1 + x3 -> y2
y1_1 y1_2 x3_1 x3_2 y2_1
y1_1* y1_2* x3_1* x3_2*
y1_2 x3_1
x3_2 y2_1 y2_2
y1_2* x3_1* x3_2* y2_1*
y2_1* y2_2*
outputy2: y2_1 y2_2

# Module 3: y2 + x4 -> y3
y2_1 y2_2 x4_1 x4_2 y3_1
y2_1* y2_2* x4_1* x4_2*
y2_2 x4_1
x4_2 y3_1 y3_2
y2_2* x4_1* x4_2* y3_1*
y3_1* y3_2*
outputy3: y3_1 y3_2

# Module 4: y3 + x5 -> y4
y3_1 y3_2 x5_1 x5_2 y4_1
y3_1* y3_2* x5_1* x5_2*
y3_2 x5_1
x5_2 y4_1 y4_2
y3_2* x5_1* x5_2* y4_1*
y4_1* y4_2*
outputy4: y4_1 y4_2
```

D.2 2-2 Linear Cascade System

```

x11 x12 x13
x21 x22 x23
x11 x12 x13 x21 x22 x23 y11 z11
x11* x12* x13* x21* x22* x23*
x12 x13 x22 x23 y11 y12 z11 z12
x12* x13* x22* x23* y11* z11*
x13 x23 y11 y12 y13 z11 z12 z13
x13* x23* y11* y12* z11* z12*
y11* y12* y13* z11* z12* z13*
y11 y12 y13
z11 z12 z13
y11 y12 y13 z11 z12 z13 y21 z21
y11* y12* y13* z11* z12* z13*
y12 y13 z12 z13 y21 y22 z21 z22
y12* z13* z12* z13* y21* z21*
y13 z13 y21 y22 y23 z21 z22 z23
y13* z13* y21* y22* z21* z22*
y21* y22* y23* z21* z22* z23*
y21 y22 y23
z21 z22 z23

```

D.3 Scaffolded DNA System

```

s[1]* s[2]* s[3]* s[4]* s[5]* s[6]* s[7]* s[8]* s[9]* s[10]*
t0[1] s[1] t0[2]*
t1[1] s[1] t1[2]*
t0[2] s[2] t0[3]*
t1[2] s[2] t1[3]*
t0[3] s[3] t0[4]*
t1[3] s[3] t1[4]*
t0[4] s[4] t0[5]*
t1[4] s[4] t1[5]*
t0[5] s[5] t0[6]*
t1[5] s[5] t1[6]*
t0[6] s[6] t0[7]*
t1[6] s[6] t1[7]*
t0[7] s[7] t0[8]*
t1[7] s[7] t1[8]*
t0[8] s[8] t0[9]*
t1[8] s[8] t1[9]*
t0[9] s[9] t0[10]*
t1[9] s[9] t1[10]*
t0[10] s[10]
t1[10] s[10]

```

D.4 AND Gate Binary Tree System

```

input: x1_1 x1_2, x2_1 x2_2, x3_1 x3_2, x4_1 x4_2
# Module 1: x1 + x2 -> y1
x1_1 x1_2 x2_1 x2_2 y1_1
x1_1* x1_2* x2_1* x2_2*
x1_2 x2_1
x2_2 y1_1 y1_2
x1_2* x2_1* x2_2* y1_1*
y1_1* y1_2*
outputy1: y1_1 y1_2
# Module 2: x3 + x4 -> y2
x3_1 x3_2 x4_1 x4_2 y2_1
x3_1* x3_2* x4_1* x4_2*
x3_2 x4_1
x4_2 y2_1 y2_2
x3_2* x4_1* x4_2* y2_1*
y2_1* y2_2*
outputy2: y2_1 y2_2
# Module 3: y1 + y2 -> y3
y1_1 y1_2 y2_1 y2_2 y3_1
y1_1* y1_2* y2_1* y2_2*
y1_2 y2_1
y2_2 y3_1 y3_2
y1_2* y2_1* y2_2* y3_1*
y3_1* y3_2*
outputy3: y3_1 y3_2

```

Contrastive Hebbian Learning for Multicomponent Liquids

Yancheng Du ✉ 

California Institute of Technology, Pasadena, CA, USA

Cameron Chalk ✉ 

California Institute of Technology, Pasadena, CA, USA

Salvador Buse ✉ 

California Institute of Technology, Pasadena, CA, USA

Lulu Qian ✉ 

California Institute of Technology, Pasadena, CA, USA

Erik Winfree ✉ 

California Institute of Technology, Pasadena, CA, USA

Abstract

Molecules with designed interactions can serve as substrates for information processing; demonstrated examples include algorithmic self-assembly of DNA tiles and DNA strand displacement reactions. These two well-established paradigms correspond to solid-phase-like behavior, where spatially structured molecular assemblies grow by crystalline attachment, and gas-phase-like behavior, where a dilute mixture of freely diffusing molecules is governed by mass-action kinetics of reactions. A third paradigm, liquid-liquid phase separation, is a fundamental phenomenon in physical and biological systems, giving rise to membraneless compartments that facilitate complex information processing. Recent studies have shown that multicomponent liquid mixtures described by lattice models are analogous to Boltzmann machines and can implement neural computation. Mean-field models have also suggested a connection to Hopfield networks and illustrated complex decision-making during condensation from a reservoir. As an alternative to previous approaches to train liquid interaction energies using backpropagation through a loss function, here we derive “local” learning rules based on contrastive Hebbian learning that may be more biologically and physically plausible. The trained systems demonstrate a range of computational tasks, including associative recall, linear and nonlinear input-output relations, pattern classification, and spatial phase separation. Our work suggests that the computational potential of liquid-phase molecular systems could be unlocked in real physical systems with local learning rules.

2012 ACM Subject Classification Hardware → Neural systems; Applied computing → Physics; Applied computing → Systems biology

Keywords and phrases multicomponent liquid, liquid-liquid phase separation, continuous Hopfield network, contrastive Hebbian learning

Digital Object Identifier 10.4230/LIPIcs.DNA.32.13

Supplementary Material *Software (Source Code, Notebooks, and Supplementary Information):*

<https://github.com/YanchengDu/LiquidCHL>

archived at `swb:1:dir:b89e452e1236f62498ae55537e3dd62972e6d523`

Funding *Yancheng Du*: Schmidt Sciences Polymath Award to LQ.

Cameron Chalk: NSF CCF/FET 2212546.

Salvador Buse: Open Philanthropy Good Ventures Foundation, NSF CCF/FET Award 2212546.

Lulu Qian: Schmidt Sciences Polymath Award, NSF CCF/FET 2212546.

Erik Winfree: NSF CCF/FET 2212546.

Acknowledgements The authors want to thank Inhoo Lee, Sandra Temgoua, Daichi Hayakawa, Arvind Murugan, Krishna Shrinivas, and others for helpful discussions.



© Yancheng Du, Cameron Chalk, Salvador Buse, Lulu Qian, and Erik Winfree; licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 13; pp. 13:1–13:27

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Molecular interactions are fundamental to cellular activities [66, 39]. While molecular biology has traditionally studied how molecular structure governs mechanical and biochemical function [68, 45, 7], the perspective of information processing at the systems level is increasingly appreciated [75, 3, 71, 9]. The design of synthetic biomolecular systems that perform computation, both theoretical and experimental, provides an approach to clearly isolate and understand how specific molecular mechanisms and phenomena relate to information processing [51, 64, 77, 52, 50]. Two prominent paradigms within DNA nanotechnology [57] are algorithmic self-assembly of DNA tiles, where information is encoded spatially within multicomponent aperiodic crystalline structures whose growth by cooperative attachment events can emulate cellular automata and Boolean logic circuits and neural pattern recognition [77, 76, 23, 78, 22, 24], and DNA strand displacement reaction networks, where information is encoded in the concentrations of freely diffusing molecules whose mass-action reaction kinetics can emulate analog dynamical circuits and Boolean logic circuits and neural computation [53, 82, 19, 65, 62, 43], including supervised learning and reservoir computing [42, 20, 28, 80]. These two paradigms respectively represent solid-phase molecular mechanisms, in the sense that DNA tiles form (relatively) permanent bonds with their neighbors, and gas-phase mechanisms, in the sense that DNA strand displacement complexes operate at dilute concentrations such that they are mostly not in contact with any neighbor except during brief reactive interactions.

The liquid phase is conceptually situated between these two states of matter, in the sense that molecules are constantly in contact with their neighbors but rapidly changing which molecules are their neighbors. In living cells, liquid-liquid phase separation (LLPS) drives the formation of membraneless organelles, which are condensates consisting of proteins and nucleic acids that are involved in the regulation of intracellular biochemical processes [8, 35]. Condensate assembly is governed by the interaction strengths of many molecular species, a feature that gives them a rich potential for complex phase behavior [36, 37, 47] that can be explored synthetically with multicomponent DNA nanostar systems [49, 38, 15]. Together, the biological ubiquity of liquid-phase condensates and the demonstrated programmability of synthetic nucleic acid systems establish multicomponent liquids as a strong candidate for a third paradigm for molecular information processing based on liquid-phase mechanisms [72].

Recent work using several complementary modeling frameworks has highlighted the potential for multicomponent liquid mixtures to process information by phase separation. Multicomponent liquid lattice models explicitly capture local spatial arrangements of molecules and solvent; Chalk et al. showed that such a framework has a formal connection to the Boltzmann machine, providing both theoretical grounding and an established learning method [16]. The mean-field model, by contrast, averages over spatial positions and describes the mixture solely through bulk composition vectors, making it analytically tractable. Teixeira et al. pointed out that bulk phases of multicomponent liquids can be metastable, supporting associative memory retrieval analogous to the Hopfield network when a Hebbian rule was used for learning molecular interaction energies, albeit only with a cubic modification to the standard energy model [12]. This “liquid Hopfield model” was then extended to a continuous spatial model of LLPS, which exhibited complex domains of the metastable phases [70]. Zentner et al. considered the case of condensation in a “surface volume” around fixed “input” molecules and used backpropagation training to find interaction energies and reservoir potential energies such that condensation of different phases implemented complex classification tasks [81]. The rich behaviors exhibited in these works motivate the search for

simpler “local” learning algorithms that don’t require global loss functions and backpropagation of gradient terms yet are still sufficient for tuning multicomponent phase separation for a variety of information processing tasks.

The continuous Hopfield network (CHN) model is a well-studied framework in neural networks for associative memory and optimization [31, 32, 33]. In this model, neurons occupy continuous states and are connected by symmetric interaction weights that define an energy landscape. The system follows deterministic dynamics that monotonically descend on the energy, as guaranteed by the existence of a Lyapunov function, and converge to stable attractors that act as stored memories or solutions to optimization problems. Contrastive Hebbian learning (CHL) is a natural training method for this framework, having originally been introduced for continuous Hopfield networks. The training method relies on energy minimization with local update rules rather than explicit loss functions, where local means that states of two interacting neurons are sufficient to determine their weight update. CHL has been shown to be equivalent to backpropagation under certain conditions [48, 79].

In this paper, we show that CHL can be adapted to two previously-studied models of multicomponent LLPS: the surface condensation model of Zentner et al. [81] and the spatial phase separation model of Teixeira et al. [70] without the cubic terms. The trained systems demonstrate a range of computational tasks, including associative recall, linear and nonlinear input-output relations, classification of MNIST digits, and programmable spatial phase separation with coexisting target phases. These results help establish multicomponent liquid mixtures as a physically realizable and trainable computational substrate, and cement CHL as a versatile learning rule for exploring the design space.

2 Learning in a surface model of multicomponent liquids

In this section, we introduce a mean-field model of a single-compartment liquid mixture in exchange with a reservoir that is equivalent to the surface condensation model of Zentner et al. [81]. Their paper discusses in detail the physical and biological credibility of the model. The mean-field energy can be derived from a lattice model (Figure 1a) and the following assumptions [26, 34, 81]: we consider a single well-mixed compartment of an incompressible liquid mixture with n equal-volume components and a fixed volume v . The composition vector fully characterizes the system state of the mixture: $\vec{\phi} = (\phi_1, \phi_2, \dots, \phi_n)$, where ϕ_i is the volume fraction of the i^{th} component and is subject to $\sum_{i=1}^n \phi_i = 1$. Thus, an n -component system has $n - 1$ independent variables. χ_{ij} represents the effective interaction between the liquid components, with positive values indicating a tendency to demix and negative values indicating a tendency to mix. It is defined as $\chi_{ij} = \frac{z}{RT}(G_{ij} - \frac{G_{ii} + G_{jj}}{2})$, where z is the average number of interactions (valency) and G_{ij} is the microscopic interaction between components i and j . Consequently, χ_{ij} is symmetric and diagonal terms χ_{ii} are equal to 0. Then the free energy of the reservoir-coupled compartment is described by a Flory-Huggins energy density with equal-volume species [26, 34]:

$$f_{FH}(\vec{\phi}) = cRT \left(\sum_{i,j=1}^n \frac{1}{2} \phi_i \chi_{ij} \phi_j + \sum_{i=1}^n \phi_i \log \phi_i \right) - c \sum_{i=1}^n \mu_{i,res} \phi_i \quad (1)$$

where c is the total molecular concentration, R is the gas constant, T is the absolute temperature, and $\mu_{i,res}$ is the external reservoir potential. For simplicity, we will make energy density dimensionless by setting $c = 1$ and $RT = 1$.

13:4 Contrastive Hebbian Learning for Multicomponent Liquids

Remarkably, this energy is formally similar to the standard Lyapunov function for continuous Hopfield networks [32, 73]

$$E_{CHN}(\vec{x}) = -\frac{1}{2} \sum_{i,j} w_{ij} x_i x_j - \sum_i b_i x_i + \sum_i \int_0^{x_i} g^{-1}(x) dx. \quad (2)$$

The corresponding dynamics are

$$\frac{ds_i}{dt} = -\frac{\partial E_{CHN}}{\partial x_i} = \sum_j w_{ij} x_j + b_i - s_i \quad \text{where} \quad x_i = g(s_i) \quad (3)$$

where x_i is the continuous activity of neuron i , w_{ij} gives the symmetric zero-diagonal synaptic weight matrix, b_i is the bias, and g is the activation function relating the neural input s_i to its output x_i . When $g(s) = 1/(1 + \exp(-s))$ so $0 < x_i < 1$, the last term of eqn 2 is $\sum_i x_i \log x_i + (1 - x_i) \log(1 - x_i)$, which is similar to the middle term of eqn 1, while the other terms are identical if $x_i = \phi_i$ and $w_{ij} = -\chi_{ij}$ and $b_i = \mu_{i,res}$. This leaves the major difference of the constraint, present only in surface condensation, that $\sum \phi_i = 1$. Interestingly, a known generalization of the continuous Hopfield model also bakes in this constraint, with the motivation that constraints of this sort are useful for optimization tasks such as the traveling salesman problem, in which case the natural Lyapunov function is identical to the Flory-Huggins energy [73]. Figure 1b illustrates the connection to the continuous Hopfield network for the case of a simple 2-component system, and emphasizes the impact of the constraint. Figures 1c and d illustrate how, like the continuous Hopfield model despite differences due to the constraint, the number of metastable states in the Flory-Huggins landscape can scale with the number of components.

A natural dynamics for the surface model is the model A dynamics [30, 81], which corresponds to gradient descent on the energy f_{FH} subject to the $\sum_i \phi_i = 1$ constraint while allowing mass exchange with the reservoir. Here, we use a Lagrange multiplier λ to enforce the constraint. We will use the modified free energy $f_{\lambda FH} = f_{FH} + \lambda(\sum_{i=1}^n \phi_i - 1)$ where $\lambda = -\frac{1}{n} \sum_{i=1}^n \mu_{i,FH} = -\langle \mu_{FH} \rangle$ is dynamically determined so as to ensure the constraint remains satisfied (in which case $f_{\lambda FH} = f_{FH}$). The modified dynamics then become

$$\frac{d\phi_i}{dt} = -M \frac{\partial f_{\lambda FH}}{\partial \phi_i} = -M(\mu_{i,FH} + \lambda) \quad (4)$$

where we define $\mu_{i,FH} = \frac{\partial f_{FH}}{\partial \phi_i} = 1 + \log \phi_i + \sum_{j=1}^n \chi_{ij} \phi_j - \mu_{i,res}$ as the chemical potential of species i and $M > 0$ is the constant mobility. The dynamics monotonically decrease this energy with an equilibrium condition $\mu_{i,FH} = \langle \mu_{FH} \rangle$ (see Appendix A.2 for detailed proof). Thus, the constrained minima of the liquid system's energy are dynamically stable fixed points under model A dynamics, making them the physical analogs of memory attractors in Hopfield networks. However, their usefulness as associative memories depends on whether energy parameters can be learned, whether arbitrary memory patterns can be stored or a limited subset (Hopfield's result pertained to random patterns that may have many overlapped bits but are statistically orthogonal), and whether the basins are structured such that the dynamics starting from perturbed patterns generally leads to correct recall (as opposed to spurious states or reversion to a single dominant memory). In the classical continuous Hopfield model, a Hebbian learning rule is used to set these parameters [31, 32]. However, the volume-fraction constraint reduces the effective degrees of freedom in the liquid mixture and alters the stability conditions relative to the unconstrained case [74]. Consequently, the Hebbian learning rule in the classical model may not be as effective here.

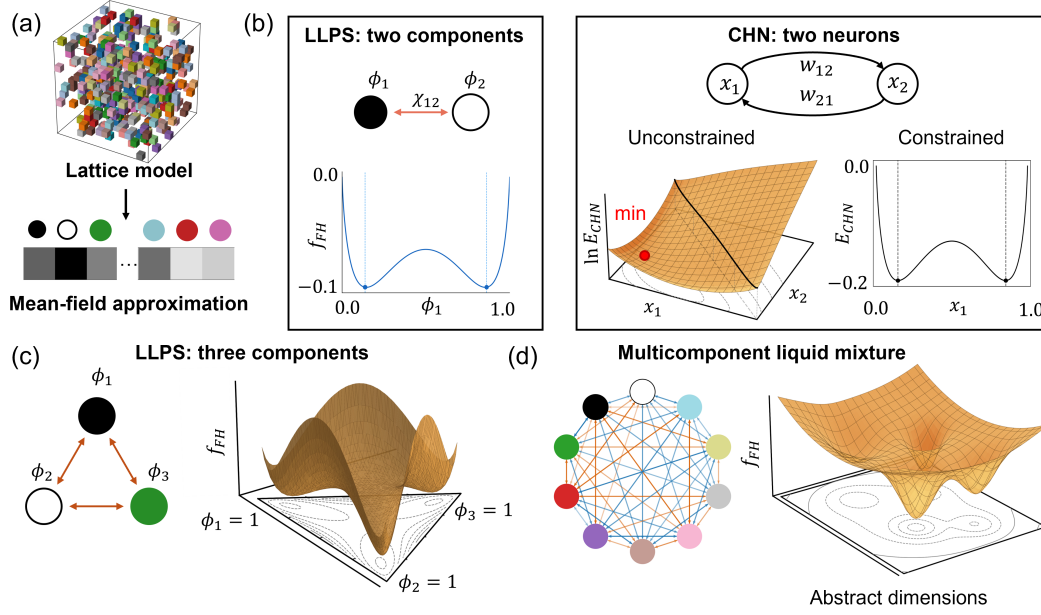


Figure 1 Energy landscapes of multicomponent liquids. (a) The liquid mixture model with Flory-Huggins energy is an approximation of a lattice model in the mean-field limit. (b) Left: A representative Flory-Huggins energy for phase separation of a pair of repulsive components ($\chi_{12} = \chi_{21} = 5$) and zero reservoir potential ($\mu_{i,res} = 0$). Two stable states can be found, where one phase has a depleted ϕ_1 component and the other is enriched in ϕ_1 . Right: A continuous Hopfield network (CHN) with two mutually inhibitory neurons ($w_{12} = w_{21} = -10$ and $b_1 = b_2 = 0$). The unconstrained energy landscape has only one global minimum. When the constraint $x_1 + x_2 = 1$ is applied, the system has two stable attractors. (c) A ternary phase system with strong repulsive interactions shows three stable states in the energy landscape. (d) A conceptual illustration showing that the number of stable phases in an LLPS system scales with the number of components.

2.1 Contrastive Hebbian learning of stable target states

We now show that CHL provides a general method for training stable attractors in a liquid-phase mixture. CHL was originally introduced by Movellan to train continuous Hopfield networks for constrained optimization problems [48]. It belongs to a family of contrastive learning methods first developed for Boltzmann machine models [14, 29, 1]. The CHL framework has several distinctive properties worth noting. First, unlike backpropagation-based methods, the training is not specified by an explicit loss function on output states; the system is instead guided by example compositions. Second, hidden units can be incorporated naturally, enriching the representational capacity of the system. Third, the weight update rule takes the Hebbian-like “fire together, wire together” form, depending only on the local states of the two connected components. Unlike pure Hebbian learning that positively reinforces desired activity, contrastive learning also has a similarly local anti-Hebbian phase that disrupts undesired activity. We derive the weight update rules for our model below.

Because model A dynamics are deterministic, given any initial composition $\vec{\phi}^o$, the system always converges to a unique final phase vector $\vec{\phi}$ with associated free energy $\vec{f}_{FH}$, where a $(\cdot)$ denotes an equilibrium value. CHL minimizes the energy difference between the target phases and the system’s natural equilibrium, training the system to store target phases as stable local minima of the free-energy landscape.

When the system is clamped, it is partitioned into two subsets $\vec{\phi} = (\vec{\phi}_{\text{fixed}}, \vec{\phi}_{\text{free}})$, where a subset of components is held fixed at target values while the remaining components are free to relax. Physically, clamping a component corresponds to maintaining its volume fraction at a prescribed fixed value (for example through exchange with a modified external reservoir held at the necessary corresponding chemical potentials). The system starts from an initial clamped state and is driven by the dynamics while holding clamped variables fixed. At equilibrium, the clamped state $\check{\phi}^{(+)}$ is associated with energy $\check{f}_{FH}^{(+)}$. Next, the clamps are released, and the system further relaxes to the free state $\check{\phi}^{(-)}$ with energy $\check{f}_{FH}^{(-)}$. The increased degrees of freedom guarantee $\check{f}_{FH}^{(+)} \geq \check{f}_{FH}^{(-)}$.

By minimizing the contrastive objective representing the energy difference between the two states,

$$J = \check{f}_{FH}^{(+)} - \check{f}_{FH}^{(-)}, \quad (5)$$

we can train the system to place stable attractors at target phases that respect the clamped compositions $\vec{\phi}_{\text{fixed}}$. This is accomplished by learning the interaction matrix χ_{ij} and the reservoir chemical potentials $\mu_{i, \text{res}}$. A conceptual schematic is presented in Figure 2a.

We now derive the learning rules for the interaction matrix χ_{ij} :

$$\frac{\partial \check{f}_{FH}}{\partial \chi_{ij}} = \check{\phi}_i \check{\phi}_j + \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} (1 + \log \check{\phi}_k + \sum_{l=1}^n \chi_{kl} \check{\phi}_l - \mu_{k, \text{res}}) = \check{\phi}_i \check{\phi}_j + \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} \mu_{k, FH}. \quad (6)$$

Given the previously noted fact that at equilibrium $\mu_{i, FH} = \langle \mu_{FH} \rangle$ and by differentiating the constraint $\sum_{k=1}^n \check{\phi}_k = 1$ to obtain $\sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} = 0$, we can see that

$$\frac{\partial \check{f}_{FH}}{\partial \chi_{ij}} = \check{\phi}_i \check{\phi}_j + \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} \langle \mu_{FH} \rangle = \check{\phi}_i \check{\phi}_j + \langle \mu_{FH} \rangle \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} = \check{\phi}_i \check{\phi}_j. \quad (7)$$

As this equation holds for any initial state or clamping condition, we can now obtain

$$\frac{\partial J}{\partial \chi_{ij}} = \frac{\partial \check{f}_{FH}^{(+)}}{\partial \chi_{ij}} - \frac{\partial \check{f}_{FH}^{(-)}}{\partial \chi_{ij}} = \check{\phi}_i^{(+)} \check{\phi}_j^{(+)} - \check{\phi}_i^{(-)} \check{\phi}_j^{(-)}. \quad (8)$$

To learn interaction strengths by performing gradient descent on J , we need $\frac{d\chi_{ij}}{dt} = -\frac{\partial J}{\partial \chi_{ij}}$, and thus the update rule is $\Delta \chi_{ij} \propto -\check{\phi}_i^{(+)} \check{\phi}_j^{(+)} + \check{\phi}_i^{(-)} \check{\phi}_j^{(-)}$. By a similar derivation (given in Appendix A.3), the learning rule for the reservoir potentials is $\Delta \mu_{i, \text{res}} \propto \check{\phi}_i^{(+)} - \check{\phi}_i^{(-)}$.

Several aspects of the training procedure are worth noting. First, the initial composition affects which attractor the system converges to. A common strategy is for the clamped stage to be initialized with the fixed components set to the target and the remaining components starting from uniform values plus random perturbations, which corresponds to a fully mixed, disordered state. This uniform starting point ensures the dynamics are not biased toward any particular attractor, and the small noise breaks the compositional symmetry to allow convergence to one of the stored phases. For the free stage, the system is initialized from the equilibrium of the clamped stage. The process is illustrated in Figure 2a. Second, although the learning rule is derived for a single target phase, it naturally extends to storing multiple phases by defining the contrastive objective as an expectation over a distribution P of target phases $\langle J \rangle_P = \mathbb{E}_{\vec{\phi} \in P} [J]$. Gradient descent of $\langle J \rangle_P$ therefore corresponds to training that cycles through the set of target phases and averages the updates in each epoch. Thus the updates accumulate and superimpose to encode multiple attractors simultaneously, in direct

analogy to how the classical Hopfield network stores multiple patterns via summed Hebbian outer products. To encourage the learned system to remain “liquid-like”, we clip energies such that $|\chi_{ij}| < 25$ after every update. Finally, as with any Hopfield-like system, the trained network may develop spurious attractors arising from interference among the encoded phases. Spurious attractors become more prevalent as the number of stored phases approaches the storage capacity of the system [5, 13]. In the following sections, we demonstrate the method through three examples. Details of training methods and simulation methods are in Appendices A.1 and A.2.

2.2 Storing and retrieving target phases

Pattern storage and retrieval is the natural entry point, as it most directly tests whether the trained energy landscape stores the desired attractors. Here we consider binary patterns $\vec{x} = (x_1, \dots, x_n), x_i \in \{0, 1\}$. Each binary pattern is mapped to a composition $\vec{\phi}$ via a softmax function $\phi_i = e^{\beta x_i} / \sum_{i=1}^n e^{\beta x_i}$ and $\beta = 0.5$, ensuring positivity and sum to unity.

As a proof of concept, we train a 32-component mixture to store and retrieve five target phases with overlapping components. In binary phase separation (for example water/oil separation), components can be distinguished by their repulsion from components in the opposing phase. Here, each component must play a distinct role in each of the five trained phases, and the interaction matrix encodes a rich associative structure in which every component’s identity is defined by its relationships with other components across all stored phases simultaneously. The task complexity is approaching the storage capacity ($\sim 0.14n$) predicted for classic Hopfield networks [4]. The set of memories is shown in Figure 2c. Training proceeds by gradient descent on the contrastive objective defined above. In each epoch, the clamped target phase $\check{\phi}^{(+)}$ is sampled uniformly from the set of memories and the free phase $\check{\phi}^{(-)}$ is obtained by releasing from the clamped target phase. The interaction matrix is initialized using the Hebbian rule, and the reservoir chemical potentials are initialized to zero. The contrastive objective J averaged over each epoch, $\langle J \rangle_{epoch}$, tends to decrease with epochs and roughly converges to near 0 (Figure 2b).

Retrieval is tested by initializing the system with randomly perturbed versions of the stored memories. Figure 2d shows an example trajectory in which the free energy decreases from the initial random composition and the system converges to one of the stored phases. We characterize the retrieval performance of the trained system as a function of the flip ratio, defined as the fraction of components in the initial composition that are randomly reassigned relative to the target phase. We perform a systematic study by randomly generating sets of five memories, where each component appears in at least one memory but bits are otherwise unbiased. Target memory sets were sorted into bins based on the average number of overlapping components. We test memory retrieval for five memory sets each from bins centered on 4 to 12 overlaps and compare the performance to the classical Hopfield network on the same memory sets. The results are shown in Figure 2e. Retrieval rates decline with increasing flip ratio as expected, and for memories with low average overlap (4 and 6 components) the liquid-phase system performs comparably to the classical Hopfield network. At higher overlaps (greater than 8), performance degrades relative to the Hopfield baseline. These results confirm, consistent with [12, 70], that a multicomponent liquid mixture can perform associative recall across a non-trivial memory load – although, consistent with prior investigations of the capacity of multicomponent liquids to encode multiple phases [59, 83, 17, 54], their memory capacity may be lower than the $\sim 0.14n$ statistical capacity of classical Hopfield networks.

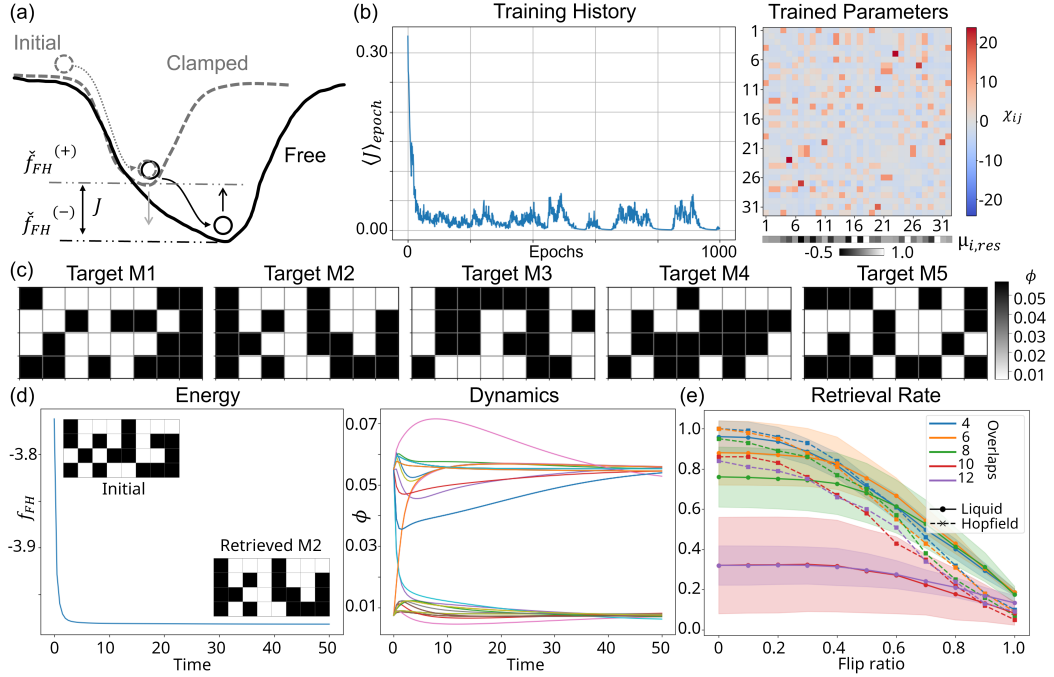


Figure 2 Training an associative memory. (a) Conceptual schematic of CHL. From an initial state with some components clamped, the system relaxes to an equilibrium state. In the free phase, some clamps are removed, and the system relaxes again to a new lower equilibrium, with increased degrees of freedom. The learning rule minimizes the contrastive objective to place stable attractors at the target compositions. (b) The contrastive objective $\langle J \rangle$ averaged in each epoch. CHL drives $\langle J \rangle$ to near zero. (c) A set of five 32-component memories used for training. Each pixel represents the volume fraction of one component; memories share several overlapping “on” components. (d) An example trajectory in which an initial composition vector relaxes to memory 2 by minimizing the free energy. (e) Retrieval rate vs. flip ratio for varying numbers of overlapping components. Solid lines show mean and shaded bands show the standard deviation across five samples of sets of five 32-bit memories. Dashed lines show the benchmark results from classical Hopfield network.

2.3 Input-output relations and simple classification

A strength of the Hopfield network is that its all-to-all architecture allows neurons to be flexibly designated as inputs, outputs or hidden units (Figure 3a). This makes the system capable of solving some constrained optimization problems. By clamping subsets of neurons, the network explores a constrained energy landscape, enabling tasks such as learning input-output relations and performing classification. In this section, we demonstrate that CHL can train a liquid-phase mixture to represent input-output relations and perform classification tasks similar to those demonstrated by Zentner et al. using backpropagation [81].

To define the tasks, the components of the phase vector are partitioned into three categories: inputs, outputs, and hidden components $\vec{\phi} = (\vec{\phi}^I, \vec{\phi}^O, \vec{\phi}^H)$. Training samples are drawn from the target input-output relations. In the clamped phase, the input and output components are fixed to the target values while the hidden components are initialized to uniform values with random noise; the system then relaxes to equilibrium. In the free phase, only the input components remain fixed; the output and hidden components are released, and the system relaxes to a new equilibrium. The CHL rules are then applied by contrasting the equilibrium states from the clamped and free phase, shaping the free-energy landscape on

the constraint manifold toward the target output states. In all input-output and classification examples, an additional inert solvent component is included to absorb the remainder of the volume fraction, decoupling the constraint from the task-relevant components and allowing the input-output mapping to be defined independently of the normalization. The astute reader will notice that the quantity being minimized is no longer a measure of the average stability of target phases in the free energy landscape, $\langle J \rangle_P$, but rather the average stability of target phases in constrained (input-clamped) energy landscapes, which we may also write as $\langle J \rangle_P$ but where the distribution P contains more information, as described in Section 4.

As a baseline, we train simple linear relations (Figure 3b), sampling input-output pairs uniformly from the target functions. A single input-output pair without hidden components can represent a linear relation, but the volume-fraction constraint severely limits the accessible slopes. With additional hidden components, the system can represent more complex nonlinear analog functions, including a sigmoid and an inverted parabola (Figure 3b). Error bars reflect the standard deviation over test inputs evaluated with fixed trained parameters.

Next, we train a two-input, two-output system to perform binary classification across several decision boundaries. In biological contexts, classification-like behavior arises in the spatial condensation of regulatory proteins: condensates form at specific genomic loci to promote or suppress gene expression [61]. In our model, the two output components are trained to respond with a high volume fraction to combinations of the two input components. We begin with a simple linear example as shown in Figure 3c. We then test nonlinear classification boundaries including AND, XOR, and circular boundaries. All interaction matrices are provided in Appendix Figure 7. Notably, we found that successful classification required weighted sampling of training examples, with higher sampling density near the decision boundary. In particular, we draw 70% of our training samples from a region defined as being within distance 0.05 from the decision boundary. This suggests that while CHL eliminates the need for an explicit loss function, the trade-off is that the sampling distribution may need to reflect the task structure. These limitations are discussed further in Section 5.

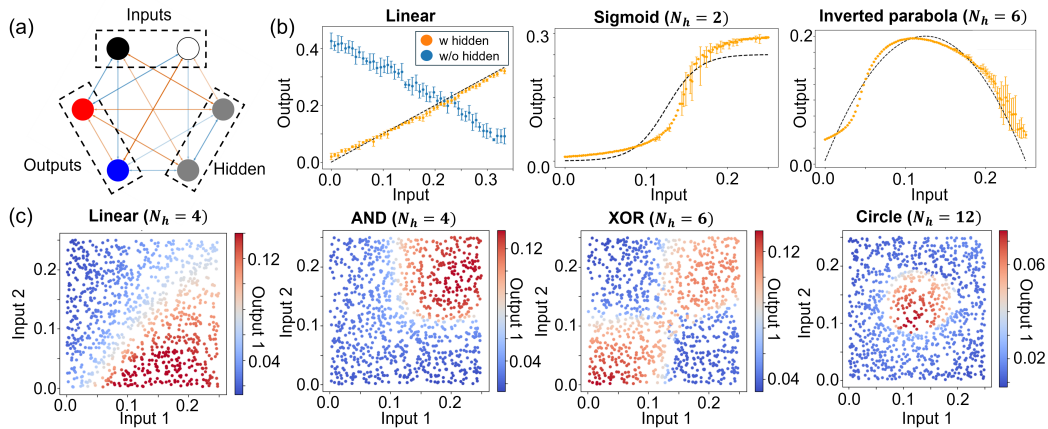


Figure 3 Training low-dimensional input-output functions. (a) Schematic showing the partitioning of components into input, output, and hidden classes. (b) Examples of trained analog input-output relations. The linear case illustrates that without hidden components the system cannot match an arbitrary slope. With hidden components, sigmoid and inverted parabola relations can be trained, demonstrating continuous function representation. Error bars show the standard deviation over test inputs (fixed trained parameters). (c) Trained 2-input, 2-output classification across decision boundaries ranging from linear to nonlinear. In all panels, N_h is the number of hidden species.

2.4 Classification of the MNIST dataset

Hopfield-type networks extend to high-dimensional classification tasks [10]. Here we benchmark the system on MNIST classification. The MNIST dataset consists of images of 28×28 pixels, corresponding to 784 input components at a 1:1 mapping. At this number of components, our integration of model A dynamics exhibits numerical instabilities; we reduce the input to $7 \times 7 = 49$ components by average pooling with a 4×4 kernel. Ten output components represent digits 0 through 9. We add 40 hidden components and one solvent component, bringing the total to 100 components (Figure 4a). CHL training is performed by sampling uniformly from the dataset, with the correct output component clamped to a high volume fraction and all other output components held low. We define a successful classification as one in which the correct output has the highest volume fraction. The best-performing trained system achieves an accuracy of $\sim 75.5\%$, in the ballpark of prior results [16, 81]. The confusion matrix is shown in Figure 4c, with trained parameters in Figure 7c.

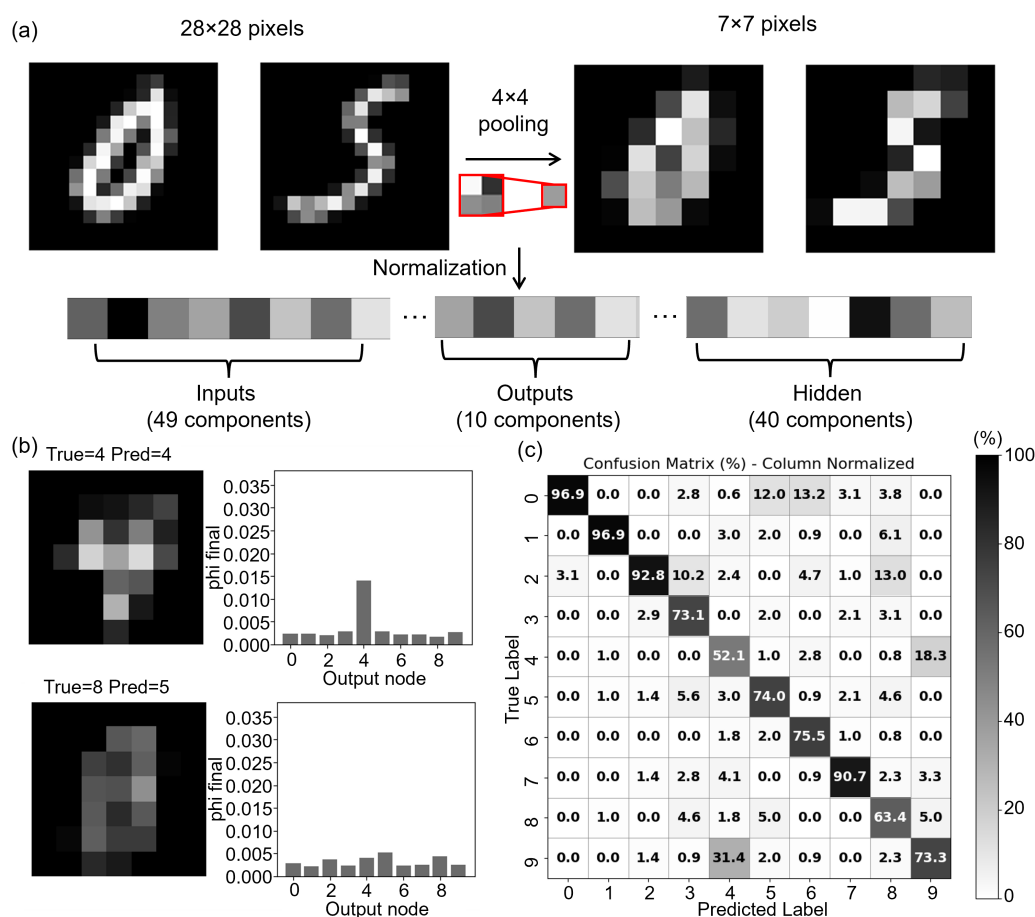


Figure 4 Training a high-dimensional classification task. (a) The MNIST dataset is downsampled to 49 input components. Inputs are normalized to occupy a total volume fraction of 0.5, with the remainder consisting of output, hidden, and solvent components. (b) Classification examples. Digit 4 is correctly classified, whereas digit 8 is incorrectly classified. (c) Confusion matrix for digit classification, where values indicate the probability of the true label given the predicted label.

3 Learning in spatial phase separation systems

In the preceding section, we explored the computational capacity of a well-mixed, single-compartment liquid mixture. Here, we extend the framework to spatially resolved phase separation. The ability to train coexisting spatial phases expands the liquid-phase computing paradigm to encompass pattern formation problems, with implications for understanding intracellular organization and designing programmable materials.

Consider a liquid of volume v with n components. The local composition at position $\mathbf{x}$ and time t is described by $\vec{\phi}(\mathbf{x}, t) = (\phi_1, \phi_2, \dots, \phi_n)$, where $\phi_i(\mathbf{x}, t)$ is the volume fraction of component i , subject to the local normalization constraint $\sum_{i=1}^n \phi_i(\mathbf{x}, t) = 1$. Further, the volume will be closed (not in exchange with a reservoir) and thus the total mass of each species will be conserved, so $\int_v \phi_i d\mathbf{x}$ is constant. Our implementation is in two dimensions with periodic boundaries, but generalization to three dimensions is straightforward.

The total free energy combines the Flory-Huggins mixing energy, omitting the reservoir potential terms, with a Cahn-Hilliard interfacial penalty [47, 60]. The training parameters therefore consist only of the interaction matrix χ_{ij} . We use a fixed and uniform spatial gradient penalty coefficient κ ; more general treatments could allow κ_{ij} to vary between pairs of components. The energy density is $f_{CH} = f_{FH} + \sum_{i=1}^n \frac{\kappa}{2} |\nabla \phi_i|^2$, and the total energy is

$$F_{CH} = \int_v \left(f_{FH} + \sum_{i=1}^n \frac{\kappa}{2} |\nabla \phi_i|^2 \right) d\mathbf{x}, \quad (9)$$

where $\nabla \phi_i = (\frac{\partial \phi_i}{\partial x}, \frac{\partial \phi_i}{\partial y})$ is the spatial gradient of species i . We can define the chemical potential $\mu_{i,CH}(\mathbf{x}, t) = \frac{\delta F_{CH}}{\delta \phi_i(\mathbf{x}, t)} = \sum_{j=1}^n \chi_{ij} \phi_j(\mathbf{x}, t) + 1 + \log \phi_i(\mathbf{x}, t) - \kappa \nabla^2 \phi_i(\mathbf{x}, t)$. Model B [30] gives us the appropriate dynamics for this mass-conserving setting, as has been used in prior spatial LLPS simulations [47, 70, 60]:

$$\frac{\partial \phi_i}{\partial t} = \nabla \cdot \left(M_i \nabla \left(\frac{\delta F_{CH}}{\delta \phi_i} \right) \right). \quad (10)$$

Again setting $M_i = M$ (constant), we obtain $\frac{\partial \phi_i}{\partial t} = M \nabla^2 \left(\frac{\delta F_{CH}}{\delta \phi_i} \right)$, where $\nabla^2 \phi_i = \frac{\partial^2 \phi_i}{\partial x^2} + \frac{\partial^2 \phi_i}{\partial y^2}$ is the Laplacian, or divergence of the gradient. In our results, the conserved quantities are within the range of machine precision.

As in the single-compartment case, a Lagrange multiplier $\lambda(\mathbf{x}, t)$ is introduced to enforce the local normalization constraint $\sum_{i=1}^n \phi_i(\mathbf{x}, t) = 1$. The modified free-energy density becomes $f_{\lambda CH} = f_{CH} + \lambda (\sum_{i=1}^n \phi_i - 1)$, which is integrated over the space to give $F_{\lambda CH}$. The modified dynamics become $\frac{\partial \phi_i}{\partial t} = M \nabla^2 \left(\frac{\delta F_{\lambda CH}}{\delta \phi_i} \right)$, with $\lambda = -\frac{1}{n} \sum_{i=1}^n \mu_{i,CH} = -\langle \mu_{CH} \rangle$. Under these dynamics, the equilibrium condition is $\mu_{k,CH}(\mathbf{x}) - \langle \mu_{CH} \rangle(\mathbf{x}) = C_k$, where C_k is a constant independent of $\mathbf{x}$. A full derivation is given in Appendix A.4.

Analogous to the CHL picture in Section 2.1, we can initialize the spatial system with various properties clamped: for instance, the spatial configuration of particular phases, or the global compositions of the particular components. Ideally, it will relax to the clamped equilibrium $\check{\phi}(\mathbf{x})^{(+)}$, with the corresponding $\check{F}_{CH}^{(+)}$. Next, the clamp is fully or partially released, allowing the system to settle to a new equilibrium $\check{\phi}(\mathbf{x})^{(-)}$ with associated $\check{F}_{CH}^{(-)}$.

As before, our contrastive objective is $J = \check{F}_{CH}^{(+)} - \check{F}_{CH}^{(-)}$. Since the free state has more degrees of freedom, $\check{F}_{CH}^{(+)} \geq \check{F}_{CH}^{(-)}$ so again $J \geq 0$. The CHL rule for the spatial model, derived in Appendix A.5, is:

$$\Delta \chi_{ij} \propto - \int_v \check{\phi}_i^{(+)} \check{\phi}_j^{(+)} d\mathbf{x} + \int_v \check{\phi}_i^{(-)} \check{\phi}_j^{(-)} d\mathbf{x}. \quad (11)$$

This has the same Hebbian structure as the rule for the surface model, integrating $\check{\phi}_i \check{\phi}_j$ pointwise over space. The update promotes interactions between species that co-occur in the target pattern, and decreases them for species that co-occur in the free equilibrium, sculpting the free energy landscape to favor the desired spatial morphology.

3.1 Training for stable coexisting phases

We now demonstrate training of the interaction matrix to stabilize prescribed coexisting phases. This is directly related to the design of synthetic condensates with specified compositions, or to biological pattern formation, where distinct spatial domains must be robustly maintained. Suppose we wish to train a system to stably support a set of m coexisting phases $S = \{\vec{\phi}^\alpha, \vec{\phi}^\beta, \dots\}$. For clamping, we assign a composition $\vec{\phi}(\mathbf{x}) \in S$ to each location $\mathbf{x}$ with uniform probability in all demonstrations below; thus $\int_v \check{\phi}_i^{(+)} \check{\phi}_j^{(+)} = \frac{1}{m} \sum_{\sigma \in \{\alpha, \beta, \dots\}} \vec{\phi}_i^\sigma \vec{\phi}_j^\sigma$. For the free phase of each training epoch, the composition field is initialized uniformly at the mean phase $\vec{\phi}^{\text{average}} = \frac{1}{|S|} \sum_{\sigma \in \{\alpha, \beta, \dots\}} \vec{\phi}^\sigma$ with small additive noise, and evolved under model B dynamics until the free energy falls below a convergence threshold. Since late-stage coarsening is a notably slow process, we consider the phase field to have converged sufficiently once it enters the coarsening regime (the threshold for convergence is provided in Appendix A.1). The CHL update rule is then applied to the interaction matrix. Here the learning aims to ensure that random initial states evolve to an energy similar to the mixture of target phases – there is no direct assessment that a given target state is stable and cannot relax further. In this sense, the training procedure here is disrupting spurious minima in the energy landscape until the correlation statistics of evolved states match the target mixture, making it conceptually more akin to the Daydreaming method for training classical Hopfield networks [58] than to the training procedure we used to justify the learning rule. This discrepancy between training methods is discussed in more detail in Section 4.

We demonstrate spatial CHL with a 32-component system trained to encode five phases with overlapping components (Figure 5). Simulations are halted in the coarsening regime, and so simulation times vary over training. By the end of training, we apply CHL at $t_{\text{train}} \approx 30$; to test, we simulate for longer, with $t_{\text{test}} \approx 70$. To determine the retrieved phases, we map the composition of each position to the target phase with the lowest Euclidean distance, $D_\sigma = \|\vec{\phi}(\mathbf{x}) - \vec{\phi}^\sigma\|_2$, with $\text{Index}(\mathbf{x}) = \arg \min_{\sigma \in \{\alpha, \beta, \dots\}} D_\sigma$, and then average (over space) the positions assigned to the same phase. The system successfully retrieves all five target phases from a disordered initial condition, with the retrieved phases closely matching the prescribed targets. The interaction matrix (Figure 5b) exhibits a rich variety of attractive and repulsive interactions, showing components play different roles to encode five phases. Figure 5c shows the closest phase mapping and energy density f_{CH} of the composition field. The space separates into regions each close to one of the target phases, with low D_σ confirming that each composition is close to a target. For simulation times similar to those used during training, deviations are mainly confined to the domain boundaries where the composition profile is dominated by the interfacial energy penalty κ , while longer simulations develop regions of phase dissimilarity – highlighting the sensitivity of the technique to the “convergence” stopping criterion. Nonetheless, this result, which yields phase separation domains visually comparable to those obtained by an inverse design method [70], demonstrates that the local CHL rule can encode multi-phase associative memory directly into the interaction matrix of a spatially resolved system, providing a route to programmable spatial organization. The [Supplementary Information](#) includes additional results for the spatial model: a time series and spatial distributions of each component for the system in Figure 5; trained outputs of some simpler examples; and a demonstration of associative recall in the nucleation regime.

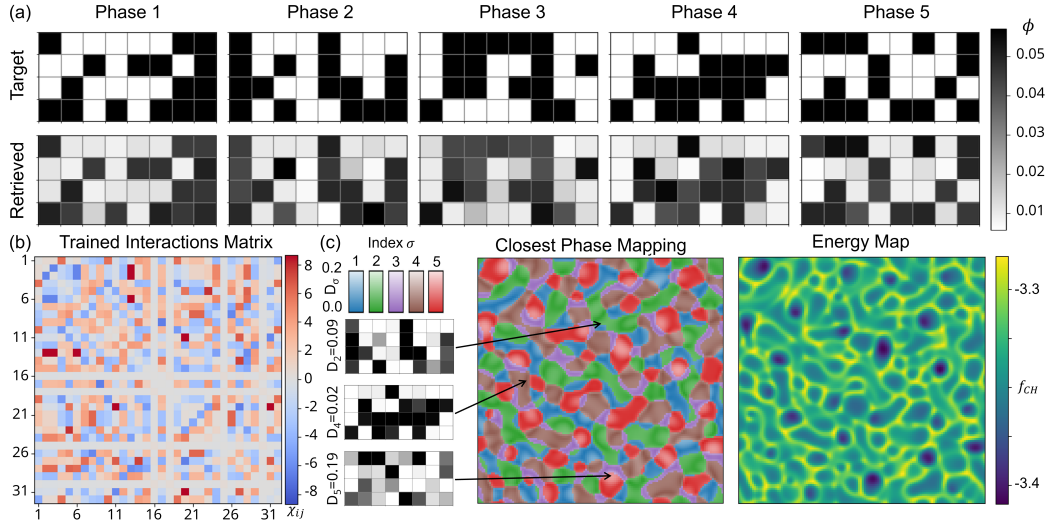


Figure 5 Training for spatial mixtures of target phases. (a) Retrieved phases at t_{test} match the target phases. (b) The trained interaction matrix. (c) The output of a simulation at t_{test} , showing the spatial distribution of the phases. For each point in space $\mathbf{x}$, we calculate the distance D_σ to each of the five target phases; assign the point to the nearest phase; and then color by distance to that phase, with darker colors indicating smaller distances. On the left, we show compositions of three pixels with varying distances to the target phases. On the right, we plot the energy density f_{CH} . Boundaries between phases are further from the minima, and so have higher f_{CH} .

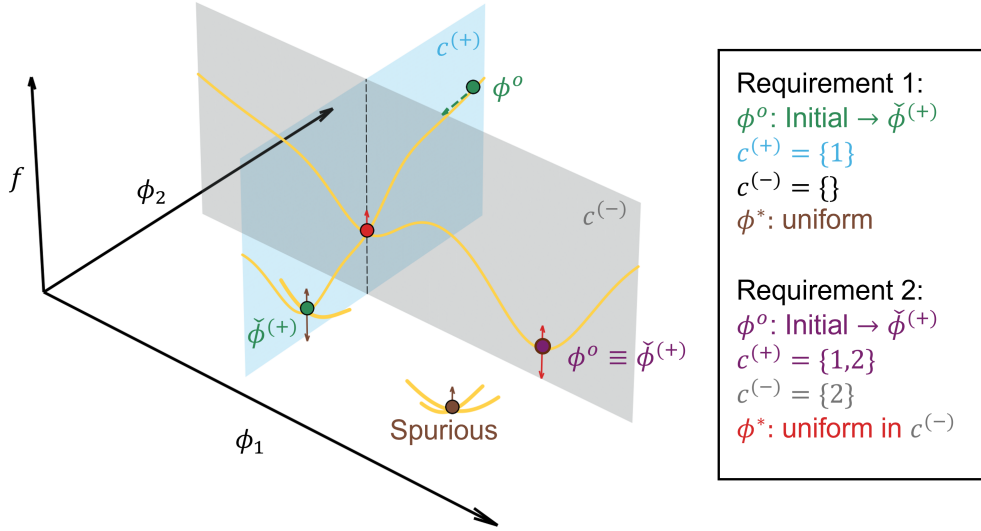
4 Unified learning with positive and negative design requirements

The preceding examples of CHL learning for multicomponent mixtures employed several variations of the method that was formally proven, including differences in what states were initialized and when and how much clamping was applied. To clarify how our proofs do, or do not, apply to these situations, we formulate a generalized framework for how contrastive learning can sculpt energy landscapes according to positive and negative design requirements. In this context, positive design requests that certain states be stable under certain conditions, while negative design requests that certain states not be stable. This is achieved by either pulling the energy down or up, respectively, for those states.

Consider a continuous dynamical system with energy landscape $f(\vec{\phi})$ on coordinate $\vec{\phi}$, so ϕ_i is the coordinate with index $i \in I$. Equilibration corresponds to downhill motion on the landscape. The landscape will be parameterized by model coefficients χ with respect to which we will perform gradient descent to optimize a loss function $\mathcal{L}$ specified below.

A learning task (or energy landscape sculpting task) is defined in terms of samples $s = (\vec{\phi}^o, c^{(+)}, c^{(-)}, \vec{\phi}^*)$, where $\vec{\phi}^o$ is an initial state for positive design, $c^{(+)} \subseteq I$ is a set of target indices being clamped, $c^{(-)} \subset c^{(+)}$ is a less constrained set of indices indicating how much the clamp will be relaxed, and the (typically random) initial state $\vec{\phi}^*$ will be used for negative design. There will be a probability distribution $P(s)$ that governs the selection of desired target states and clamping conditions, the assignment of initial states of hidden species, and the random initial states that should not lead to new spurious minima.

For positive design, define $\check{\phi}^{(+)}$ to be the equilibrium state achieved from $\vec{\phi}^o$ when clamping variables in $c^{(+)}$, while $\check{\phi}^{(-)}$ is the subsequent equilibrium state after relaxing the clamping to $c^{(-)}$. As before, the drop in energy $J(s) = f(\check{\phi}^{(+)}) - f(\check{\phi}^{(-)}) \geq 0$ with equality if the equilibrium is stable under these conditions.



■ **Figure 6** Sculpting an energy landscape. Two types of design requirements are illustrated for a two-dimensional energy landscape f . Samples for requirement 1 chose a random value for the initial ϕ_2 (the “hidden species”) while clamping ϕ_1 to the blue subspace; the positive design criterion is that when the clamp is released, ϕ_1 remains at a local minimum; the negative design criterion is that there are no unconstrained local minima (such as the brown state). Samples for requirement 2 initially clamp both ϕ_1 and ϕ_2 , and then release ϕ_1 so dynamics explores the grey subspace; the positive design criterion is that ϕ^o is a constrained local minimum in the grey subspace; the negative design criterion is that there are no other constrained local minima (such as the red state). Vertical arrows indicate the effects of positive and negative learning. The illustrated energy landscape is at an intermediate stage of learning where the positive design requirements are met but the negative design requirements are not yet met.

For negative design, the goal is to destabilize spurious minima in the $c^{(-)}$ subspace, i.e. those other than the targeted states that are being trained to be stable when $c^{(-)}$ is clamped. Defining $\check{\phi}^{(*,-)}$ to be the equilibrium state evolved from $\vec{\phi}^*$ while clamping $c^{(-)}$, we are interested in the difference $H(s) = f(\check{\phi}^{(+)}) - f(\check{\phi}^{(*,-)})$ that, on average, measures how much lower the energies reached from random initial states are below those achieved by the desired states. The justification for this criterion is subtle and not universal, as discussed below.

Figure 6 illustrates positive and negative design of an energy landscape. It should be clear that with multiple requirements in place, for example many parallel slices through the landscape via clamping, many detailed features of the landscape can be designed, including valleys and pathways, not just the minima.

The unified contrastive learning rule performs gradient descent on the loss function

$$\mathcal{L}(\chi) = \alpha \sum_s P(s) J(s) + (1 - \alpha) \sum_s P(s) H(s) = \alpha \langle J \rangle_P + (1 - \alpha) \langle H \rangle_P \quad (12)$$

where $0 \leq \alpha \leq 1$ is the user-defined balance between positive and negative design requirements. Prior derivations of $\frac{\partial f(\check{\phi})}{\partial \chi_{ij}}$ can be applied without change to obtain

$$\Delta \chi_{ij} \propto -\check{\phi}_i^{(+)} \check{\phi}_j^{(+)} + \alpha \check{\phi}_i^{(-)} \check{\phi}_j^{(-)} + (1 - \alpha) \check{\phi}_i^{(*,-)} \check{\phi}_j^{(*,-)} \quad (13)$$

for sample s chosen from $P(s)$, where a spatial integral is added for the spatial case. In other words, the energies of desired states are pushed down, while the energies of the two kinds of undesired states are pushed up.

The learning examples demonstrated earlier can be interpreted in this context. For training associative memories (Figure 2), sample $s = (\vec{\phi}^o, c^{(+)}, c^{(-)}, \vec{\phi}^*)$ is chosen with $\vec{\phi}^o$ being uniformly one of the five memories, $c^{(+)}$ clamping all 32 species, $c^{(-)}$ clamping none, and $\vec{\phi}^*$ being irrelevant since $\alpha = 1$. For training 2-input classification (Figure 3c), sample s is chosen with $\vec{\phi}^o$ being uniform for non-input species and preferentially near the decision boundary for input species, $c^{(+)}$ clamping the input and output species, $c^{(-)}$ clamping just the input species, and $\vec{\phi}^*$ being irrelevant since again $\alpha = 1$. For training coexisting spatial phases, sample s is chosen with $\vec{\phi}^o$ being (effectively equivalent to) a single spatial distribution with one large domain for each of the five memories, $c^{(+)}$ clamping all species, $c^{(-)}$ clamping none, and $\vec{\phi}^*$ being the uniform average of the memories plus a small bit of noise, since $\alpha = 0$. In all these examples, it is easy to imagine that performance could have been better with an intermediate value of α and, for non-spatial models, $\vec{\phi}^*$ being uniform for unclamped species.

Although the local update rule for $\Delta\chi_{ij}$ does perform stochastic-sampled gradient descent on $\mathcal{L}(\chi)$, how do we justify this loss function in general? Since $J(s) \geq 0$ with equality if and only if the minimum is stable with the lower level of clamping, any training run that successfully reaches near zero should satisfy the positive requirements. However, $H(s)$ may be positive or negative on any given sample, and one can concoct example landscapes and samples such that $\langle H \rangle_P$ is positive or negative as well – for example, $\langle H \rangle_P < 0$ for a landscape with one very deep and narrow target minimum and one very shallow but broad target minimum, and changes to χ that make them more extreme will continue to decrease $\langle H \rangle_P$, consistent with gradient descent but presumably undesired. We can only heuristically observe that for Hopfield networks and for multicomponent liquids, changes that deepen an energy basin tend to also broaden it, while changes that fill in a basin also make it narrower – in which case we expect the negative design learning to successfully eliminate spurious minima. In that case, we can observe that $\langle H \rangle_P = 0$ if all spurious minima have been eliminated and the basin size of target minima is in proportion to its probability $P(s)$ for each clamping condition; this appears to us as a plausible but not guaranteed outcome. The negative design principle here is closely related to the Daydreaming method for training classical Hopfield networks to reduce spurious minima [58], and thus the general success of that method is promising for ours.

5 Discussion

The CHL learning rule was formalized in our surface and spatial multicomponent LLPS models to carve energy landscapes achieving several tasks. Encoding target compositions as stable minima of the surface-model Flory-Huggins energy connects our work to a broader literature of associative recall via energy minimization, first studied in abstract Hopfield neural networks, to the biologically-relevant process of multicomponent LLPS. Computation of low-dimensional input-output functions demonstrates that surface condensation may play an information-processing role even in systems with few unique components. Classification of the MNIST dataset in the surface model affirms the neural properties of multicomponent LLPS: many unique components aggregate information via relatively weak signals in order to establish an output. Finally, training of stable coexisting phases in the spatial Cahn-Hilliard framework provides validation beyond the surface model of phases with overlapping components, incorporating spatial physical considerations such as the phases' interfacial penalties. These tasks were achieved by the same learning framework: specification of a contrastive objective function J that describes the difference in energy minima between more- and less-constrained systems. Our demonstration of these tasks across two distinct

multicomponent LLPS models illustrates how one specifies various target behaviors by contrastive objective functions J and establishes CHL as a versatile learning rule, and illustrates that memory retrieval, classification, analog function computation, and phase coexistence design may all be treated as instances of energy-landscape sculpting.

The standard approach in modern machine learning is to specify and minimize a (potentially arbitrarily complex) loss function for specifying how well the system state or output matches a learning target. For example, backpropagation computes gradients of the loss function to send error signals through the network, which may be less biologically plausible [44]. A learning rule closer to CHL is equilibrium propagation [56], which is a local contrastive learning rule that compares the difference between a system’s energy minima and the minima of an energy function “nudged” by the cost function. While these two methods may require different physical implementations for different learning rules, CHL requires only a physical implementation of clamping, shifting the burden to designing clamped and unclamped steps whose energy difference encodes the desired behavior.

The computational potential of multicomponent liquid mixtures demonstrated in our work inspires further questions. In the surface model, we show an example of memory storage and retrieval with five memories and 32 components. Empirically, storage capacity in Hopfield networks scales linearly with increasing number of units, while studies of the liquid systems suggest poorer scaling [37, 59, 12]. A feature lacking in prior design of coexisting phases is the ability to train interactions for hidden components, which may help stabilize target phases. Our demonstration of linear and nonlinear function computation also raises scaling questions. How does the number of required hidden components scale with the complexity of the decision boundaries? Zentner et al. [81] argue that, in the surface model, each extra hidden component adds an extra hidden linear decision boundary, and from this fact and other assumptions depict the potential for the model as a universal decision boundary approximator. Our one-input analog function tasks (the linear, sigmoidal, and inverted parabolic curves) present a new class of tasks in the surface model for which further analysis may explain hidden component scaling and potential universality of the model. Beyond these particular input-output tasks, a general theory that articulates the potential for sculpting arbitrary target energy landscapes via CHL may encompass memory storage, decision boundaries, function computation, and other behaviors in a unified framework.

The spatial model will enable investigation of pattern formation tasks beyond the phase coexistence we explored here. The design of a stable spatial pattern as a system equilibrium confers self-healing: equilibrium relaxation restores the pattern after perturbations. Although there is a rich literature on spatial pattern formation in reaction-diffusion systems [41, 43], which must be out-of-equilibrium, the formation of stable patterns at equilibrium requires interfacial penalties captured by the Cahn-Hilliard model. Interactions for complex stable spatial patterns beyond coexisting phases could be trained via CHL. We used a single constant κ for the gradient penalty coefficient; since κ is, physically, a pairwise term κ_{ij} , including it in the design space could potentially enable enriched spatial patterning. Our coexisting phases in the spatial model only loosely evoke Hopfield’s original picture of memory storage and associative recall. Extension of classification and analog computation tasks illustrated in the surface model may extend naturally to the spatial model given a specification of spatial input and output representation. Questions related to scaling of function complexity with hidden component scaling and universality apply here just as well as the surface model case.

Finally, we discuss the prospects for physical implementation of our models in real molecular systems. Advances in molecular programming have greatly improved our ability to design molecular interactions with precision [27, 6]. Specific phase morphologies such as core-

shell structures have been shown to be rationally designable [49, 38, 2, 69, 63, 55]. Recently, multicomponent liquid condensates based on DNA nanostars have been demonstrated with up to nine components [15], approaching the regime where neural computation in liquid systems may become experimentally accessible. However, gaps remain between theory and experiment: concrete mappings between theoretical interaction parameters and real molecular design rules have yet to be established. Further, our model makes two key assumptions that potentially hinder physical implementation. First, our mean-field free energy assumes that the species are well-mixed in each volume. The degree to which this assumption may be violated can be checked post-training, e.g. via fine-grained lattice model simulations. Second, our single-compartment surface model assumes that the reservoir maintains chemical potentials μ_i for any learned interactions χ_{ij} . Explicit construction of such a reservoir that itself does not internally phase separate is nontrivial: Teixeira et al. [12] propose n chemical reservoirs, one for each component, and Zentner et al. [81] propose a composition for a single reservoir. An alternative approach towards physical realization of the surface model results is to adapt them to the Cahn-Hilliard framework discussed in Section 3, which reflects a simpler physical set-up without a reservoir. However, in physical implementations of the Cahn-Hilliard framework, the gradient penalty coefficient κ will likely depend on microscopic interaction energies G_{ij} , and thus sufficient modeling may require the determination of pair-dependent κ_{ij} parameters. Further, our model B dynamics do not capture other real physical phenomena, such as the bulk motion of droplets or fluids, nor large-scale Brownian effects. Our learning method requires only knowledge of equilibrium states, and is therefore compatible with non-deterministic dynamics. The Cahn-Hilliard equation may be modified to include noise terms (Langevin dynamics, [21]); this, alongside other extensions to the dynamics to improve physical realism, may help with translating from simulation parameters to real molecules.

The physical implementation of the CHL learning itself requires three key parts. First is the ability to clamp components' volume fractions. In the surface model, an example clamping mechanism would be a programmable semi-permeable membrane designed to allow passage of only a specified subset of the components, with the ability for that subset to be changed over time. Second is the ability to modify interaction parameters χ_{ij} and/or reservoir potentials $\mu_{i,res}$. The locality condition – that $d\chi_{ij}/dt$ depends only on statistics about components i and j – eases the requirements on physical implementation. For example, as suggested by Stern et al. [67], if an $i-j$ mediator species' concentration induces an effective interaction $\tilde{G}_{ij}$ between components i and j , production of that mediator via proximity-based ligation under mass-action kinetics could produce a Hebbian update of the form $d\tilde{G}_{ij}/dt \propto \phi_i(t)\phi_j(t)$, where $\phi_i(t)$ is the volume fraction at time t . Third is the switching between more- and less-clamped conditions with the associated switch between Hebbian and anti-Hebbian parameter updates. An implicit-memory mechanism has been proposed to enable contrastive switching between Hebbian and anti-Hebbian phases in molecular systems [25], but a clear clamping mechanism for implementing this approach has yet to be proposed. Advancements in these three key tasks and their integration could yield a generalized framework for purely physical learning that would allow exploration of high-dimensional LLPS beyond the limited accuracy of *in silico* models, and would cement energy-landscape sculpting as a physical mechanism for biomolecular learning.

References

- 1 David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985. doi:10.1207/S15516709C0G0901_7.
- 2 Siddharth Agarwal, Dino Osmanovic, Mahdi Dizani, Melissa A Klocke, and Elisa Franco. Dynamic control of DNA condensation. *Nature Communications*, 15(1):1915, 2024.

- 3 Blaise Agüera y Arcas, Jyrki Alakuijala, James Evans, Ben Laurie, Alexander Mordvintsev, Eyvind Niklasson, Ettore Randazzo, and Luca Versari. Computational life: How well-formed, self-replicating programs emerge from simple interaction. *arXiv:2406.19108*, 2024.
- 4 Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55(14):1530, 1985.
- 5 Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Statistical mechanics of neural networks near saturation. *Annals of Physics*, 173(1):30–67, 1987.
- 6 Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.
- 7 Jack E Baldwin and Hans Krebs. The evolution of metabolic cycles. *Nature*, 291(5814):381–382, 1981.
- 8 Salman F Banani, Hyun O Lee, Anthony A Hyman, and Michael K Rosen. Biomolecular condensates: organizers of cellular biochemistry. *Nature Reviews Molecular Cell Biology*, 18(5):285–298, 2017.
- 9 Naama Barkai and Stan Leibler. Robustness in simple biochemical networks. *Nature*, 387(6636):913–917, 1997.
- 10 MA Belyaev and AA Velichko. Classification of handwritten digits using the Hopfield network. In *IOP Conference Series: Materials Science and Engineering*, volume 862, page 052048. IOP Publishing, 2020.
- 11 James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL: <http://github.com/google/jax>.
- 12 Rodrigo Braz Teixeira, Giorgio Carugno, Izaak Neri, and Pablo Sartori. Liquid Hopfield model: Retrieval and localization in multicomponent liquid mixtures. *Proceedings of the National Academy of Sciences*, 121(48):e2320504121, 2024.
- 13 Jehoshua Bruck and Vwani P Roychowdhury. On the number of spurious memories in the Hopfield model (neural network). *IEEE Transactions on Information Theory*, 36(2):393–397, 1990.
- 14 Miguel A Carreira-Perpinan and Geoffrey Hinton. On contrastive divergence learning. In *International Workshop on Artificial Intelligence and Statistics*, pages 33–40. PMLR, 2005.
- 15 Aria S Chaderjian, Sam Wilken, and Omar A Saleh. Diverse, distinct, and densely packed DNA nanostar droplets. *Proceedings of the National Academy of Sciences*, 123(7):e2523462123, 2026.
- 16 Cameron Chalk, Salvador Buse, Krishna Shrinivas, Arvind Murugan, and Erik Winfree. Learning and inference in a lattice model of multicomponent condensates. In *30th International Conference on DNA Computing and Molecular Programming (DNA 30)*, volume 314, pages 5:1–5:24, 2024. doi:10.4230/LIPIcs.DNA.30.5.
- 17 Fan Chen and William M Jacobs. Programmable phase behavior in fluids with designable interactions. *The Journal of Chemical Physics*, 158(21), 2023.
- 18 Long Qing Chen and Jie Shen. Applications of semi-implicit Fourier-spectral method to phase field equations. *Computer Physics Communications*, 108(2-3):147–158, 1998.
- 19 Kevin M Cherry and Lulu Qian. Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. *Nature*, 559(7714):370–376, 2018.
- 20 Kevin M Cherry and Lulu Qian. Supervised learning in DNA neural networks. *Nature*, 645(8081):639–647, 2025.
- 21 HE Cook. Brownian motion in spinodal decomposition. *Acta Metallurgica*, 18(3):297–306, 1970.
- 22 David Doty, Jack H Lutz, Matthew J Patitz, Robert T Schweller, Scott M Summers, and Damien Woods. The tile assembly model is intrinsically universal. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 302–310. IEEE, 2012. doi:10.1109/FOCS.2012.76.

- 23 Constantine G Evans and Erik Winfree. Physical principles for DNA tile self-assembly. *Chemical Society Reviews*, 46(12):3808–3829, 2017.
- 24 Constantine Glen Evans, Jackson O’Brien, Erik Winfree, and Arvind Murugan. Pattern recognition in the nucleation kinetics of non-equilibrium self-assembly. *Nature*, 625(7995):500–507, 2024.
- 25 Martin J Falk, Adam T Strupp, Benjamin Scellier, and Arvind Murugan. Temporal contrastive learning through implicit non-equilibrium memory. *Nature Communications*, 16(1):2163, 2025.
- 26 Paul J Flory. Thermodynamics of high polymer solutions. *The Journal of Chemical Physics*, 10(1):51–61, 1942.
- 27 Mark E Fornace, Jining Huang, Cody T Newman, Avinash Nanjundiah, Nicholas J Porubsky, Marshall B Pierce, and Niles A Pierce. NUPACK: Computational nucleic acid analysis and design. *ACS Synthetic Biology*, 15(4):1426–1441, 2026.
- 28 Alireza Goudarzi, Matthew R Lakin, and Darko Stefanovic. DNA reservoir computing: a novel molecular computing approach. In *International Workshop on DNA-Based Computers*, pages 76–89. Springer, 2013. doi:10.1007/978-3-319-01928-4_6.
- 29 Geoffrey E Hinton. Deterministic Boltzmann learning performs steepest descent in weight-space. *Neural Computation*, 1(1):143–150, 1989. doi:10.1162/NECO.1989.1.1.143.
- 30 Pierre C Hohenberg and Bertrand I Halperin. Theory of dynamic critical phenomena. *Reviews of Modern Physics*, 49(3):435, 1977.
- 31 John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.
- 32 John J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81(10):3088–3092, 1984.
- 33 John J Hopfield and David W Tank. “Neural” computation of decisions in optimization problems. *Biological Cybernetics*, 52(3):141–152, 1985.
- 34 Maurice L Huggins. Solutions of long chain compounds. *The Journal of Chemical Physics*, 9(5):440–440, 1941.
- 35 Anthony A Hyman, Christoph A Weber, and Frank Jülicher. Liquid-liquid phase separation in biology. *Annual Review of Cell and Developmental Biology*, 30:39–58, 2014.
- 36 William M Jacobs and Daan Frenkel. Predicting phase behavior in multicomponent mixtures. *The Journal of Chemical Physics*, 139(2):024108, 2013.
- 37 William M Jacobs and Daan Frenkel. Phase transitions in biological systems with many components. *Biophysical Journal*, 112(4):683–691, 2017.
- 38 Byoung-jin Jeon, Dan T Nguyen, and Omar A Saleh. Sequence-controlled adhesion and microemulsification in a two-phase system of DNA liquid droplets. *The Journal of Physical Chemistry B*, 124(40):8888–8895, 2020.
- 39 Hawoong Jeong, Sean P Mason, A-L Barabási, and Zoltan N Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, 2001.
- 40 Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021.
- 41 Shigeru Kondo and Takashi Miura. Reaction-diffusion model as a framework for understanding biological pattern formation. *Science*, 329(5999):1616–1620, 2010.
- 42 Matthew R Lakin and Darko Stefanovic. Supervised learning in adaptive DNA strand displacement networks. *ACS Synthetic Biology*, 5(8):885–897, 2016.
- 43 Inhoo Lee, Salvador Buse, and Erik Winfree. Differentiable programming of indexed chemical reaction networks and reaction-diffusion systems. In *31st International Conference on DNA Computing and Molecular Programming (DNA 31)*, pages 4:1–4:23, 2025. doi:10.4230/LIPICs.DNA.31.4.
- 44 Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.
- 45 Fritz Lipmann. The ATP–phosphate cycle. In *Current Topics in Cellular Regulation*, volume 18, pages 301–311. Elsevier, 1981.

- 46 Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv:1711.05101*, 2017.
- 47 Sheng Mao, Derek Kuldinow, Mikko P Haataja, and Andrej Košmrlj. Phase behavior and morphology of multicomponent liquid mixtures. *Soft Matter*, 15(6):1297–1311, 2019.
- 48 Javier R Movellan. Contrastive Hebbian learning in the continuous Hopfield model. In *Connectionist Models: Proceedings of the 1990 Summer School*, pages 10–17. Morgan Kaufmann, 1991.
- 49 Dan T Nguyen, Byoung-jin Jeon, Gabrielle R Abraham, and Omar A Saleh. Length-dependence and spatial structure of DNA partitioning into a DNA liquid. *Langmuir*, 35(46):14849–14854, 2019.
- 50 Shu Okumura, Guillaume Gines, Nicolas Lobato-Dauzier, Alexandre Baccouche, Robin Deteix, Teruo Fujii, Yannick Rondelez, and Anthony J Genot. Nonlinear decision-making with enzymatic neural networks. *Nature*, 610(7932):496–501, 2022.
- 51 Jacob Parres-Gold, Matthew Levine, Benjamin Emert, Andrew Stuart, and Michael B Elowitz. Contextual computation by competitive protein dimerization networks. *Cell*, 188(7):1984–2002, 2025.
- 52 Lulu Qian, David Soloveichik, and Erik Winfree. Efficient Turing-universal computation with DNA polymers. In *International Workshop on DNA-Based Computers*, pages 123–140. Springer, 2010. doi:10.1007/978-3-642-18305-8_12.
- 53 Lulu Qian, Erik Winfree, and Jehoshua Bruck. Neural network computation with DNA strand displacement cascades. *Nature*, 475(7356):368–372, 2011. doi:10.1038/NATURE10262.
- 54 Yicheng Qiang, Chengjie Luo, and David Zwicker. Scaling laws for phase coexistence in multicomponent mixtures. *Physical Review Research*, 7(4):043008, 2025.
- 55 Yusuke Sato, Tetsuro Sakamoto, and Masahiro Takinoue. Sequence-based engineering of dynamic functions of micrometer-sized DNA droplets. *Science Advances*, 6(23):eaba3471, 2020.
- 56 Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, 11:24, 2017. doi:10.3389/FNCOM.2017.00024.
- 57 Nadrian C. Seeman. *Structural DNA Nanotechnology*. Cambridge University Press, Cambridge, United Kingdom, 2016.
- 58 Ludovica Serricchio, Dario Bocchi, Claudio Chilin, Raffaele Marino, Matteo Negri, Chiara Cammarota, and Federico Ricci-Tersenghi. Daydreaming Hopfield networks and their surprising effectiveness on correlated data. *Neural Networks*, 186:107216, 2025. doi:10.1016/J.NEUNET.2025.107216.
- 59 Krishna Shrinivas and Michael P Brenner. Multiphase coexistence capacity in complex fluids. *bioRxiv:2022.10.19.512909*, 2022.
- 60 Krishna Shrinivas and Michael P Brenner. Phase separation in fluids with many interacting components. *Proceedings of the National Academy of Sciences*, 118(45):e2108551118, 2021.
- 61 Krishna Shrinivas, Benjamin R Sabari, Eliot L Coffey, Isaac A Klein, Ann Boija, Alicia V Zamudio, Jurian Schuijers, Nancy M Hannett, Phillip A Sharp, Richard A Young, et al. Enhancer features that drive formation of transcriptional condensates. *Molecular Cell*, 75(3):549–561, 2019.
- 62 Friedrich C Simmel, Bernard Yurke, and Hari R Singh. Principles and applications of nucleic acid strand displacement reactions. *Chemical Reviews*, 119(10):6326–6369, 2019.
- 63 Karuna Skipper and Shelley FJ Wickham. DNA nanostars that self-assemble into core-shell condensate microdroplets. *Nanoscale Horizons*, 2026.
- 64 David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393–5398, 2010.
- 65 Tianqi Song and Lulu Qian. Heat-rechargeable computation in DNA logic circuits and neural networks. *Nature*, 646(8084):315–322, 2025.

- 66 Ulrich Stelzl, Uwe Worm, Maciej Lalowski, Christian Haenig, Felix H Brembeck, Heike Goehler, Martin Stroedicke, Martina Zenkner, Anke Schoenherr, Susanne Koeppen, et al. A human protein-protein interaction network: a resource for annotating the proteome. *Cell*, 122(6):957–968, 2005.
- 67 Menachem Stern and Arvind Murugan. Learning without neurons in physical systems. *Annual Review of Condensed Matter Physics*, 14(1):417–441, 2023.
- 68 Jonathan Stricker, Tobias Falzone, and Margaret L Gardel. Mechanics of the F-actin cytoskeleton. *Journal of Biomechanics*, 43(1):9–14, 2010.
- 69 Diana A Tanase, Dino Osmanović, Roger Rubio-Sánchez, Layla Malouf, Elisa Franco, and Lorenzo Di Michele. Internal phase separation in synthetic DNA condensates. *Advanced Science*, 12(41):e06275, 2025.
- 70 Rodrigo Braz Teixeira, Izaak Neri, and Pablo Sartori. Metastable phase separation and information retrieval in multicomponent mixtures. *arXiv:2509.10705*, 2025.
- 71 Gašper Tkačik and William Bialek. Information processing in living systems. *Annual Review of Condensed Matter Physics*, 7(1):89–117, 2016.
- 72 Hirotake Udono, Jing Gong, Yusuke Sato, and Masahiro Takinoue. DNA droplets: intelligent, dynamic fluid. *Advanced Biology*, 7(3):2200180, 2023.
- 73 Jan van den Berg. The most general framework of continuous Hopfield neural networks. In *Proceedings of International Workshop on Neural Networks for Identification, Control, Robotics and Signal/Image Processing*, pages 92–100. IEEE, 1996.
- 74 Jan van den Berg. Generalized Hopfield networks for constrained optimization. Technical Report EUR-FEW-CS-98-04, Erasmus School of Economics, Erasmus University Rotterdam, 1998. URL: <http://hdl.handle.net/1765/761>.
- 75 Christian Waltermann and Edda Klipp. Information theory based approaches to cellular signaling. *Biochimica et Biophysica Acta (BBA)-General Subjects*, 1810(10):924–932, 2011.
- 76 Erik Winfree. Algorithmic self-assembly of DNA: Theoretical motivations and 2D assembly experiments. *Journal of Biomolecular Structure and Dynamics*, 17(sup1):263–270, 2000.
- 77 Erik Winfree, Xiaoping Yang, and Nadrian C Seeman. Universal computation via self-assembly of DNA: Some theory and experiments. In *DNA Based Computers II*, volume 44 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science*, pages 191–213. American Mathematical Society, 1998.
- 78 Damien Woods, David Doty, Cameron Myhrvold, Joy Hui, Felix Zhou, Peng Yin, and Erik Winfree. Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. *Nature*, 567(7748):366–372, 2019. doi:10.1038/S41586-019-1014-9.
- 79 Xiaohui Xie and H Sebastian Seung. Equivalence of backpropagation and contrastive Hebbian learning in a layered network. *Neural Computation*, 15(2):441–454, 2003. doi:10.1162/089976603762552988.
- 80 Wataru Yahiro, Nathanael Aubert-Kato, and Masami Hagiya. A reservoir computing approach for molecular computing. In *Artificial Life Conference Proceedings*, pages 31–38. MIT Press, 2018. doi:10.1162/ISAL_A_00013.
- 81 Aidan Zentner, Ethan V Halingstad, Cameron Chalk, Michael P Brenner, Arvind Murugan, Erik Winfree, and Krishna Shrinivas. Information processing driven by multicomponent surface condensates. *arXiv:2509.08100*, 2025.
- 82 David Yu Zhang and Georg Seelig. Dynamic DNA nanotechnology using strand-displacement reactions. *Nature Chemistry*, 3(2):103–113, 2011.
- 83 David Zwicker and Liedewij Laan. Evolved interactions stabilize many coexisting phases in multicomponent liquids. *Proceedings of the National Academy of Sciences*, 119(28):e2201250119, 2022.

A Appendix

A.1 Methods for numerical integration and training

Details of the numerical methods we used to integrate model A and model B dynamics and train the parameters are described below. Our simulations are written in JAX, use Diffrax solvers [11, 40], and follow integration schemes inspired by prior studies of multicomponent phase separation [47, 60, 81].

Our parameter initializations and training configurations are as follows. For Figure 2 (memory retrieval), the interaction matrix is initialized with a Hebbian product over target phase compositions, and the chemical potentials are initialized as 0. We train with a learning rate of 0.1 for 500-1000 epochs. For Figures 3 and 4 (classification), the interaction matrix is initialized by sampling uniformly in $[-10, 10]$, with diagonal terms, interactions between the input components, and all the solvent interactions set to 0 and kept at 0 throughout. The chemical potentials are initialized at 0, and input and solvent chemical potentials are kept at 0 throughout. We use a learning rate of 0.02, with 300-500 samples per epoch for 500-2000 epochs. For Figure 5, we use a learning rate of 0.1 for 25-100 epochs. The interaction matrix is again initialized with a Hebbian rule.

In all cases, the parameters are updated according to the CHL rule derived in the main text. The actual parameter θ updates in each epoch are the average $\frac{\partial J}{\partial \theta}$ gradients over the samples in that epoch, multiplied by a discrete scaling factor (i.e., stochastic gradient descent). The AdamW optimizer from Optax [11] is used throughout to improve training stability [46], and χ_{ij} interactions are clipped to a maximum absolute value of 25.

All differential equations are solved numerically with finite difference methods, using discretized time, and also discretized space in the PDE case. Mobility $M = 1.0$ is used for all simulations. For the spatial simulations, we use $\kappa = 5.0$. The spatial simulations are determined to be converged and halted when the maximum step-to-step absolute change of all components at all positions is below 10^{-6} , or the number of simulation steps exceeds 2×10^5 . [Source code and notebooks](#) are available on GitHub.

A.2 Model A dynamics in surface model with Lagrange multipliers

The single-compartment surface model treats the multicomponent liquid as a well-mixed system described by a composition vector $\vec{\phi}$ and the Flory-Huggins energy $f_{FH}(\vec{\phi})$. Standard model A dynamics [30] take the form $\frac{d\phi_i}{dt} = -M_i \frac{\partial f_{FH}}{\partial \phi_i}$, where M_i is the mobility and for simplicity we choose $M_i = M$, a positive constant for all species. To enforce the constraint that $\sum_{i=1}^n \phi_i = 1$ while preserving the gradient-flow structure of the dynamics, we introduce a Lagrange multiplier λ and the modified free energy $f_{\lambda FH} = f_{FH} + \lambda(\sum_{i=1}^n \phi_i - 1)$, with λ determined dynamically to ensure the constraint (in which case $f_{\lambda FH} = f_{FH}$). The modified dynamics then become

$$\frac{d\phi_i}{dt} = -M \frac{\partial f_{\lambda FH}}{\partial \phi_i} = -M(\mu_{i,FH} + \lambda) \quad (14)$$

where we define $\mu_{i,FH} = \frac{\partial f_{FH}}{\partial \phi_i} = 1 + \log \phi_i + \sum_{j=1}^n \chi_{ij} \phi_j - \mu_{i,res}$ as the chemical potential of species i . Applying the constraint $\sum_{i=1}^n \frac{d\phi_i}{dt} = 0$, we can solve for λ :

$$\lambda = -\frac{1}{n} \sum_{i=1}^n \mu_{i,FH} = -\langle \mu_{FH} \rangle. \quad (15)$$

These dynamics guarantee a monotonic decrease of the free energy:

$$\begin{aligned} \frac{df_{\lambda FH}}{dt} &= \sum_{i=1}^n \frac{\partial f_{\lambda FH}}{\partial \phi_i} \cdot \frac{d\phi_i}{dt} = - \sum_{i=1}^n (\mu_{i,FH} - \langle \mu_{FH} \rangle) \cdot M(\mu_{i,FH} - \langle \mu_{FH} \rangle) \\ &= -M \sum_{i=1}^n (\mu_{i,FH} - \langle \mu_{FH} \rangle)^2 \leq 0. \end{aligned} \quad (16)$$

The energy converges only when the system reaches the equilibrium condition $\frac{d\phi_i}{dt} = 0$, i.e., $\mu_{i,FH} = \langle \mu_{FH} \rangle$ or $\mu_{i,FH} + \lambda = 0$.

For numerical implementation, the state of the system at t is captured by $\vec{\phi}^t = (\phi_1, \phi_2, \dots, \phi_n)^t$, and the total system volume $v = 1$ in dimensionless units. Our model A dynamics occur in a space of transformed variables $x_i = \log \phi_i$, with $\phi_i = e^{x_i}$ recovered at each step, following Zentner et al. [81]. This logarithmic reparameterization enforces positivity and improves numerical stability near the boundary of the simplex. We perform numerical integration using the Dopri5 adaptive 4th/5th-order Runge-Kutta solver from DiffraX.

A.3 CHL rule for reservoir potential in the single-compartment case

We derive the CHL update rule for the reservoir chemical potential $\mu_{i,res}$, following the approach used for learning the interaction matrix χ_{ij} described in the main text. Recall that the contrastive objective is $J = \check{f}_{FH}^{(+)} - \check{f}_{FH}^{(-)}$, and so we seek $\partial J / \partial \mu_{i,res}$ to learn $\mu_{i,res}$.

We begin by expanding the derivative of the modified free energy with respect to $\mu_{i,res}$:

$$\frac{\partial \check{f}_{FH}}{\partial \mu_{i,res}} = \frac{\partial (\sum_{k,l=1}^n \frac{1}{2} \check{\phi}_k \chi_{kl} \check{\phi}_l + \sum_{k=1}^n \check{\phi}_k \log \check{\phi}_k - \sum_{k=1}^n \mu_{k,res} \check{\phi}_k)}{\partial \mu_{i,res}}. \quad (17)$$

Differentiating the free energy density gives

$$\frac{\partial \check{f}_{FH}}{\partial \mu_{i,res}} = -\check{\phi}_i + \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \mu_{i,res}} (1 + \log \check{\phi}_k + \sum_{l=1}^n \chi_{kl} \check{\phi}_l - \mu_{k,res}) = -\check{\phi}_i + \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \mu_{i,res}} \mu_{k,FH}. \quad (18)$$

As before, we note that at equilibrium $\mu_{k,FH} = \langle \mu_{FH} \rangle$ and by differentiating the constraint $\sum_{k=1}^n \check{\phi}_k = 1$ to obtain $\sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \mu_{i,res}} = 0$, we can see that

$$\frac{\partial \check{f}_{FH}}{\partial \mu_{i,res}} = -\check{\phi}_i. \quad (19)$$

The gradient of the contrastive objective is then simply

$$\frac{\partial \check{J}}{\partial \mu_{i,res}} = \frac{\partial \check{f}_{FH}^{(+)}}{\partial \mu_{i,res}} - \frac{\partial \check{f}_{FH}^{(-)}}{\partial \mu_{i,res}} = -\check{\phi}_i^{(+)} + \check{\phi}_i^{(-)}, \quad (20)$$

and the CHL update rule for the reservoir chemical potential is $\Delta \mu_{i,res} \propto \check{\phi}_i^{(+)} - \check{\phi}_i^{(-)}$.

This has the same Hebbian structure as the χ_{ij} update rule: the reservoir potential for species i is increased when the target (clamped) composition of that species exceeds its free equilibrium value, pulling the energy landscape toward the desired attractor.

A.4 Spatial phase separation dynamics with Lagrange multipliers

Similar to the surface model case, a Lagrange multiplier $\lambda(\mathbf{x}, t)$ is introduced to enforce the local normalization constraint $\sum_{i=1}^n \phi_i(\mathbf{x}, t) = 1$. The modified free-energy density is

$$\begin{aligned} f_{\lambda CH} &= f_{CH} + \lambda \left(\sum_{i=1}^n \phi_i - 1 \right) \\ &= \frac{1}{2} \sum_{i,j=1}^n \phi_i \chi_{ij} \phi_j + \sum_{i=1}^n \phi_i \log \phi_i + \sum_{i=1}^n \frac{\kappa}{2} |\nabla \phi_i|^2 + \lambda \left(\sum_{i=1}^n \phi_i - 1 \right). \end{aligned} \quad (21)$$

The modified chemical potential of species i is then

$$\mu_{i,\lambda CH} = \frac{\partial f_{\lambda CH}}{\partial \phi_i} = \frac{\partial f_{CH}}{\partial \phi_i} + \frac{\partial \lambda (\sum_{i=1}^n \phi_i - 1)}{\partial \phi_i} = \mu_{i,CH} + \lambda. \quad (22)$$

Applying the constraint $\sum_{i=1}^n \phi_i = 1$, which requires $\sum_{i=1}^n \frac{\partial \phi_i}{\partial t} = 0$, we obtain

$$\sum_{i=1}^n \frac{\partial \phi_i}{\partial t} = M \sum_{i=1}^n \nabla^2 \mu_{i,\lambda CH} = M \sum_{i=1}^n \nabla^2 (\mu_{i,CH} + \lambda) = M \nabla^2 \left(\sum_{i=1}^n \mu_{i,CH} + n\lambda \right) = 0. \quad (23)$$

Under periodic or no-flux boundary conditions, this reduces to

$$\sum_{i=1}^n \mu_{i,CH} + n\lambda = \gamma. \quad (24)$$

Note that any choice of constant γ yields the same dynamics, since it is spatially uniform and vanishes under the Laplacian ∇^2 . The choice $\gamma = 0$ is therefore made without loss of generality, allowing us to solve for λ :

$$\lambda = \frac{-\sum_{i=1}^n \mu_{i,CH}}{n} = -\langle \mu_{CH} \rangle. \quad (25)$$

Here $\lambda(\mathbf{x}, t)$ is a local field, not a global constant. In the discretized implementation, λ is computed cell-by-cell from the local mean chemical potential, ensuring the constraint holds pointwise throughout the simulation. Our modified model B dynamics become

$$\frac{\partial \phi_i}{\partial t} = M \nabla^2 \mu_{i,\lambda CH} = M \nabla^2 (\mu_{i,CH} - \langle \mu_{CH} \rangle). \quad (26)$$

These dynamics also guarantee monotonic decrease of the total free energy:

$$\begin{aligned} \frac{dF_{\lambda CH}}{dt} &= \int_v \sum_{i=1}^n \frac{\delta F_{\lambda CH}}{\delta \phi_i} \frac{\partial \phi_i}{\partial t} d\mathbf{x} = \int_v \sum_{i=1}^n \mu_{i,\lambda CH} M \nabla^2 \mu_{i,\lambda CH} d\mathbf{x} \\ &= - \int_v \sum_{i=1}^n \nabla \mu_{i,\lambda CH} M \nabla \mu_{i,\lambda CH} d\mathbf{x} + \int_{\partial\Omega} \sum_{i=1}^n \mu_{i,\lambda CH} M \nabla \mu_{i,\lambda CH} \cdot \mathbf{n} d\Omega. \end{aligned} \quad (27)$$

With periodic boundary conditions $\int_{\partial\Omega} \sum_{i=1}^n \mu_{i,\lambda CH} M \nabla \mu_{i,\lambda CH} \cdot \mathbf{n} d\Omega = 0$, and

$$\frac{dF_{\lambda CH}}{dt} = - \int_v \sum_{i=1}^n \nabla \mu_{i,\lambda CH} M \nabla \mu_{i,\lambda CH} d\mathbf{x} \leq 0 \quad (28)$$

for all species with $\phi_i > 0$. At equilibrium, we obtain $\nabla \mu_{i,\lambda CH} = 0$, which means $\mu_{i,\lambda CH}$ is spatially uniform:

$$\mu_{i,\lambda CH} = \mu_{i,CH} - \langle \mu_{CH} \rangle = C_k, \quad (29)$$

where C_k is a global constant, independent of position $\mathbf{x}$.

A.5 CHL rule for spatial phase separation

Here we derive the CHL rule for the spatial case presented in Section 3. First, taking the derivative of $\check{F}_{CH}$, the total energy at equilibrium, we have

$$\begin{aligned}\frac{\partial \check{F}_{CH}}{\partial \chi_{ij}} &= \frac{\partial}{\partial \chi_{ij}} \int_v \left(\sum_{k,l=1}^n \frac{1}{2} \check{\phi}_k \chi_{kl} \check{\phi}_l + \sum_{k=1}^n \check{\phi}_k \log \check{\phi}_k + \sum_{k=1}^n \frac{\kappa}{2} |\nabla \check{\phi}_k|^2 \right) d\mathbf{x} \\ &= \int_v \left(\check{\phi}_i \check{\phi}_j + \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} \mu_{k,CH} \right) d\mathbf{x}.\end{aligned}\quad (30)$$

At equilibrium, $\mu_{k,CH} - \langle \mu_{CH} \rangle = C_k$, giving us

$$\frac{\partial \check{F}_{CH}}{\partial \chi_{ij}} = \int_v \left(\check{\phi}_i \check{\phi}_j + \langle \mu_{CH} \rangle \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} + \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} C_k \right) d\mathbf{x}.\quad (31)$$

The term $\langle \mu_{CH} \rangle \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}}$ vanishes because of the constraint $\sum_{k=1}^n \partial \check{\phi}_k = 0$. Exchanging integral and sum, and taking C_k outside the integral,

$$\int_v \sum_{k=1}^n \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} C_k d\mathbf{x} = \sum_{k=1}^n C_k \int_v \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} d\mathbf{x}.\quad (32)$$

Since $\int_v \check{\phi}_k d\mathbf{x}$ is conserved for each component, we have $\int_v \frac{\partial \check{\phi}_k}{\partial \chi_{ij}} d\mathbf{x} = 0$. The derivative becomes:

$$\frac{\partial \check{F}_{CH}}{\partial \chi_{ij}} = \int_v \check{\phi}_i \check{\phi}_j d\mathbf{x}.\quad (33)$$

The gradient of the contrastive objective is therefore

$$\frac{\partial \check{J}}{\partial \chi_{ij}} = \frac{\partial \check{F}_{CH}^{(+)}}{\partial \chi_{ij}} - \frac{\partial \check{F}_{CH}^{(-)}}{\partial \chi_{ij}} = \int_v \check{\phi}_i^{(+)} \check{\phi}_j^{(+)} d\mathbf{x} - \int_v \check{\phi}_i^{(-)} \check{\phi}_j^{(-)} d\mathbf{x},\quad (34)$$

and the spatial CHL update rule is

$$\Delta \chi_{ij} \propto - \int_v \check{\phi}_i^{(+)} \check{\phi}_j^{(+)} d\mathbf{x} + \int_v \check{\phi}_i^{(-)} \check{\phi}_j^{(-)} d\mathbf{x}.\quad (35)$$

A.6 Implicit-explicit integration of spatial phase separation

The spatial model tracks the composition field $\vec{\phi}(\mathbf{x}, t) = (\phi_1, \phi_2, \dots, \phi_n)$ at every point on a 2D grid with periodic boundaries, with the local constraint $\sum_{i=1}^n \phi_i(\mathbf{x}, t) = 1$ enforced at each grid cell independently. No chemical reservoir is included. We use a square grid of size (nx, ny) with grid spacing (dx, dy) , taking $nx = ny = 128$ and $dx = dy = 1.0$ here.

The total energy of the system is F_{CH} , and its variational derivative with respect to $\phi_i(\mathbf{x})$ gives the local chemical potential:

$$\mu_{i,CH}(\mathbf{x}, t) = \frac{\delta F_{CH}}{\delta \phi_i(\mathbf{x}, t)} = f'_{i,FH} - \kappa \nabla^2 \phi_i = \sum_{j=1}^n \chi_{ij} \phi_j + 1 + \log \phi_i - \kappa \nabla^2 \phi_i, \quad (36)$$

where $f'_{i,FH} = \partial f_{FH} / \partial \phi_i$ is the derivative of the bulk free energy density. The Laplacian term $-\kappa \nabla^2 \phi_i$ penalizes spatial curvature and gives rise to the interfacial energy.

The local sum-to-unity constraint is enforced by introducing the Lagrange multiplier, as described in the previous section. Recalling that our Lagrange-modified model B dynamics are $\frac{\partial \phi_i}{\partial t} = M \nabla^2 \mu_{i,\lambda CH} = M \nabla^2 (\mu_{i,CH} - \langle \mu_{CH} \rangle)$, we can substitute the expression for $\mu_{i,CH}$, to obtain our PDE:

$$\frac{\partial \phi_i}{\partial t} = M \nabla^2 f'_{i,FH} - M \kappa \nabla^4 \phi_i - M \nabla^2 \langle \mu_{CH} \rangle . \quad (37)$$

We solve this PDE using a semi-implicit spectral method. Taking the Fourier transform, spatial derivatives map as $\nabla^2 \rightarrow -k^2$ and $\nabla^4 \rightarrow k^4$, where $k = |\mathbf{k}|$ is the wavenumber magnitude, we have

$$\frac{\partial \hat{\phi}_i}{\partial t} = -M k^2 \left[\left(\hat{f}'_{i,FH} + \kappa k^2 \hat{\phi}_i \right) - \langle \hat{\mu}_{CH} \rangle \right] = -M k^2 \left(\hat{f}'_{i,FH} - \langle \hat{\mu}_{CH} \rangle \right) - M \kappa k^4 \hat{\phi}_i , \quad (38)$$

where $\hat{f}'_{i,FH}$ denotes the Fourier transform of the nonlinear bulk chemical potential. The $M \kappa k^4$ term is stiff at large wavenumbers – requiring prohibitively small time steps if treated explicitly – so it is treated implicitly, while the nonlinear terms are treated explicitly:

$$\frac{\hat{\phi}_i^{t+1} - \hat{\phi}_i^t}{\Delta t} = -M k^2 \left((\hat{f}'_{i,FH})^t - \langle \hat{\mu}_{CH} \rangle^t \right) - M \kappa k^4 \hat{\phi}_i^{t+1} . \quad (39)$$

When the nonlinear bulk term $\hat{f}'_{i,FH}$ varies rapidly, the basic semi-implicit scheme can still become unstable. To improve robustness, an r-stabilization scheme is employed [18]. A linear term $r \phi_i$ is simultaneously added and subtracted, shifting part of the nonlinear contribution into the implicit part without altering the equation:

$$\frac{\partial \phi_i}{\partial t} = M \nabla^2 (f'_{i,FH} - r \phi_i) + M \nabla^2 (r \phi_i) - M \kappa \nabla^4 \phi_i - M \nabla^2 \langle \mu_{CH} \rangle . \quad (40)$$

The first term is treated explicitly and the remaining linear terms implicitly. In Fourier space, this becomes

$$\frac{\partial \hat{\phi}_i}{\partial t} = -M k^2 \left(\hat{f}'_{i,FH} - r \hat{\phi}_i - \langle \hat{\mu}_{CH} \rangle \right) - M (r k^2 \hat{\phi}_i + \kappa k^4 \hat{\phi}_i) . \quad (41)$$

The semi-implicit update is then

$$\hat{\phi}_i^{t+1} = \frac{\hat{\phi}_i^t - \Delta t M k^2 \left((\hat{f}'_{i,FH})^t - r \hat{\phi}_i^t - \langle \hat{\mu}_{CH} \rangle^t \right)}{1 + \Delta t M (r k^2 + \kappa k^4)} . \quad (42)$$

The stabilization parameter $r = 2.0$ is used in all spatial simulations. An adaptive time step Δt is used to avoid simulation artifacts, guaranteeing monotonic decrease of the total energy and non-negative volume fractions throughout the simulation.

A.7 Trained interaction parameters

Here we provide the parameters (χ_{ij} , μ_i) obtained by CHL for the low-dimensional input-output tasks described in Section 2.3, and the MNIST task of Section 2.4.

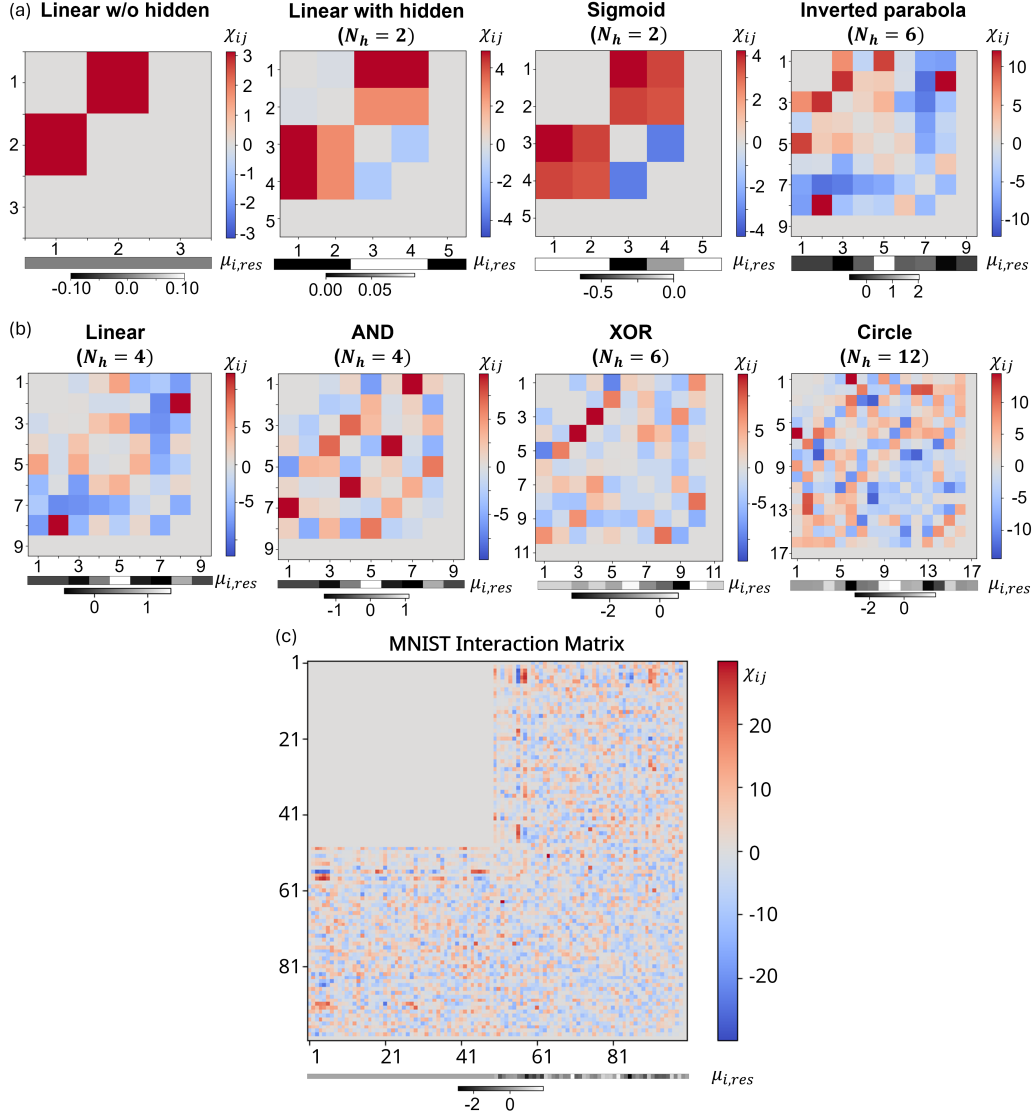


Figure 7 Trained parameters for low-dimensional input-output functions in Figure 3. (a) Trained interaction matrices χ_{ij} and reservoir potentials $\mu_{i,res}$ for 1-input 1-output analog relations. Beyond linear cases, we need to add hidden components to expand the accessible energy landscape and solve these tasks. (b) Trained interaction matrices and reservoir potentials for 2-input 2-output classification tasks. The structure of χ_{ij} reflects the classification boundary: stronger repulsive interactions between input and output components encode the boundary, while hidden components mediate nonlinear separations. The last rows and columns indicate the solvent component. (c) Trained interaction matrix and reservoir potentials for the MNIST dataset.

Bounded Analog Complexity

Ho-Lin Chen 

Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan

Xiang Huang 

Department of Computer Science, University of Illinois Springfield, IL, USA

Abstract

Current analog complexity theory, built on the General-Purpose Analog Computer (GPAC) model and polynomial ODEs, allows unbounded state variables – an assumption that is physically unrealistic for chemical reaction networks and other laboratory-scale analog computers. We develop a *bounded analog complexity theory* in which all state variables remain in compact intervals and physical time is the only diverging resource.

Our main technical contribution is *bounded surrogate compilation*, a compilation framework that transforms unbounded polynomial ODE systems into bounded ones while preserving computational limits and time-to-precision guarantees. We prove that on compact domains, physical time and trajectory length differ by at most constant factors; combined with the Bournez–Graça–Pouly characterization, this yields: bounded-GPAC polynomial time equals $\mathbf{P}$ over the reals.

We exhibit concrete constructions demonstrating fine-grained bounded time complexity – a tunable polynomial-degree family, a Lambert- W -based system achieving $\Theta(r \log r)$ time-to-precision (where r is the desired precision parameter, in nats: $|x(t) - \alpha| < e^{-r}$), and an iterated-logarithm tower realizing arbitrarily high complexity classes – all for the task of computing the constant 1. We show that bounded GPACs are closed under exponentiation (α^β) with time complexity equal to the harder input, and that the full GPAC-to-CRN compilation pipeline preserves time complexity class via a low-pass filter analysis of readout modules.

2012 ACM Subject Classification Theory of computation $\rightarrow$ Models of computation; Theory of computation $\rightarrow$ Computational complexity and cryptography

Keywords and phrases Analog computation, GPAC, bounded complexity, chemical reaction networks, polynomial ODE

Digital Object Identifier 10.4230/LIPIcs.DNA.32.14

Supplementary Material *Software*: https://colab.research.google.com/github/zinan-huang/notebooks/blob/main/bounded_gpac_exponentiation.ipynb

Funding *Ho-Lin Chen*: Supported in part by NSTC (Taiwan) grant 113-2221-E-002-204-MY3. *Xiang Huang*: Supported in part by Department of Energy EXPRESS grant DE-SC0024278.

Acknowledgements We thank the anonymous DNA32 reviewers for their detailed and constructive feedback.

1 Introduction

This paper develops a time complexity theory for chemical reaction network (CRN) computation in which all species concentrations remain bounded – as they must in any physical implementation. Our main technical contribution is a compilation method – *bounded surrogate compilation* – that transforms any polynomial ODE system into a bounded one while preserving all computed limits. In a bounded system, physical time – the time parameter t of the ODE, representing the actual elapsed time of the system – is a faithful complexity resource: it agrees with trajectory length up to constant factors.



© Ho-Lin Chen and Xiang Huang;

licensed under Creative Commons License CC-BY 4.0

32nd International Conference on DNA Computing and Molecular Programming (DNA 32).

Editors: Dominic Scalise and Robert Schweller; Article No. 14; pp. 14:1–14:22

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We work primarily with GPACs (polynomial ODEs), which offer a larger design space and greater algebraic flexibility than CRNs. However, our ultimate target is CRN implementation: every bounded GPAC construction in this paper can be translated into a CRN via dual-rail encoding and a readout subtraction module (Section 7). We prove that the subtraction module – the only step that could degrade precision – preserves time complexity.

We begin by recalling the line of work that leads to the present paper.

1.1 From computable numbers to analog complexity

- (i) Turing [19] introduced the notion of *computable numbers*: real numbers whose digits can be produced by a mechanical procedure.
- (ii) Shannon [17] introduced the General-Purpose Analog Computer (GPAC) as a mathematical abstraction of Bush’s differential analyzer. After refinements by Pour-El [16], Lipshitz–Rubel [15], and Graça–Costa [10], GPACs are now characterized by polynomial initial value problems (PIVPs): systems of the form $y' = p(y)$, $y(0) = y_0$, where p is a vector of polynomials with rational coefficients.
- (iii) Bournez, Graça, and Pouly [4] proved that GPACs, measured by *trajectory length* $L(t) = \int_0^t \|y'(s)\| ds$, are equivalent to Turing machines in both computability and complexity. In particular, polynomial trajectory length characterizes exactly the class **P** of polynomial-time computable functions.
- (iv) Huang et al. [13] defined CRN-computable numbers: real numbers computed as limits of chemical reaction network (CRN) trajectories. Notably, in their framework *all species concentrations are required to be bounded*. This ensures that the time parameter t of the ODE aligns with the notion of physical time, preventing Zeno-type speedups caused by diverging variables. Under this condition, the class of real-time CRN-computable numbers was shown to be a field containing transcendental numbers such as e and π . Our complexity theory can be viewed as extending this framework from a single convergence rate (real-time) to a full hierarchy of rates.
- (v) Huang and Huls [12] extended these ideas to large-population protocols (LPPs), showing that LPPs compute the same class of real numbers as GPACs and CRNs. Population protocols are inherently bounded: species live on a probability simplex.

A common thread runs through items (iv) and (v): the computations are bounded by design, as dictated by chemistry. No test tube holds infinite molecules; no population exceeds its total count. In these settings, the notion of time is unambiguous – physical time *is* the complexity resource, because no cost can be hidden in diverging auxiliary variables.

In contrast, the Bournez–Graça–Pouly framework in (iii) permits unbounded state variables – mathematically convenient but physically unrealistic for CRN implementations. This is why trajectory length, rather than physical time, serves as their complexity measure. Our goal is to bridge this gap: develop a complexity theory that respects the bounded nature of chemical systems while retaining the full power of the GPAC/PIVP framework as a design tool.

1.2 Why trajectory length, and why we depart from it

In the Bournez–Graça–Pouly framework, the complexity resource is not physical time t but the *trajectory length* (arc length) $L(t) = \int_0^t \|y'(s)\| ds$. This is the total distance traveled by the state vector in $\mathbb{R}^d$, measured by integrating the *speed* $\|y'(t)\|$ of the solution curve. When variables are unbounded, the speed $\|y'(t)\| = \|p(y(t))\|$ often grows without bound

as well, since p is a polynomial evaluated at a growing argument. In such cases, $L(t)$ can diverge even over a finite time interval, and arc length and physical time decouple: L may be infinite while t is finite.

A concrete example makes this vivid. Consider $x'(t) = (1-x)(1+v)^2$, $v'(t) = (1+v)^2$, $x(0) = v(0) = 0$. The output $x(t) = 1 - e^{1-1/(1-t)}$ converges to 1 at the finite time $t = 1$ – a naïve reading suggests this system “computes 1 in under one second.” But the auxiliary variable $v(t) = 1/(1-t) - 1$ diverges, and the trajectory speed $\|y'(t)\| \geq (1+v)^2 = 1/(1-t)^2$ explodes. The arc length $L(t) \geq t/(1-t) \rightarrow \infty$: Bournez et al.’s “time” is infinite, even though the clock reads less than one second. This illustrates the robustness of arc length as a complexity measure: it is invariant under time reparametrization, so no finite-time speedup via function composition can reduce the true cost. Despite the apparent speedup, the arc-length cost for r bits of precision is $\Theta(r)$ – the explosion hides computational work in the diverging variable, not in genuine progress toward the answer.

A dual scenario shows that arc length is also unreasonable as a resource on the convergent side.

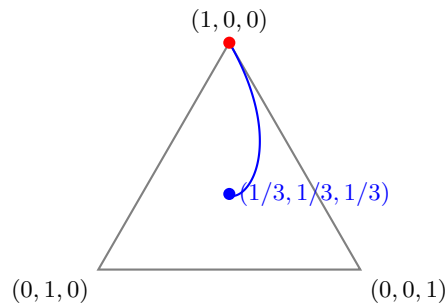
► **Example 1.1** (BFK protocol for $1/3$). The Bournez–Fraigniaud–Koeigler large-population protocol for $1/3$ [3] runs over three states $Q = \{1, 2, 3\}$ with cyclic transition rules

$$i \ j \rightarrow (i+1) \ (j+1) \pmod{3} \quad \text{for every } (i, j) \in Q^2.$$

The mean-field ODE on the 3-simplex $\{(x_1, x_2, x_3) : x_i \geq 0, x_1 + x_2 + x_3 = 1\}$ is

$$\dot{x}_i = 2(x_{i-1} - x_i), \quad i \in \{1, 2, 3\} \pmod{3},$$

with unique stable equilibrium $(1/3, 1/3, 1/3)$. Starting from the corner $(1, 0, 0)$, the trajectory spirals toward this equilibrium with envelope decaying as e^{-3t} (Figure 1). The arc length is uniformly bounded ($L(t) \leq \int_0^\infty \|y'(s)\| ds < \infty$), so by the BGP count the cost is finite. But the limit is approached only as $t \rightarrow \infty$, and the rate of approach – which is the natural and substantive complexity question for such systems – is not visible in the arc length.



■ **Figure 1** Trajectory of the BFK protocol for $1/3$ from $(1, 0, 0)$ to the equilibrium $(1/3, 1/3, 1/3)$ on the 2-simplex. The curve has finite total arc length but reaches the equilibrium only as $t \rightarrow \infty$.

When all state variables are bounded, the speed $\|y'(t)\| = \|p(y(t))\|$ is bounded on the compact state space, so $L(t) \leq v_{\max} t$: the trajectory length cannot grow faster than t . We adopt the time variable t itself as the complexity resource, and define the corresponding complexity theory on this basis.

But not every system of interest has bounded variables. The resolution is *bounded surrogate compilation* (Section 3): any polynomial ODE system can be compiled into a bounded one that computes the same output, with at most polynomial overhead in dimension.

The key idea is to slow the internal clock in proportion to the growth of unbounded variables, replacing diverging quantities with bounded surrogates in $[0, 1]$. Boundedness therefore does not restrict the class of computable objects – it reinterprets all analog computations in a setting where physical time is a faithful complexity resource.

1.3 Complexity in the bounded setting

Once physical time is faithful, we define time complexity in the style of computable analysis [14, 20]: the *time modulus* $\mu(r)$ is the physical time needed to achieve r bits of precision,¹ and its growth rate classifies the computation. This is finer than the Anderson–Joshi speed [1], which captures exponential convergence rates but does not distinguish sub-exponential profiles, and it is the natural analog of the framework in which CRN-computable numbers [13] were originally studied.

1.4 Results

We prove the following results.

1. **Bounded surrogate compilation** (Section 3). Any unbounded PIVP can be compiled into a bounded one preserving all computed limits, using bounded surrogate variables $U_{n,m} = f^m / (1 + f^n) \in [0, 1]$. Unbounded quantities – including the original time variable t and diverging state variables – are eliminated from the compiled system via what we call the *pass-through technique*: they influence the dynamics but are never represented as state variables (Remark 6.5).
2. **Bounded analog complexity for limit-readout computation** (Section 4). The complexity developed here is for computing real numbers under the limit-readout convention – a bounded PIVP computes α if a designated output variable converges to α at a quantified rate – not for encoding Turing-machine decision problems via GPAC or CRN. Bounded concentrations are strictly stronger than bounded derivatives, but the latter reduces to the former via compilation (Remark 4.2). The main result is that any (possibly unbounded) PIVP whose trajectory length and physical time are both polynomial in the BGP sense [4] compiles to a bounded PIVP with polynomial time modulus (Theorem 4.5). We do not claim a converse: BGP arc length is uninformative on convergent systems (Example 1.1).
3. **Constructible complexity classes** (Section 5). We construct bounded PIVPs realizing prescribed time complexities for the task of computing 1: polynomial of any degree n , $\Theta(r \log r)$ via the Lambert W function, and arbitrarily high classes via iterated-logarithm towers.
4. **Closure under exponentiation** (Section 6). If $\alpha > 0$ and β are bounded-GPAC computable, so is α^β , with time modulus $\mu_{\alpha^\beta}(r) = \max(\mu_\alpha(r), \mu_\beta(r)) + O(1)$. The output variable $z(t) = e^{R(t)}$ is strictly positive and does not require dual-railing; by the selective dual-rail technique of [11], only the intermediate variables need dual-rail encoding.
5. **CRN complexity preservation** (Section 7). The full GPAC-to-CRN pipeline – bounded surrogate compilation, dual-rail encoding, and readout – preserves time complexity class. The key step is the readout subtraction module, which acts as a low-pass filter: it is transparent to all sub-exponential inputs and thus preserves every constructible class of Section 5.

¹ Our formal modulus uses $|x - \alpha| < e^{-r}$, so r is in nats; polynomial classifications are invariant under the bits/nats rescaling.

1.5 Related work

The GPAC model originates with Shannon [17], with refinements by Pour-El [16], Lipshitz–Rubel [15], and Graça–Costa [10]. The computability and complexity theory of polynomial ODEs is due to Bournez, Graça, and Pouly [4, 5]. CRN computation has been studied under both stochastic [18] and deterministic [6, 7] semantics. CRN-computable numbers [13] and large-population protocols [12] connect chemical systems to computable analysis, with the notable feature that all species concentrations are bounded. Anderson and Joshi [1] develop modular CRN arithmetic with input-independent convergence rates, providing the key tool for our readout analysis (Section 7). Population protocols [2] provide another bounded setting; their connection to continuous models is established by Kurtz’s theorem.

Since many existing frameworks – CRN-computable numbers [13], population protocols [12], and the dual-rail compilation of [8, 11] – already require bounded species, our bounded surrogate compilation can serve as a *front end* that brings unbounded GPAC designs into their scope. In particular, combined with the balancing dilation technique of [12], our bounded compilations can be further translated into population protocols.

2 Preliminaries

We recall the basic definitions of polynomial initial value problems, the GPAC model, and the Bournez–Graça–Pouly complexity framework.

2.1 Polynomial Initial Value Problems

► **Definition 2.1** (PIVP). *A polynomial initial value problem (PIVP) is a system*

$$y'(t) = p(y(t)), \quad y(0) = y_0 \in \mathbb{Q}^d,$$

where $p: \mathbb{R}^d \rightarrow \mathbb{R}^d$ is a vector of polynomials with rational coefficients. We write $y(t) \in \mathbb{R}^d$ for the (unique, maximal) solution.

A PIVP is *bounded* if there exists $M > 0$ such that $\|y(t)\| \leq M$ for all $t \geq 0$. Equivalently, the trajectory remains in a compact set.

► **Proposition 2.2** (Bounded concentrations imply bounded derivatives). *If $y' = p(y)$ with $\|y(t)\| \leq M$ for all $t \geq 0$, then $\|y'(t)\| \leq C(p, M)$, where C depends only on p and M .*

Proof. p is continuous on the compact set $\|y\| \leq M$. ◀

2.2 The GPAC Model

Shannon’s General-Purpose Analog Computer (GPAC) generates functions by composing elementary units: constant multipliers, adders, integrators, and multipliers. It is well known that the class of functions generated by a GPAC coincides with the solutions of PIVPs [17]. Conversely, any PIVP can be realized by a GPAC circuit. We therefore use “GPAC” and “PIVP” interchangeably throughout.

2.3 Computability and Complexity via Trajectory Length

Bournez, Graça, and Pouly [4] define analog computation using a convergence semantics: a PIVP computes a real number α if a designated output variable $x(t) \rightarrow \alpha$ as $t \rightarrow \infty$. They measure complexity by the *trajectory length*

$$L(T) = \int_0^T \|y'(t)\| dt,$$

and show that the class of functions computable by PIVPs of polynomial trajectory length coincides with the polynomial-time computable functions of computable analysis.

This framework is elegant but permits unbounded state variables. Our goal is to enforce boundedness while preserving the complexity characterization.

2.4 Bounded-Time Computability

► **Definition 2.3** (Bounded-time computability). *Let $x(t)$ be a designated output variable of a bounded PIVP. The system computes a real number α if there exists a time modulus $\mu : \mathbb{N} \rightarrow [0, \infty)$ such that*

$$|x(t) - \alpha| < e^{-r} \quad \text{whenever } t > \mu(r).$$

The time complexity of the computation is the asymptotic growth of $\mu(r)$.

This aligns with computable analysis [14, 20]: the precision parameter r is in nats (one nat equals $1/\log 2 \approx 1.44$ bits), and the polynomial classification we work with is invariant under this constant rescaling. Throughout, $\log$ denotes the natural logarithm.

3 Bounded Compilation Framework

We present *bounded surrogate compilation*, a method that eliminates unbounded variables from polynomial ODE systems while preserving all computed limits.

The key idea is a time reparametrization. Composing the solution with a new time $t = s(\tau)$ produces the reparametrized variable $\bar{f}(\tau) := f(s(\tau))$ and introduces a factor $s'(\tau)$ into every ODE via the chain rule. If f^n is the highest power of the unbounded variable f in the system, setting $s'(\tau) = 1/(1 + \bar{f}(\tau)^n)$ converts every monomial $f^m \cdot s'(\tau)$ into the bounded quantity $U_{n,m} = \bar{f}^m/(1 + \bar{f}^n) \in [0, 1]$ – a *surrogate*. The surrogates satisfy a closed polynomial ODE: differentiating $U_{n,m}$ by the quotient rule introduces $d\bar{f}/d\tau$, but this quantity itself decomposes as $\sum_k h_k U_{n,k}$ – the rate of change of the unbounded variable is captured by its own surrogates. The system thus closes over itself, with no reference to f .

► **Construction 3.1** (Bounded surrogate compilation). *Let $Z = (z_1, \dots, z_{d-1}, f)$ satisfy $\dot{Z} = p(Z)$ with $p \in \mathbb{Q}[Z]^d$, where f is the unbounded variable to eliminate. Let n be the highest degree of f in any component of p . The construction proceeds in four steps.*

Step 1: Time change. Define $\bar{f}(\tau) = f(s(\tau))$ and $\bar{z}_j(\tau) = z_j(s(\tau))$, where $t = s(\tau)$ satisfies $s'(\tau) = 1/(1 + \bar{f}(\tau)^n)$. Define the bounded surrogates

$$U_{n,m}(\tau) = \frac{\bar{f}(\tau)^m}{1 + \bar{f}(\tau)^n} \in [0, 1], \quad m = 0, \dots, n, \quad U_{n,0} + U_{n,n} = 1.$$

Step 2: Rewrite $\bar{z}_j$ equations. Order each p_j by powers of $\bar{f}$: $p_j(\bar{Z}) = \sum_{m=0}^n g_{j,m}(\bar{z}) \bar{f}^m$, where $g_{j,m}$ is a polynomial in variables other than f . The chain rule and $s'(\tau) = 1/(1 + \bar{f}^n)$ give:

$$\frac{d\bar{z}_j}{d\tau} = \sum_{m=0}^n g_{j,m}(\bar{z}) \cdot \underbrace{\frac{\bar{f}^m}{1 + \bar{f}^n}}_{U_{n,m}}.$$

Step 3: Derive surrogates' ODE. By the quotient rule,

$$\frac{dU_{n,m}}{d\tau} = (mU_{n,m-1} - nU_{n,n-1}U_{n,m}) \cdot \frac{d\bar{f}}{d\tau}.$$

To express $d\bar{f}/d\tau$: order $p_d(\bar{Z}) = \sum_m h_m(\bar{z}) \bar{f}^m$ and apply $s'(\tau) = U_{n,0}$:

$$\frac{d\bar{f}}{d\tau} = p_d(\bar{Z}) \cdot s'(\tau) = \sum_m h_m(\bar{z}) \cdot \underbrace{\frac{\bar{f}^m}{1 + \bar{f}^n}}_{U_{n,m}} = \sum_m h_m \cdot U_{n,m}.$$

Substituting:

$$\boxed{\frac{dU_{n,m}}{d\tau} = (mU_{n,m-1} - nU_{n,n-1}U_{n,m}) \cdot \sum_k h_k(\bar{z}) U_{n,k}.} \quad (1)$$

The compiled system is a polynomial ODE in the variables $(\bar{z}_1, \dots, \bar{z}_{d-1}, U_{n,0}, \dots, U_{n,n})$. The unbounded variable $\bar{f}$ no longer appears: it has been fully absorbed by the bounded surrogates. The original time $t = s(\tau)$ is also absent from the compiled system. If any of $\bar{z}_1, \dots, \bar{z}_{d-1}$ remain unbounded, apply the construction again to eliminate the next unbounded variable, and repeat until all variables are bounded.

► **Proposition 3.2** (Limit preservation). *Let $T^* \leq \infty$ be the maximal existence time of the original system. If $\lim_{t \rightarrow T^*} z_j(t) = L$, then $\lim_{\tau \rightarrow \infty} \bar{z}_j(\tau) = L$.*

Proof. Since $s'(\tau) = 1/(1 + \bar{f}^n) > 0$, the function $t = s(\tau)$ is strictly increasing. If $T^* = \infty$ (no blowup), then $s(\tau) \leq \tau \rightarrow \infty$. If $T^* < \infty$ (finite-time blowup), then $\bar{f}(\tau) \rightarrow \infty$ as $s(\tau) \rightarrow T^{*-}$, forcing $s'(\tau) \rightarrow 0$; thus $s(\tau) \rightarrow T^*$ but never reaches it, and the approach takes $\tau \rightarrow \infty$. In both cases, $\bar{z}_j(\tau) = z_j(s(\tau)) \rightarrow L$. ◀

Applying Construction 3.1 iteratively to each unbounded variable and invoking Proposition 3.2 at each step, we obtain:

► **Theorem 3.3** (Bounded GPACs compute the same real numbers). *Every real number computable by a (possibly unbounded) GPAC is also computable by a bounded GPAC.*

In other words, unboundedness does not enlarge the class of GPAC-computable real numbers.

3.1 Worked example: finite-time explosion

Consider $x' = (1 - x)(1 + v)^2$, $v' = (1 + v)^2$, $x(0) = v(0) = 0$. The auxiliary variable $v(t) = 1/(1 - t) - 1$ explodes at $t = 1$, while the output $x(t) = 1 - e^{1-1/(1-t)}$ converges to 1 as $t \rightarrow 1^-$. Since $1 + v(t) = 1/(1 - t)$, the arc length is $L(T) \geq T/(1 - T) \rightarrow \infty$; the cost for r bits of precision is $\Theta(r)$ – no better than $z' = 1 - z$. Applying Construction 3.1 with $n = 2$, the surrogates $U_{0,2} = 1/(1 + \bar{v}^2)$ and $U_{1,2} = \bar{v}/(1 + \bar{v}^2)$ yield the bounded system:

$$\boxed{\begin{aligned} \bar{x}' &= (1 - \bar{x})(1 + 2U_{1,2}), \\ U_{1,2}' &= (U_{0,2} + U_{1,2})^2(2U_{0,2} - 1), \\ U_{0,2}' &= -2(U_{0,2} + U_{1,2})^2U_{1,2}, \end{aligned}} \quad \bar{x}(0) = 0, \quad U_{0,2}(0) = 1, \quad U_{1,2}(0) = 0, \quad (2)$$

with all variables in $[0, 1]$ for all $\tau \geq 0$. The variable v has been eliminated; the finite-time blowup is a disguise, not a speedup.

3.2 Application: computing π via pole compilation

Bounded surrogate compilation reveals a duality: *finite-time singularities in unbounded systems become infinite-time limits in bounded systems*. We illustrate by computing $\pi/2$ from the pole of $\tan(t)$.

The tangent function satisfies $x' = 1 + x^2$, $x(0) = 0$, with pole at $t = \pi/2$. Applying bounded surrogate compilation with $\phi(x) = 1/(1 + x^2)$ and introducing surrogates $U_0 = 1/(1 + \tau^2)$, $U_1 = \tau/(1 + \tau^2)$:

$$\boxed{U_0' = -2U_0U_1, \quad U_1' = U_0(2U_0 - 1).} \quad U_0(0) = 1, \quad U_1(0) = 0. \quad (3)$$

All variables bounded. The physical time $t(\tau) = \arctan(\tau) \rightarrow \pi/2$: the pole location becomes the computed output, readable from the surrogates (Figure 2).

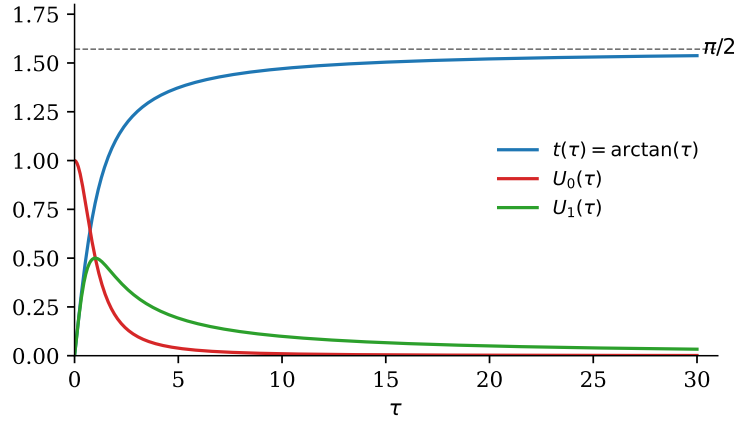


Figure 2 Time evolution of the compiled bounded system (3) in the new time variable τ . The physical time $t(\tau) = \arctan(\tau)$ converges to $\pi/2$ as $\tau \rightarrow \infty$ (dashed line). The surrogates U_0, U_1 remain in $[0, 1]$ throughout: the pole of $\tan(t)$ at $t = \pi/2$ has been compactified into an infinite-time limit in τ .

The error $\pi/2 - t(\tau) \sim 1/\tau$ gives time modulus $\mu(r) = \Theta(e^r)$. This is not optimal for computing π – specialized constructions achieve $\mu(r) = \Theta(r)$ [13] – but the method is fully generic, as the following theorem shows.

► **Theorem 3.4** (Poles of GPAC-implementable functions are GPAC-computable). *Let $y' = p(y)$, $y(0) = y_0 \in \mathbb{Q}^d$ be a PIVP with finite maximal existence time $T^* < \infty$. Then T^* is computable by a bounded PIVP.*

Proof. See Appendix A.1. ◀

4 Bounded Analog Complexity

The bounded surrogate compilation of Section 3 shows that every GPAC or CRN computation can be transformed into one whose state variables remain bounded, without loss of generality. Physical time t is then the only diverging resource, and we can use it to measure computational cost directly. With the time modulus of Definition 2.3 in hand, we now develop a complexity theory for bounded analog systems.

The complexity studied here is for analog computation of *real numbers* under the limit-readout convention of Definition 2.3; we do not encode Turing-machine computation in bounded CRNs, nor do we discuss decision problems under such setting in this work.

Terminological convention. A standard but important inversion relative to dynamical-systems terminology: *exponential convergence* ($O(e^{-kt})$) corresponds to *linear time* ($\mu(r) = \Theta(r)$), while *polynomial convergence* ($O(t^{-k})$) corresponds to *exponential time* ($\mu(r) = \Theta(e^{r/k})$). Throughout, “polynomial time” refers to the computable-analysis notion: $\mu(r) = O(r^k)$.

4.1 One-Sided Trajectory–Time Bound on Compact Domains

► **Lemma 4.1** (One-sided bound). *Let $y'(t) = p(y(t))$ be a PIVP whose trajectory remains in a compact domain D . Then the polynomial vector field p satisfies $\|p(y)\| \leq v_{\max}$ on D for some $v_{\max} < \infty$, hence*

$$L(t) = \int_0^t \|y'(\tau)\| d\tau \leq v_{\max} t.$$

Proof. By compactness, $\|p\|$ attains a maximum $v_{\max}$ on D . The bound on $L(t)$ follows directly. ◀

The converse one-sided inequality $L(t) \geq v_{\min} t$ requires a non-degeneracy hypothesis $\|p(y(t))\| \geq v_{\min} > 0$ which is incompatible with convergence to a fixed point and therefore fails for the very systems described by Definition 2.3. We do not assume it; only the upper bound of Lemma 4.1 is used below.

► **Remark 4.2** (Bounded concentrations vs. bounded derivatives). Bounded concentrations imply bounded derivatives (compactness of the state space), but not conversely: $x' = 1$ has bounded derivative yet $x(t) \rightarrow \infty$. A system with bounded derivatives admits the one-sided bound $L(t) \leq Mt$ but is not automatically bounded; Construction 3.1 compiles it into a bounded-concentration system without changing the computed object.

4.2 Iterative Compilation Preserves Polynomial Arc Length

The proof of bounded compilation is by induction on the number of unbounded variables eliminated. The tracked resource through each compilation step is the trajectory arc length L . The base case is the BGP polynomial trajectory length of the original system. The inductive step (Lemma 4.4) shows that one compilation iteration sends a polynomial-arc-length system to another polynomial-arc-length system. When the iteration terminates with all variables bounded, Lemma 4.1 converts polynomial arc length to polynomial physical time directly. The converse direction is immediate: any bounded PIVP whose physical time is polynomial has polynomial arc length by Lemma 4.1.

► **Lemma 4.3** (Polynomial state bound from polynomial arc length). *Let $y'(t) = p(y(t))$ be a PIVP for which there is a polynomial q such that $L(t_r) \leq q(r)$ for every precision r , where t_r is the physical time at which the output reaches precision r (Definition 2.3). Then for every component y_i and every $s \in [0, t_r]$,*

$$|y_i(s)| \leq |y_i(0)| + q(r).$$

Proof. $|y_i(s) - y_i(0)| \leq \int_0^s |y'_i(u)| du \leq \int_0^s \|p(y(u))\| du = L(s) \leq L(t_r) \leq q(r)$. ◀

► **Lemma 4.4** (One compilation step preserves polynomial arc length). *Let S_{pre} be a PIVP whose trajectory length and physical time both satisfy $L_{\text{pre}}(t_r), t_r \leq q(r)$ for some polynomial q and every precision r . Apply bounded surrogate compilation (Construction 3.1) to eliminate one unbounded variable f of maximal degree n in p . Then the post-compilation system S_{post} , with new time variable τ , satisfies*

$$\tau_r \leq Q_t(r) \quad \text{and} \quad L_{\text{post}}(\tau_r) \leq Q_L(r)$$

for some polynomials Q_t, Q_L determined by q and n .

Proof. By Lemma 4.3, $|f(s)| \leq |f(0)| + q(r) =: M_f(r) = \text{poly}(r)$ for $s \in [0, t_r]$.

For the new time variable: from $d\tau/dt = 1 + \bar{f}^n$ (Construction 3.1, Step 1) and $|f(u)| \leq M_f(r)$,

$$\tau_r = \int_0^{t_r} (1 + f(u)^n) du \leq t_r(1 + M_f(r)^n) \leq q(r)(1 + M_f(r)^n) = \text{poly}(r) =: Q_t(r).$$

For the new arc length: the post-compilation vector field p_{post} acts on the state $(\bar{z}_1, \dots, \bar{z}_{d-1}, U_{n,0}, \dots, U_{n,n})$, where each $U_{n,m} \in [0, 1]$ (Construction 3.1, Step 1). By Steps 2 and 3 of the construction, $d\bar{z}_j/d\tau = \sum_m g_{j,m}(\bar{z})U_{n,m}$ and $dU_{n,m}/d\tau$ is polynomial in $(\bar{z}, U_{n,\cdot})$. Since the $U_{n,m}$ lie in $[0, 1]$ and $|\bar{z}_j(s)| \leq |\bar{z}_j(0)| + q(r) = M_z(r) = \text{poly}(r)$ on the precision- r prefix by Lemma 4.3, every component of p_{post} is bounded by a polynomial in $M_z(r) + M_f(r)$, hence $\|p_{\text{post}}\| \leq M_p(r) = \text{poly}(r)$. Therefore

$$L_{\text{post}}(\tau_r) = \int_0^{\tau_r} \|p_{\text{post}}\| d\sigma \leq M_p(r) \tau_r \leq \text{poly}(r) =: Q_L(r). \quad \blacktriangleleft$$

► **Theorem 4.5** (Iterative bounded compilation). *Let $y'(t) = p(y(t))$ be a (possibly unbounded) PIVP whose trajectory length and physical time both satisfy $L(t_r), t_r \leq q(r)$ for some polynomial q and every precision r (BGP normalization). Iterated bounded surrogate compilation (Construction 3.1) produces a fully bounded PIVP computing the same real number (or function), whose trajectory length and physical time at precision r are both polynomial in r . The dimensional overhead is polynomial in the maximal monomial degree of p .*

Proof. By induction on the number of remaining unbounded variables.

Base case. $S_0 := y$ has $L_0(t_r), t_r \leq q(r)$ by hypothesis.

Inductive step. Suppose after i compilation iterations the system S_i satisfies $L_i(\tau_r^{(i)}), \tau_r^{(i)} \leq Q^{(i)}(r)$ for some polynomial $Q^{(i)}$, and still contains at least one unbounded variable f_{i+1} . Apply Lemma 4.4 to S_i with input bound polynomial $Q^{(i)}$, eliminating f_{i+1} . The output system S_{i+1} satisfies $L_{i+1}, \tau_r^{(i+1)} \leq Q^{(i+1)}(r)$ for the polynomial $Q^{(i+1)} := \max(Q_t, Q_L)$.

Termination. The original system has finitely many unbounded variables $f_1, \dots, f_k$, each one fully absorbed by its compilation step (Construction 3.1). The induction therefore terminates after k iterations with S_k fully bounded, having $L_k, \tau_r^{(k)} \leq Q^{(k)}(r) = \text{poly}(r)$. Limit preservation throughout by Proposition 3.2. Each step adds at most $n + 1$ surrogate variables, so the total dimensional overhead is at most $k(n + 1)$. $\blacktriangleleft$

► **Theorem 4.6** (From BGP poly-length to bounded poly-time). *If a real number (or function on a compact domain) is computable by a (possibly unbounded) PIVP in polynomial trajectory length and polynomial physical time in the sense of Bournez–Graça–Pouly [4], then it is computable by a bounded PIVP in polynomial physical time (Definition 2.3).*

Proof. Theorem 4.5. $\blacktriangleleft$

► **Remark 4.7** (The converse direction is not pursued). We do not claim a converse. For a PIVP whose vector field has exponentially decaying norm along the trajectory, the arc length $L(t)$ stays finite as $t \rightarrow \infty$, while the time required to reach precision e^{-r} can range from linear to exponential in r . Example 1.1 is the canonical instance: BGP arc length is bounded by a constant; the time-to-precision modulus is $\Theta(r)$. Two such systems whose times-to-precision differ by exponentials in r can share the same finite arc length, so the BGP resource does not distinguish between them. The BGP framework records the cost of explosive (state-diverging) computation faithfully, but on convergent systems it loses the rate-of-approach information that constitutes the natural complexity question.

5 Constructible Bounded Time Complexity

Analyzing the convergence rate of a general ODE is notoriously difficult. Rather than analyzing arbitrary systems, we take the opposite approach: we start from known functions with well-understood asymptotics and *construct* bounded polynomial ODE systems that realize prescribed time complexity classes. All constructions compute the same value: the real number 1.

Throughout this section, $r \in \mathbb{N}$ denotes the *precision parameter*: the system is required to produce r nats of accuracy at the output, i.e., $|x_1(t) - 1| < e^{-r}$ (Definition 2.3; equivalently, r nats are $r/\log 2$ bits in the standard computable-analysis sense). The *time modulus* $\mu(r)$ is the time the bounded PIVP needs to reach this precision.

The design recipe is as follows. Given a target time modulus $\mu(r)$, we:

1. **Choose a precision clock.** Pick a monotone function $g(t) \rightarrow \infty$ with $g(\mu(r)) \geq r$, intuitively the number of nats of precision delivered by time t . The clock g is a design abstraction, not a state variable of the PIVP: it is generally unbounded and therefore not directly realizable in a bounded system.
2. **Apply the Möbius surrogate.** Substitute the fractional-linear (Möbius) compactification $u(t) = g(t)/(1+g(t)) \in [0, 1)$, so $u \rightarrow 1$ as $g \rightarrow \infty$. The rational inverse $g = u/(1-u)$ turns polynomial dependence on g into a rational function of u with denominators that are powers of $(1-u)$; residual denominators are cleared by the time reparametrization $s'(t) = (1-u)^{-k}$ familiar from Section 3. The surrogate u , together with any auxiliary bounded states, is what the PIVP actually instantiates: g never appears as a state, in the sense of Remark 6.5. (Throughout this section the direct Möbius closure happens to be already polynomial, so no reparametrization is needed.)
3. **Read out via exponential approach.** Drive the output by $x'_1(t) = (1 - x_1(t)) \cdot h(\text{bounded states})$ so that $1 - x_1(t) = e^{-g(t)}$ (or a constant multiple). Then $|x_1(t) - 1| < e^{-r}$ exactly when $g(t) > r$, i.e., when $t > \mu(r)$.

5.1 Linear and Polynomial Time

The simplest instance is $x'_1 = 1 - x_1$, $x_1(0) = 0$, giving $x_1(t) = 1 - e^{-t}$ and $\mu(r) = r$. This is the degenerate case of the recipe with $g(t) = t$. Since $g'(t) \equiv 1$ is already constant, no Möbius surrogate is needed: step 2 of the recipe is trivial and the bounded PIVP closes on the single output variable x_1 . We generalize this to a tunable polynomial family in Section 5.2 below.

5.2 Tunable Polynomial Time: $\mu(r) = \Theta(r^n)$

► **Theorem 5.1** (Polynomial-time constructibility). *For every $n \in \mathbb{N}$, there exists a bounded PIVP that computes the constant 1 with time modulus $\mu_n(r) = \Theta(r^n)$.*

Proof. We instantiate the recipe with precision clock $g(t) = (1+t)^{1/n} - 1$ and Möbius surrogate

$$u(t) = \frac{g(t)}{1+g(t)} = 1 - (1+t)^{-1/n} \in [0, 1).$$

The bounded system on the two state variables $(u, x) \in [0, 1]^2$ is

$$\begin{aligned} u'(t) &= \frac{1}{n} (1 - u(t))^{n+1}, \\ x'(t) &= (1 - x(t)) \cdot \frac{1}{n} (1 - u(t))^{n-1}, \end{aligned} \quad u(0) = x(0) = 0, \quad (4)$$

a polynomial PIVP of degree $n+1$. Closure follows from $1 - u = (1+t)^{-1/n}$, hence $1+t = (1-u)^{-n}$, which gives

$$u' = \frac{1}{n} (1+t)^{-1/n-1} = \frac{1}{n} (1-u)(1+t)^{-1} = \frac{1}{n} (1-u) \cdot (1-u)^n = \frac{1}{n} (1-u)^{n+1}.$$

For the output, $g'(t) = \frac{1}{n} (1+t)^{(1-n)/n} = \frac{1}{n} (1-u)^{n-1}$, so $x' = (1-x)g' = (1-x)\frac{1}{n}(1-u)^{n-1}$. Integration yields $1 - x(t) = \exp(-g(t)) = \exp(-(1+t)^{1/n} + 1)$. Therefore $|x(t) - 1| < e^{-r}$ iff $(1+t)^{1/n} - 1 > r$, i.e., $t > (r+1)^n - 1 = \Theta(r^n)$. ◀

► **Remark 5.2** (Smooth representative of the asymptotic class). The clock $g(t) = (1+t)^{1/n} - 1$ is the smooth representative of the asymptotic class $t^{1/n}$: the shift $t \mapsto 1+t$ moves the branch point of $t^{1/n}$ off the operating region $t \geq 0$. Using $g_0(t) = t^{1/n}$ directly yields the closure $u'(t) = \frac{1}{n} (1-u)^{n+1}/u^{n-1}$, which is singular at the initial state $u = 0$ for $n \geq 2$; the “+1” shift is the minimal regularization that restores polynomial closure with finite initial slope $u'(0) = 1/n$. The same regularization appears implicitly in the Lambert- W construction of Section 5.2 and the iterated-logarithm tower of Appendix A.2.

Thus every polynomial degree is constructible: linear for $n = 1$, quadratic for $n = 2$, cubic for $n = 3$, etc.

$\Theta(r \log r)$ Time via Lambert W

► **Proposition 5.3** ($\Theta(r \log r)$ constructibility). *There exists a bounded PIVP that computes the constant 1 with time modulus $\mu(r) = \Theta(r \log r)$.*

Step 1 (precision clock and target readout). Let W denote the Lambert function satisfying $W(t)e^{W(t)} = t$. We take the precision clock

$$g(t) = e^{W(t)} - 1,$$

which is monotone with $g(0) = 0$ and $g(t) \rightarrow \infty$, and design the readout to satisfy

$$1 - x_1(t) = e^{-g(t)} = e^{1-e^{W(t)}}.$$

The precision condition $|1 - x_1(t)| < e^{-r}$ is then equivalent to $g(t) \geq r$, i.e., $e^{W(t)} \geq r+1$, equivalently $W(t) \geq \log(r+1)$. The Lambert identity $t = We^W$ gives $t \geq (r+1)\log(r+1) = \Theta(r \log r)$, so the readout has the desired modulus $\mu(r) = \Theta(r \log r)$.

Step 2 (Möbius surrogates). The Lambert evolution

$$W'(t) = \frac{1}{(1+W)e^W}$$

involves *two* unbounded quantities, $1+W$ and e^W . A bounded polynomial encoding therefore needs to carry information about both. We introduce two Möbius surrogates:

$$v(t) := \frac{W(t)}{1+W(t)} \in [0, 1) \quad (\text{Möbius surrogate of } W),$$

$$u(t) := \frac{g(t)}{1+g(t)} = 1 - e^{-W(t)} \in [0, 1) \quad (\text{Möbius surrogate of } g).$$

With $u(0) = v(0) = 0$ and $u, v \rightarrow 1$, The rational inversions $1+W = 1/(1-v)$ and $e^W = 1/(1-u)$ together polynomialize the Lambert dynamics: from $W' = 1/((1+W)e^W) = (1-u)(1-v)$,

$$u' = e^{-W} W' = (1-u)^2(1-v), \quad v' = \frac{W'}{(1+W)^2} = (1-v)^3(1-u).$$

Neither surrogate alone would close in polynomial form: v' requires e^W (read off u), and u' requires W' which in turn requires $1+W$ (read off v). The pair (u, v) is the minimal bounded encoding of the Lambert pair (W, e^W) .

Step 3 (readout dynamics). The readout x_1 is the designated output variable of the bounded PIVP. Following the recipe, it satisfies the exponential-approach equation $x_1'(t) = (1-x_1(t))g'(t)$, which with $x_1(0) = 0$ integrates to $1-x_1(t) = e^{-g(t)}$ exactly, as targeted in Step 1. The chain rule gives

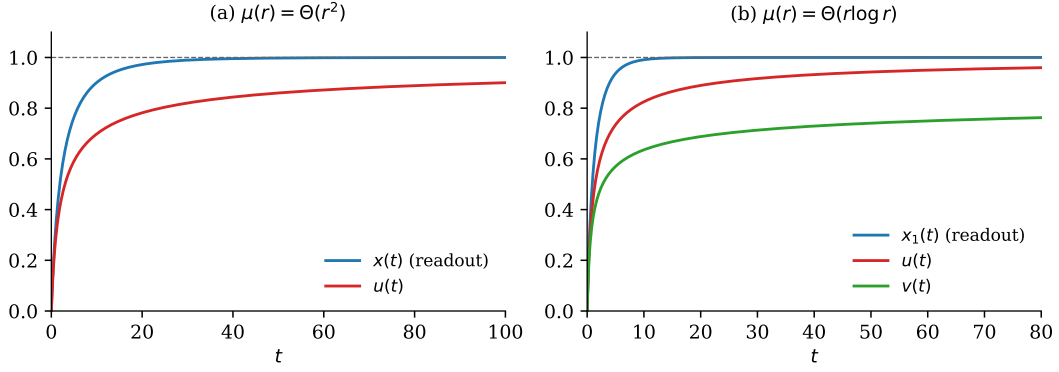
$$g'(t) = e^{W(t)} W'(t) = \frac{e^W}{(1+W)e^W} = \frac{1}{1+W} = 1-v,$$

so the readout equation is $x_1' = (1-x_1)(1-v)$. The surrogates u, v are auxiliary states that exist only to polynomialize g' ; they are not themselves the output.

Bounded polynomial system. Collecting steps 1–3 yields the polynomial PIVP on $(u, v, x_1) \in [0, 1]^3$:

$$\boxed{\begin{aligned} u' &= (1-u)^2(1-v), \\ v' &= (1-v)^3(1-u), \\ x_1' &= (1-x_1)(1-v), \end{aligned}} \quad u(0) = v(0) = x_1(0) = 0. \quad (5)$$

Proof of Proposition 5.3. The bounded polynomial system (5) arises from the recipe instantiation in Steps 1–3 above. Integrating $x_1'/(1-x_1) = 1-v = 1/(1+W)$ from $x_1(0) = 0$ yields $1-x_1(t) = e^{-g(t)} = e^{1-e^{W(t)}}$ exactly, so $|1-x_1(t)| < e^{-r}$ holds iff $g(t) \geq r$, i.e., iff $t \geq (r+1)\log(r+1) = \Theta(r \log r)$. ◀



■ **Figure 3** Time evolution of the constructible families. (a) Polynomial $\Theta(r^2)$: $u(t) = 1 - (1+t)^{-1/2}$, $x(t) = 1 - e^{-(\sqrt{1+t}-1)}$. (b) Lambert- W $\Theta(r \log r)$: $u(t) = 1 - e^{-W(t)}$, $v(t) = W(t)/(1 + W(t))$, $x_1(t) = 1 - e \cdot e^{-e^{W(t)}}$. All states remain in $[0, 1)$; the output convergence rate determines the time modulus.

An iterated-logarithm tower construction extends the linear- $\Theta(r \log r)$ - $\Theta(r^n)$ hierarchy to every level of the elementary exponential hierarchy: a depth- k construction yields time modulus $\mu(r) = \exp^{(k+1)}(r + O(1))$. The construction, analysis, and tabulation are recorded in Appendix A.2.

► **Remark 5.4.** All constructions in this section – the linear, polynomial, Lambert W , and iterated-logarithm families – share the common design pattern of the recipe in Section 5: choose a precision clock with known asymptotics, encode it via a Möbius surrogate, and read off the time modulus. Every construction computes the same trivial value 1; the complexity is intrinsic to the dynamics, not to the target.

6 Closure and Richness of Bounded Computation

The preceding sections established the foundations of bounded analog complexity: the compilation framework the core complexity definitions, and constructible time complexity classes. We now demonstrate the computational richness of bounded GPACs by showing closure under exponentiation.

6.1 Closure Under Exponentiation

Beyond compilation, we demonstrate the computational richness of bounded GPACs by showing closure under exponentiation.

► **Theorem 6.1** (Bounded GPAC exponentiation). *If $\alpha > 0$ and $\beta \in \mathbb{R}$ are computable by bounded GPACs, then α^β is computable by a bounded GPAC.*

The key identity is $\alpha^\beta = e^{\beta \ln \alpha}$. Direct computation of $\ln(x(t))$ is problematic when $x(0) = 0$. The solution is to introduce a shifted variable $x_1(t) \rightarrow \alpha - 1$ with $x_1(0) = 0$, so that $\ln(1 + x_1)$ starts at $\ln(1) = 0$, avoiding the singularity.

► **Construction 6.2** (Computing α^β). Given upstream GPACs computing $x(t) \rightarrow \alpha$ and $y(t) \rightarrow \beta$, define:

$$\begin{cases} x'_1 = (x - 1) - x_1, \\ u' = (1 - v) x'_1, \\ v' = (1 - v)^2 x'_1, \\ z' = z (y' u + y (1 - v) x'_1), \end{cases} \quad (6)$$

with $x_1(0) = u(0) = v(0) = 0$ and $z(0) = 1$. Here $u = \ln(1 + x_1)$ and $v = x_1/(1 + x_1)$.

Proof of Theorem 6.1. See Appendix A.4. ◀

► **Corollary 6.3** (CRN exponentiation). The construction of Construction 6.2 can be implemented as a CRN. For $\alpha < 1$, the intermediate variables x_1, u, v may become negative and require dual-rail encoding. However, every term in the ODE for z contains z as a factor ($z' = z \cdot (\dots)$), so no negative contribution to z' can arise independently of z itself. By the selective dual-railing criterion of [11], z is not infected and does not require dual-rail encoding: it is represented directly as a single species concentration.

► **Theorem 6.4** (Complexity of α^β). Let $\alpha > 0$ and $\beta \in \mathbb{R}$ be computable by bounded GPACs with time moduli $\mu_\alpha(r)$ and $\mu_\beta(r)$. Then α^β is computable by a bounded GPAC with time modulus

$$\mu_{\alpha^\beta}(r) = \max(\mu_\alpha(r + C), \mu_\beta(r + C)) + O(1),$$

where $C = C(\alpha, \beta)$ is a constant depending only on α and β . In particular, exponentiation preserves the time complexity class: the cost is determined by the slower input.

Proof. See Appendix A.5. ◀

Exponentiation preserves the complexity class: the time modulus of α^β is determined by the *slower* input. For example, both π and e are computable in linear time ($\mu(r) = \Theta(r)$) by known GPAC constructions [13]; therefore $(\pi/4)^e$ and e^π are also computable in linear time.

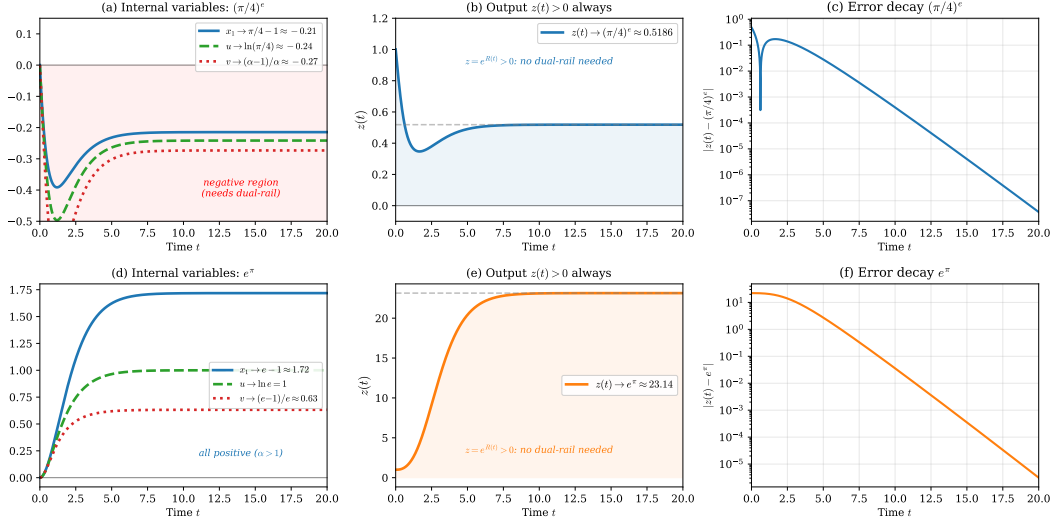
► **Remark 6.5** (The pass-through technique). The quantity $R(t) = y(t) \cdot \ln(1 + x_1(t))$ appears in the exponent of z but is never tracked as an explicit variable – only its derivative R' enters the ODE for z . This is a general design technique for bounded GPACs: intermediate quantities that are transcendental or unbounded can *pass through* the system as long as their derivatives close over the existing bounded state variables. Bounded surrogate compilation (Construction 3.1) is another instance: both the physical time t and the unbounded variable f pass through the compiled system without appearing in the final ODE.

7 Bounded CRN Time Complexity

The GPAC-to-CRN pipeline has two steps: *dual-rail encoding* converts a signed polynomial ODE into mass-action kinetics, and a *readout module* extracts the algebraic difference $X_1^+ - X_1^-$ as a single species concentration. Dual-rail encoding, following [8] (see also [13]), is exact: $X_1^+ - X_1^-$ satisfies the original ODE, and all dual-rail species remain bounded.

The question is whether the readout step – subtraction – also preserves time complexity. Subtraction is itself a dynamical system with its own convergence rate; if it is too slow, it could degrade the complexity class inherited from the GPAC. We show that subtraction acts as a low-pass filter: it is transparent to all sub-exponential inputs, and thus preserves the time complexity class for all constructible classes of Section 5.

Bounded GPAC exponentiation: negative intermediates, strictly positive output



Interactive notebook: https://colab.research.google.com/github/zinan-huang/notebooks/blob/main/bounded_gpac_exponentiation.ipynb

Figure 4 Computing $(\pi/4)^e$ (top) and e^π (bottom) via the bounded GPAC construction (Construction 6.2), using the ODE constructions of [13] for $\pi/4$, e , and π . **(a)** When $\alpha < 1$, the intermediate variables x_1 , u , v take negative values (red shading), requiring selective dual-rail encoding for CRN implementation [11]. **(b, e)** The output $z(t) = e^{R(t)}$ is strictly positive for all $t \geq 0$, regardless of the sign of the intermediates: no dual-railing is needed for the output species. **(d)** When $\alpha > 1$ (as in e^π), all intermediate variables remain positive. **(c, f)** Both computations converge exponentially, consistent with linear time modulus $\mu(r) = \Theta(r)$.

7.1 Readout subtraction modules

After dual-rail encoding, $x_1(t) = X_1^+(t) - X_1^-(t)$ satisfies the original ODE but is not a single species concentration. A readout species Z must compute $Z(t) \rightarrow \alpha$ via mass-action reactions. Anderson and Joshi [1] provide the *absolute-difference module* ($A + Z \rightarrow A + 2Z$, $B + Z \rightarrow B$, $2Z \rightarrow Z$):

$$\dot{Z} = x_1(t)Z - Z^2, \quad Z^* = \alpha. \quad (7)$$

An alternative is *double inversion* [13], computing $1/x_1$ then $1/(1/x_1) = x_1$ via:

$$\dot{Y} = 1 - x_1(t)Y, \quad \dot{Z} = 1 - YZ, \quad (8)$$

with $Y^* = 1/\alpha$ and $Z^* = \alpha$.

Anderson and Joshi [1] define the *speed* $\rho := -\limsup_{t \rightarrow \infty} \frac{\ln |x(t) - x^*|}{t}$ and show each arithmetic module has speed ≥ 1 . Positive speed ($\rho > 0$) is equivalent to exponential convergence, i.e., linear time $\mu(r) = \Theta(r)$ – the real-time class studied in [13]. Our time modulus (Definition 2.3) refines and generalizes this notion: it distinguishes the sub-exponential profiles ($\rho = 0$) that the speed framework treats uniformly, with $\rho = \lim_{r \rightarrow \infty} r/\mu(r)$ when the limit exists.

7.2 Low-pass filter analysis

Write $x_1(t) = \alpha + \epsilon(t)$ and $Z(t) = \alpha + \delta(t)$. Linearizing (7) around $Z^* = \alpha > 0$:

$$\dot{\delta} + \alpha \delta = \alpha \epsilon(t), \quad (9)$$

with solution

$$\delta(t) = e^{-\alpha t} \delta(0) + \alpha \int_0^t e^{-\alpha(t-s)} \epsilon(s) \, ds. \quad (10)$$

This is a first-order low-pass filter with cutoff rate α . The convolution kernel $e^{-\alpha(t-s)}$ exponentially forgets the past: if $\epsilon(s)$ varies slowly relative to the forgetting rate, it can be pulled out of the integral, giving $\delta(t) \approx \epsilon(t)$ – the module is transparent. Conversely, if $\epsilon(s) = C e^{-\beta s}$ with $\beta > \alpha$, the integral evaluates to $\frac{\alpha C}{\beta - \alpha} e^{-\alpha t}$: the fast input component vanishes and the output decays at the module's own rate α . In short: *a fast module cannot speed up a slow input; it can only slow down a fast one.*

For the double-inversion subtraction of [13], the cascade has effective speed $\min(\alpha, 1/\alpha)$; the analysis is analogous.

7.3 Convergence preservation

The convolution (10) determines how the readout error $\delta(t)$ relates to the input error $\epsilon(t)$. The key dichotomy:

► **Lemma 7.1** (Input-limited regime). *If $\epsilon(t)$ decays slower than $e^{-\alpha t}$ (i.e., $e^{\alpha t} |\epsilon(t)| \rightarrow \infty$), then*

$$\delta(t) \sim \epsilon(t) \quad (t \rightarrow \infty).$$

The readout error tracks the input error asymptotically.

Proof. The homogeneous part $e^{-\alpha t} \delta(0) = o(\epsilon(t))$. For the convolution, define $I(t) = \alpha \int_0^t e^{\alpha s} \epsilon(s) \, ds$. Since $e^{\alpha t} |\epsilon(t)| \rightarrow \infty$, both $|I(t)| \rightarrow \infty$ and $e^{\alpha t} \rightarrow \infty$. By L'Hôpital's rule:

$$\lim_{t \rightarrow \infty} \frac{I(t)}{e^{\alpha t}} = \lim_{t \rightarrow \infty} \frac{\alpha e^{\alpha t} \epsilon(t)}{\alpha e^{\alpha t}} = \lim_{t \rightarrow \infty} \epsilon(t) = 0.$$

More precisely, $e^{-\alpha t} I(t) - \epsilon(t) \rightarrow 0$, giving $\delta(t) \sim \epsilon(t)$. ◀

► **Lemma 7.2** (Module-limited regime). *If $\epsilon(t) = C_\epsilon e^{-\beta t}$ with $\beta > \alpha$, then*

$$\delta(t) = \frac{\alpha C_\epsilon}{\beta - \alpha} e^{-\alpha t} + O(e^{-\beta t}).$$

The readout is throttled to the module's intrinsic speed α .

Proof. Direct evaluation of the convolution integral with $\epsilon(s) = C_\epsilon e^{-\beta s}$. ◀

► **Theorem 7.3** (CRN readout preserves time complexity). *Let a bounded GPAC compute $\alpha > 0$ with time modulus $\mu(r)$. Let Z be the readout species implemented via either the Anderson–Joshi absolute-difference module or the double-inversion method. Then:*

1. *If $\mu(r) = \omega(r/\alpha)$ (input convergence slower than the module's intrinsic rate $e^{-\alpha t}$), then $\mu_Z(r) = \mu(r) + O(1)$. The time complexity class is **preserved exactly**.*
2. *If $\mu(r) = O(r/\alpha)$ (input convergence at least as fast as $e^{-\alpha t}$), then $\mu_Z(r) = r/\alpha + O(1)$. The readout is **throttled** to linear time with rate α .*

In either case, the readout speed in the Anderson–Joshi sense is $\rho_Z = \min(\rho_{\text{in}}, \alpha)$, consistent with their Composition Theorem.

Proof. See Appendix A.3. ◀

► **Corollary 7.4** (Constructible classes are CRN-realizable). *All constructible bounded time complexity classes of Section 5 (computing $\alpha = 1$) are realizable by CRNs with the same time modulus:*

<i>Class</i>	<i>GPAC modulus $\mu(r)$</i>	<i>CRN readout modulus $\mu_Z(r)$</i>
<i>Linear</i>	$\Theta(r)$	$\Theta(r)$
<i>Lambert W</i>	$\Theta(r \log r)$	$\Theta(r \log r)$
<i>Polynomial degree n</i>	$\Theta(r^n)$	$\Theta(r^n)$

Proof. With $\alpha = 1$, the module speed is 1 (exponential decay e^{-t}). The Lambert W and polynomial constructions converge sub-exponentially, hence slower than e^{-t} . By Theorem 7.3(1), the time modulus is preserved. The linear case ($\mu(r) = \Theta(r)$) is the boundary: $\epsilon(t) = e^{-t}$ matches the module’s intrinsic rate. L’Hôpital gives a polynomial prefactor ($\delta(t) \sim t e^{-t}$), changing the modulus by at most $O(\log r)$, which is absorbed into $\Theta(r)$. ◀

8 Conclusion

We have introduced bounded analog complexity theory, in which all state variables remain in compact intervals and the time variable t is the sole diverging resource. Bounded surrogate compilation transforms unbounded PIVPs into bounded ones preserving all computed limits, and a polynomial change of time scale sends BGP polynomial-trajectory-length-and-time computation to bounded polynomial- t computation. Constructive examples realize fine-grained time moduli from $\Theta(r)$ through $\Theta(r \log r)$ and $\Theta(r^n)$ to every level of the elementary exponential hierarchy. Closure under exponentiation demonstrates the computational richness of the bounded setting, with the output variable remaining strictly positive and thus directly realizable as a CRN species without dual-railing. The full GPAC-to-CRN pipeline preserves time complexity, via a low-pass filter analysis of the readout module.

The complexity developed here concerns computation of real numbers via the limit readout, and our compilation absorbs unbounded variables only insofar as they affect the asymptotic limit. Existing polynomial-ODE simulations of Turing machines in the BGP framework [9, 4, 8] encode the tape positionally in state variables that grow as 10^n in tape length n . Whether Turing-complete simulation can be carried out in entirely bounded polynomial state, in the rational-coefficient PIVP and mass-action CRN form of the present paper, is left open.

References

- 1 David F. Anderson and Badal Joshi. Chemical mass-action systems as analog computers: implementing arithmetic computations at specified speed. *arXiv preprint*, 2024. arXiv:2404.04396.
- 2 Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. The computational power of population protocols. *Distributed Computing*, 20(4):279–304, 2007. doi:10.1007/S00446-007-0040-2.
- 3 Olivier Bournez, Pierre Fraigniaud, and Xavier Koegler. Computing with large populations using interactions. In *Mathematical Foundations of Computer Science (MFCS)*, volume 7464 of *Lecture Notes in Computer Science*, pages 234–246. Springer, 2012. doi:10.1007/978-3-642-32589-2_23.

- 4 Olivier Bournez, Daniel S. Graça, and Amaury Pouly. Polynomial time corresponds to solutions of polynomial ordinary differential equations of polynomial length. *Journal of the ACM*, 64(6):38:1–38:76, 2017. doi:10.1145/3127496.
- 5 Olivier Bournez and Amaury Pouly. A universal ordinary differential equation. In *Proc. 45th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 107 of *LIPIcs*, pages 116:1–116:14, 2018. doi:10.4230/LIPIcs.ICALP.2017.116.
- 6 Ho-Lin Chen, David Doty, and David Soloveichik. Deterministic function computation with chemical reaction networks. *Natural Computing*, 13(4):517–534, 2014. doi:10.1007/S11047-013-9393-6.
- 7 Ho-Lin Chen, David Doty, and David Soloveichik. Rate-independent computation in continuous chemical reaction networks. *Journal of the ACM*, 70(3):1–61, 2023. arXiv:2107.13681.
- 8 François Fages, Guillaume Le Guludec, Olivier Bournez, and Amaury Pouly. Strong Turing completeness of continuous chemical reaction networks and compilation of mixed analog-digital programs. In *Computational Methods in Systems Biology (CMSB 2017)*, volume 10545 of *Lecture Notes in Computer Science*, pages 108–127. Springer, 2017. doi:10.1007/978-3-319-67471-1_7.
- 9 Daniel S. Graça, Manuel L. Campagnolo, and Jorge Buescu. Computability with polynomial differential equations. *Advances in Applied Mathematics*, 40(3):330–349, 2008. doi:10.1016/J.AAM.2007.02.003.
- 10 Daniel S. Graça and José Félix Costa. Analog computers and recursive functions over the reals. *Journal of Complexity*, 19(5):644–664, 2003. doi:10.1016/S0885-064X(03)00034-7.
- 11 Nicholas Haisler, Xiang Huang, Andrei N. Migunov, Khalid Mohammed, and Garrett Provence. A selective dual-railing technique for general-purpose analog computers. In *Unconventional Computation and Natural Computation (UCNC 2025)*, volume 16364 of *Lecture Notes in Computer Science*. Springer, 2025. doi:10.1007/978-3-032-15641-9_26.
- 12 Xiang Huang and Rachel N. Huls. Computing real numbers with large-population protocols having a continuum of equilibria. In *28th International Conference on DNA Computing and Molecular Programming (DNA 28)*, volume 238 of *LIPIcs*, pages 6:1–6:17. Schloss Dagstuhl, 2022. doi:10.4230/LIPIcs.DNA.28.7.
- 13 Xiang Huang, Titus H. Klinge, and James I. Lathrop. Real-time equivalence of chemical reaction networks and analog computers. In *25th International Conference on DNA Computing and Molecular Programming (DNA 25)*, volume 11648 of *Lecture Notes in Computer Science*, pages 37–53. Springer, 2019. doi:10.1007/978-3-030-26807-7_3.
- 14 Ker-I Ko. *Complexity Theory of Real Functions*. Birkhäuser, 1991.
- 15 Leonard Lipshitz and Lee A. Rubel. A differentially algebraic replacement theorem, and analog computability. *Proceedings of the American Mathematical Society*, 99(2):367–372, 1987.
- 16 Marian B. Pour-El. Abstract computability and its relation to the general purpose analog computer. *Transactions of the American Mathematical Society*, 199:1–28, 1974.
- 17 Claude E. Shannon. Mathematical theory of the differential analyzer. *Journal of Mathematics and Physics*, 20(1–4):337–354, 1941.
- 18 David Soloveichik, Matthew Cook, Erik Winfree, and Jehoshua Bruck. Computation with finite stochastic chemical reaction networks. *Natural Computing*, 7(4):615–633, 2008. doi:10.1007/S11047-008-9067-Y.
- 19 Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(1):230–265, 1936. doi:10.1112/PLMS/S2-42.1.230.
- 20 Klaus Weihrauch. *Computable analysis: an introduction*. Springer Science & Business Media, 2000.

A

 Deferred proofs

A.1 Proof of Theorem 3.4

Proof. Apply bounded surrogate compilation (Construction 3.1) to eliminate all unbounded variables, obtaining a bounded system in surrogate variables with time parameter τ . The physical time satisfies $s'(\tau) = U_{n,0}(\tau)$, $s(0) = 0$, where $U_{n,0} = 1/(1 + \bar{f}^n) \in (0, 1]$.

Adjoin s as an additional variable to the compiled system. This adds one equation $s' = U_{n,0}$ to a bounded polynomial ODE, and $s(\tau) < T^*$ for all τ , so the augmented system remains bounded.

We claim $s(\tau) \rightarrow T^*$ as $\tau \rightarrow \infty$. Since s is increasing and bounded above by T^* , the limit $L = \lim_{\tau \rightarrow \infty} s(\tau)$ exists. If $L < T^*$, then $\bar{f}(\tau) = \bar{f}(s(\tau)) \rightarrow \bar{f}(L)$, which is finite since the solution exists on $[0, T^*)$. But then $s'(\tau) = 1/(1 + \bar{f}(\tau)^n) \rightarrow 1/(1 + \bar{f}(L)^n) > 0$, contradicting the convergence of $s(\tau)$ to a finite limit. Hence $L = T^*$.

Therefore $s(\tau) \rightarrow T^*$ in a bounded PIVP, and T^* is GPAC-computable. ◀

A.2 Iterated-logarithm tower constructions

This appendix records the iterated-logarithm tower construction deferred from Section 5.

► **Construction A.1** (Iterated-logarithm tower). For $k \geq 0$, define inductively

$$w'_k = (1 - w_k)^2 \prod_{j=0}^{k-1} (1 - w_j), \quad w_k(0) = 0, \quad (11)$$

with the convention that the empty product ($k = 0$) equals 1. The readout variable is

$$x'_1 = (1 - x_1) \prod_{j=0}^k (1 - w_j), \quad x_1(0) = 0. \quad (12)$$

► **Proposition A.2** (Tower solutions). Define $L_0(t) = t$ and $L_k(t) = \log(1 + L_{k-1}(t))$ for $k \geq 1$. Then for each $k \geq 0$:

$$1 - w_k(t) = \frac{1}{1 + L_k(t)}.$$

In particular, as $t \rightarrow \infty$, $L_k(t) = \log^{(k)}(t) + O(1)$ (the k -fold iterated logarithm).

Proof. By induction on k . Base case $k = 0$: $w'_0 = (1 - w_0)^2$ has solution $w_0(t) = t/(1 + t)$, giving $1 - w_0 = 1/(1 + t) = 1/(1 + L_0(t))$. Inductive step: let $u_k = 1 - w_k$. Then $u'_k = -u_k^2 \prod_{j=0}^{k-1} (1 + L_j(t))^{-1}$. The chain rule gives $L'_k(t) = L'_{k-1}(t)/(1 + L_{k-1}(t)) = \prod_{j=0}^{k-1} (1 + L_j(t))^{-1}$, so $du_k/u_k^2 = -L'_k(t) dt$. Integrating: $1/u_k(t) = 1 + L_k(t)$. ◀

► **Theorem A.3** (Exponential-tower modulus at depth k). For each $k \geq 0$, the depth- k tower construction computes $x_1(t) \rightarrow 1$ with $1 - x_1(t) = \exp(-L_{k+1}(t))$, and time modulus $\mu(r) = \exp^{(k+1)}(r + O(1))$, the $(k+1)$ -fold iterated exponential.

Proof. By Proposition A.2, $\prod_{j=0}^k (1 - w_j) = L'_{k+1}(t)$, hence $x'_1/(1 - x_1) = L'_{k+1}(t)$ and $-\log(1 - x_1) = L_{k+1}(t)$. Precision e^{-r} then requires $L_{k+1}(t) \geq r$, i.e., $t \geq \exp^{(k+1)}(r - O(1))$. ◀

Depth k	Convergence profile $1 - x_1(t)$	Time modulus $\mu(r)$
-1 (no tower)	e^{-t}	$\Theta(r)$
0	$e^{-\log(1+t)} = (1+t)^{-1}$	$\Theta(e^r)$
1	$e^{-\log \log(1+t)}$	$\Theta(e^{e^r})$
k	$e^{-\log^{(k+1)}(t)}$	$\Theta(\exp^{(k+1)}(r))$

A.3 Proof of Theorem 7.3

Proof. Case (1) follows from Lemma 7.1: since $\delta(t) \sim \epsilon(t)$, the time to reach precision e^{-r} is determined by ϵ , not by the module.

Case (2) follows from Lemma 7.2: the dominant term is $C e^{-\alpha t}$, giving $\mu_Z(r) = r/\alpha + O(1)$.

For the double-inversion method, the same argument applies with α replaced by $\min(\alpha, 1/\alpha)$. ◀

A.4 Proof of Theorem 6.1

Proof.

ODE derivation. Set $x_1 = x - 1$ (shifted to have $x_1(0) = 0$); the tracking ODE $x'_1 = (x - 1) - x_1$ is a low-pass filter that drives $x_1 \rightarrow \alpha - 1$. Define $u = \ln(1 + x_1)$. By the chain rule, $u' = x'_1/(1 + x_1)$. Setting $v = x_1/(1 + x_1)$ (equivalently, $v = 1 - 1/(1 + x_1)$), we obtain $u' = (1 - v) x'_1$, since $1/(1 + x_1) = 1 - v$. Differentiating v by the quotient rule gives $v' = x'_1/(1 + x_1)^2 = (1 - v)^2 x'_1$. Finally, $z = e^{y \cdot u}$, so $z' = z(y' u + y u') = z(y' u + y(1 - v) x'_1)$. All right-hand sides are polynomial in the state variables (x_1, u, v, z) and the upstream variables (x, y) .

Convergence. As $t \rightarrow \infty$: $x_1 \rightarrow \alpha - 1$, $u \rightarrow \ln(\alpha)$, $v \rightarrow (\alpha - 1)/\alpha$, and the exponent $y \cdot u \rightarrow \beta \ln \alpha$, giving $z \rightarrow e^{\beta \ln \alpha} = \alpha^\beta$.

Well-definedness. The trajectory satisfies $1 + x_1(t) > 0$ for all $t \geq 0$: since $x_1(0) = 0$ and x_1 converges to $\alpha - 1 > -1$ (for $\alpha > 0$), the denominator in u and v never vanishes. ◀

A.5 Proof of Theorem 6.4

Proof. Since the construction operates entirely within the GPAC model (no dual-railing), the analysis is standard computable-analysis error propagation [14, 20] through four steps. Let $\varepsilon_\alpha(t) = |x(t) - \alpha|$ and $\varepsilon_\beta(t) = |y(t) - \beta|$.

Step 1: x_1 tracking error. Define $e_1(t) = x_1(t) - (\alpha - 1)$. From (6): $e'_1 = (x(t) - \alpha) - e_1$, $e_1(0) = -(\alpha - 1)$. This is a first-order linear ODE with rate 1, giving

$$|e_1(t)| \leq e^{-t} |\alpha - 1| + \int_0^t e^{-(t-s)} \varepsilon_\alpha(s) ds.$$

This is the low-pass filter of Section 7.2 with cutoff 1. By Lemma 7.1, $|e_1(t)|$ inherits the decay profile of $\varepsilon_\alpha(t)$.

Step 2: u and v errors. By the mean value theorem, $|u(t) - \ln \alpha| \leq |e_1(t)|/\alpha_{\min}$ and $|v(t) - (\alpha - 1)/\alpha| \leq |e_1(t)|/\alpha_{\min}^2$, where $\alpha_{\min} = \min_t (1 + x_1(t)) > 0$. These are proportional to $|e_1|$ with constant factors.

14:22 Bounded Analog Complexity

Step 3: Exponent error. $R(t) = y(t) u(t)$ targets $R^* = \beta \ln \alpha$. By the triangle inequality,

$$|R(t) - R^*| \leq |\ln \alpha| \varepsilon_\beta(t) + \frac{|\beta|}{\alpha_{\min}} |e_1(t)| + o(1).$$

Step 4: Output error. Since $z = e^R$ and $\alpha^\beta = e^{R^*}$, the mean value theorem gives $|z - \alpha^\beta| \leq e^{\max(R, R^*)} |R - R^*|$. For large t , there exists $C = C(\alpha, \beta) > 0$ such that $|z - \alpha^\beta| < e^{-r}$ whenever both $\varepsilon_\beta(t) < e^{-(r+C)}$ and $|e_1(t)| < e^{-(r+C)}$. Hence $\mu_{\alpha^\beta}(r) = \max(\mu_\alpha(r + C), \mu_\beta(r + C)) + O(1)$. ◀