# Limits of Symmetric Computation

**Anuj Dawar**
Department of Computer Science and Technology, University of Cambridge, UK

## Abstract

I survey recent work on symmetric computation. A number of strands of work, from logic, circuit complexity, combinatorial optimization and other areas have converged on similar notions of symmetry in computation. This write-up of an invited talk gives a whirlwind tour through the results and pointers to the relevant literature.

**2012 ACM Subject Classification** Theory of computation → Complexity theory and logic; Theory of computation → Finite Model Theory; Theory of computation → Complexity classes

**Keywords and phrases** Logic, Complexity Theory, Symmetric Computation

**Digital Object Identifier** 10.4230/LIPIcs.ICALP.2024.1

**Category** Invited Talk

**Funding** Research funded by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee: grant number EP/X028259/1.

## Introduction

In the 1980s, descriptive complexity was a new approach to the problems of complexity theory. It carried the hope that methods from logic, particularly finite model theory, could be deployed to settle the difficult questions of complexity. It was one of many promising approaches at the time but the hard problems of complexity proved resistant to all of them. An amusing article from ACM Sigact News in 1996 [24] imagines the many possible titles of a paper announcing a resolution of the P vs.NP question. One of them is through Immerman's approach to descriptive complexity which recasts the question of separting P from NP as the question of separating the expressive power of fixed point logic FP from existential second-order logic on *ordered structures*. This captures the essential gap between the promise of descriptive complexity and its delivery. The methods from finite model theory that it makes available for proving inexpressibility in logics such as FP work well on *unordered* structures. But, the correspondence with complexity classes works well on *ordered* structures.

A more recent viewpoint on this is that the inexpressibility results from finite model theory establish lower bounds for restricted, *symmetric* models of computation. This is exemplified by the results of [2, 21], which show that the logic FPC (fixed-point logic with counting) corresponds to a natural model of *symmetric circuits*. The logic FPC is a natural and powerful logic within P for which unconditional lower bounds have been proved (see [11] for an overview).

Understanding the inexpressibilty methods of descriptive complexity as lower bounds for symmetric models of computation leads to a number of interesting further directions of investigation, which I review in the present talk. In particular, I look at the following directions.

1. We can extend the expressive power of FPC by considering more powerful operations than counting, while remaining within P. These give rise to further notions of symmetric computation, essentially weakening the symmetry restriction. Recently lower bounds have been obtained for these as well.
2. We can investigate what efficient algorithms can be implemented within these symmetric models. It turns out that many natural algorithmic methods are symmetric and therefore subject to the lower bound methods of descriptive complexity.



3. We can look at symmetry as it arises in other models of computation, and see to what extent *asymmetry* is used as a resource. I illustrate this with two cases, that of linear programs and of arithmetic circuits.
4. We relate the lower bounds to other classifications of problems according to their symmetries. In particular, the very successful classification of constraint satisfaction problems into tractable and intractable ones is based on a different, but related notion of symmetry.

In the following, after a brief introduction of the relationship between FPC and symmetric circuits, I give a summary of each of the above directions. The main aim is to provide pointers to the relevant literature and to identify fertile directions for future work. Formal definitions, statements of the results and proofs may be found in the cited literature.

## FPC and symmetric circuits

The symmetry restriction we are considering is most clearly explained in a circuit model of computation. Recall that any decision problem $L \subseteq \{0,1\}^*$ can be seen as a family of Boolean functions $(f_n)_{n\in\omega}$ where $f_n : \{0,1\}^n \to \{0,1\}$. Moreover, each function $f_n$ can be described by a circuit $C_n$: a directed acyclic graph with $n$ inputs and gates labelled by Boolean functions from some fixed basis, such as $\wedge$, $\vee$, $\neg$ or extensions with a majority gate or threshold gates. The language $L$ is decidable in polynomial time if, and only if, it has such a family of circuits that is P-uniform. In other words, the circuit $C_n$ can be constructed in poly$(n)$ time. One fact to note here is that when we consider what Boolean functions we can use in the basis, we are restricted to *symmetric* functions. These are functions whose value depends only on the number of 1s and 0s in the input and not on the order in which the inputs appear. The functions $\wedge$, $\vee$, $\neg$, majority and threshold all have this property. This is necessary in order for the circuit to be defined as a DAG with no further structure on it, such as an order of the gates.

When we consider decision problems on structures such as graphs, we are typically interested in deciding properties that are invariant under graph isomorphisms. A key perspective of descriptive complexity is that we consider formalisms in which only such properties can be expressed. In the context of circuit complexity, we can consider a property $P$ of (directed) graphs as being given by a family of Boolean functions $(p_n)_{n\in\omega}$ where $p_n : \{0,1\}^{n\times n} \to \{0,1\}$ takes the adjacency matrix of a graph and maps it to 1 just in case the graph has property $P$. Such Boolean functions have natural symmetries in the sense that for any permutation $\pi$ of $[n]$, we have $p_n(x) = p_n(x^\pi)$ where the string $x^\pi$ is defined to have $x^\pi_{ij} = x_{\pi(i)\pi(j)}$. We say that a circuit $C$ computing $p_n$ is *symmetric* if this action of permutations $\pi \in S_n$ on the inputs always extends to an automorphism of the circuit $C$ itself.

The key result from [2] is that a graph property is definable in FPC (for a formal definition of FPC, I refer the reader to [11]) if, and only if, it is decidable by a $P$-uniform family of symmetric circuits. The result there is stated for circuits with threshold gates (indeed, just a majority gate would suffice) but, as observed in [21], adding further symmetric Boolean functions to the basis does not change it. Thus, we get quite a robust notion of symmetric polynomial-time computation and it corresponds exactly to definability in FPC. Moreover, while I have stated it here for graphs, it is proved more generally for finite relational structures.

The important aspect of the connection between FPC definability and decidability with symmetric circuits is that we have methods for proving inexpressibility results in FPC and these yield proofs of unconditional lower bounds for symmetric circuits. In particular, FPC definable classes of graphs exhibit stronger invariance conditions than just being closed under

isomorphism. This is made precise by considering the Weisfeiler-Leman (WL) equivalences. For a precise definition of the $k$-dimensional Weisfeiler-Leman equivalence, see [17, Sec. 2]. Here we just note that this is, for any fixed $k$, a coarser relation than graph isomorphism. The connection with FPC definability comes from the fact that for any property $P$ of graphs that is definable by a formula of FPC, there is a constant $k$ such that $P$ is invariant under $k$-dimensional WL equivalence.

Cai et al. [10] first showed that there is no fixed value of $k$ for which $k$-dimensional WL equivalence is the same as isomorphism, and this leads to a construction of a class of graphs in P which is not FPC definable. This fundamental construction has been at the heart of many lower bounds since. That is, most results showing that some property $P$ is not definable in FPC and therefore not decidable by symmetric circuits proceed by showing that $P$ is not invariant under $k$-dimensional WL for any fixed $k$. Graph properties for which this has been shown include Hamiltonicity and 3-colourability.

## Symmetric algorithms

The fact that we can prove unconditional lower bounds for classes of symmetric circuits would not be so interesting if these classes formed a very weak model of computation. It turns out, however, that many natural algorithmic techniques are in fact symmetric. First of all, it is worth recalling some of the original motivation for the interest of finite model theory in computer science, which came from the study of database query languages (see [31]). Languages for querying relational databases, based on first-order logic and its extensions, naturally give rise to symmetric algorithms (in the precise sense of symmetric circuits considered above) when automatically compiled (see, for instance, the connection to circuits given in [23, 29]). In this sense, FPC provides a good formal model of database query languages that extend the relational calculus with recursion and counting mechanisms. When FPC was first introduced [27] it was proposed as a possible language in which all polynomial-time decidable queries could be expressed. Even after Cai et al. showed that this was not the case, it was often said that all *natural* polynomial-time decidable properties are expressible in FPC. One way to understand this is that problems for which the obvious algorithm is in polynomial time can usually be formulated in FPC. However, the power of FPC, and hence of symmetric computation, is surprising and a number of problems for which the polynomial-time algorithms are far from trivial have been shown nonetheless to admit symmetric algorithms. A few are worth highlighting.

The most significant one is Grohe's monumental work [26] showing that any polynomial-time decidable property of graphs excluding some fixed minor is in FPC, and so invariant under $k$-WL equivalence for some fixed $k$. In [3], my co-authors and I show that the ellipsoid method for optimizing linear programs can be expressed in FPC, and so many natural combinatorial optimization problems have bounded WL dimension. In particular this is true of the problem of determining the size of a maximum matching in a graph. The result can be further extended to hierarchies of semi-definite programs [19, 6]. This shows that some of our most powerful techniques for constructing efficient algorithms can be implemented in a way that is symmetry preserving.

## Linear-algebraic extensions

While many powerful algorithmic techniques can be implemented symmetrically, there are some simple efficient algorithms that just cannot be symmetrized without an exponential blow-up. It has been observed that the construction of Cai et al. essentially shows that

the problem of solving systems of linear equations over a a finite field cannot be expressed in FPC [4]. It follows that the Gaussian elimination algorithm cannot be implemented symmetrically. Indeed, since linear algebra, and more generally equation-solving, provides a rich source of examples of problems that cannot be expressed in FPC [12], research in descriptive complexity has investigated extensions of this logic with linear-algebraic operators. The resultung logics are provably more expressive than FPC. Here I want to point to connections of these with symmetric circuits, and with approximations of isomorphism stronger than the WL equivalences.

The first proposed extension of FPC by means of linear-algebraic operators was *fixed-point logic with rank* (FPR), which allows for operators that compute the rank of a matrix over a finite field [14, 25]. The expressive power of this logic has been shown to be characterized by symmetric circuits with *rank gates* [21]. To make this work, we need to modify the definition of circuit. To be precise, the Boolean function computed by a rank gate is not a fully symmetric function and so we can no longer think of a circuit as a DAG. It needs to have additional structure to give a matrix structure to the inputs of a rank gate. This relaxed notion of circuit gives a weaker requirement of symmetry which can be formalized and used to give a circuit characterization of FPR.

In order to study the expressiveness of FPR, a strengthening of the family of WL equivalences was defined in [16] which we call the *invertible map* (IM) equivalences. The WL equivalences can be seen as giving, on a fixed graph $G$, a partition of the $k$-tuples of vertices that approximates the partition into orbits of the automorphism group. The $k$-WL partition is the coarsest partition of the $k$-tuples of $G$ into classes $P_1, \ldots, P_t$ satisfying a natural *stability condition*. This condition says that two tuples $\mathbf{u}$ and $\mathbf{v}$ in the same class $P_i$ cannot be distinguished by counting the number of substitutions we can make in them to get a tuple in class $P_j$. The $k$-IM equivalences (denoted $\equiv^k_{\mathrm{IM}}$) are similarly defined but with a different stability condition. This essentially amounts to saying that the partition $P_1, \ldots, P_t$ cannot be further refined using *linear algebraic operators* over fields of characteristic $p$ where $p$ is a prime from some fixed set $\Omega$. Technical details of the definition and characterization in terms of linear algebraic operators can be found in [13].

In a breakthrough result, Lichter [28] has shown that there is no constant $k$ for which $\equiv^k_{\mathrm{IM}}$ is the same as isomorphism. The implications of this construction for the expressive power of any extension of FPC with linear-algebraic operators are spelled out in [15].

## Symmetry and asymmetry in other models

Once we recognize symmetry as a feature of algorithms, it makes sense to identify how it appears in other models of computation, beyond logic and circuits. It is, of course, built in as a natural feature in the logics we study in descriptive complexity. And we have identified the corresponding notion in the context of circuits. One aim of identifying meaningful restrictions by symmetry in various models of computation is to see how asymmetry (or symmetry-breaking) is a resource and how it trades off with other computational resources. In this section, I aim to provide pointers to two specific models where we have obtained interesting insights by analysing symmetric restrictions to natural known computational paradigms.

The first is that of linear programming. I noted above that the ellipsoid method for solving linear programs can be implemented symmetrically. Moreover, when combinatorial problems are formualted as linear programs, the programs have natural symmetries inherited from the problem. Symmetric linear programs in this sense were studied by Yannakakis [32]. The focus there was on linear programs, or *extended formulations*. For example, consider

graphs over the vertex set $[n]$. We can consider these as functions $G : X \to \{0,1\}$ where $X = \{x_{ij} \mid i,j \in [n]\}$ is the set of potential edges. Equivalently, we can think of $G$ as a 0-1 valued vector in the Euclidean space $\mathbb{R}^X$. A collection of graphs is then a set $P \subseteq \{0,1\}^X$ and various graph optimization problems can be expressed as optimizing a linear function over $P$. If we can represent the convex hull $\text{conv}(P)$ as the projection of a polytope $Q \subseteq \mathbb{R}^{X \times Y}$ using additional variables $Y$, with a number of facets polynomial in $n$, we can solve these problems in polynomial time. Yannakakis proved that the travelling salesman and matching polytopes do not have such polynomial-size *symmetric* extended formulations. The notion of symmetry is the natural one. Any permutation of $[n]$ has a natural action on $X$ and hence on $\mathbb{R}^X$. The symmetry requirement says that for any such permutation $\pi \in S_n$ we can find a permutation $\sigma$ of $Y$ such that for $\mathbf{xy} \in R^{X \times Y}$, $\mathbf{xy} \in Q$ if, and only if, $\pi(\mathbf{x})\sigma(\mathbf{y}) \in Q$. While the lower bound proof of Yannakakis relies heavily on the notion of symmetry, it turns out that this is not essential to the result as Rothvoß [30] obtains exponential lower bounds without the assumption.

We can consider another way of representing the set $P \subseteq \{0,1\}^X$ as a linear program. We say that a polytope $\mathcal{P} \subseteq \mathbb{R}^X$ *recognizes* $P$ if $P \subseteq \mathcal{P}$ and $\{0,1\}^X \setminus P$ is disjoint from $\mathcal{P}$. Now, a class of graphs that is decidable in polynomial time necessarily is recognized by a polynomial-size family of extended formulations. But, what classes are recognized by *symmetric* such families? It turns out that they are exactly the classes of bounded WL dimension. In other words those that are recognized by (possibly non-uniform) families of polynomial-size symmetric circuits with threshold gates [5].

Another computational model where the assumption of symmetry has revealed remarkable structure is that of *arithmetic circuits*. Formally, an arithmetic circuit over a field $K$ and a set of variables $X$ is a directed acyclic graph where every input (i.e. node of indegree 0) is labelled by an element of $X$ or an element of $K$, and every internal node is labelled either $+$ (a sum gate) or $\times$ (a product gate). A distinguished *output* gate can then be seen as computing a polynomial in the ring $K[X]$. Two polynomials (strictly speaking they are families of polynomials) that are much studied in the field are the *determinant* and the *permanent*. They are both defined on a set of variables $X$ representing the entries of an $n \times n$ matrix, so $X = \{x_{ij} \mid 1 \leq i,j \leq n\}$. It is known that there are polynomial-size circuits for computing the determinant $\det(X)$ while it is conjectured that there are no polynomial-size circuits for computing the permanent $\text{perm}(X)$. Both $\det(X)$ and $\text{perm}(X)$ are invariant under permutations of the variables $X$ which are induced by the natural action of $S_n$. So, it makes sense to ask whether these polynomials can be computed by polynomial-size *symmetric* circuits. It turns out [20] that this is the case for the determinant but provably not for the permanent. Furthermore, these polynomials have symmetries that go beyond the action of $S_n$ simultaneously on rows and columns. For example, the permanent is invariant under *any* permutation of the rows and columns of the matrix and the determinant under separate permutations of the rows and columns that have the same sign. We are able to establish lower bounds for arithmetic circuits assuming these more stringent symmetry requirements [22]. This provides an interesting case study in the tradeoff between symmetry and other resources, in this case circuit size. The lower bounds are obtained by adapting proof methods from finite model theory, even though the connection to logic is now remote.

## Constraint satisfaction problems

One of the great breakthroughs in theoretical computer science in recent years was the dichotomy theorem for constraint satisfaction problems (CSP) proved independently by Bulatov [8] and Zhuk [33]. A specific CSP is given by a finite relational structure $\mathbb{D}$: the

domain $D$ along with a collection of relations $R_1, \ldots, R_m$ on it. An instance to be solved is specified by a similar relational structure $\mathbb{V}$: the set $V$ of variables and for each relation $R_i$, a set of tuples from $V$ whose interpretation must be in the relation. A solution is just a homomorphism from $\mathbb{V}$ to $\mathbb{D}$. It turns out that the computational complexity of determining whether a given instance is solvable is completely determined by the algebraic structure of the so-called *clone of polymorphisms* of $\mathbb{D}$ [9]. Looking at the polymorphism clones of a structure, rather than the automorphism groups is a different notion of symmetry which is more relevant when we are interested in the homomorphisms between structures. Nonetheless, there is a potentially intriguing relationship between the two notions of symmetry.

While the Bulatov-Zhuk dichotomy theorem classifies all CSP into two classes: those that are solvable in polynomial time and those that are NP-complete, for our purposes a trifurcation of CSP is interesting. That is, we can further subdivide the polynomial-time solvable problems into those that have *bounded width* and those that do not. The CSP of bounded width can be solved by means of a simple algorithm, known as *local consistency* [7]. This is parameterized by a natural number $k$. Since, for fixed $k$, the $k$-local consistency check is a polynomial-time algorithm, all bounded-width CSP admit a polynomial-time algorithm. However, the converse is not true. A number of CSP are known which are efficiently solvable, but do not have bounded width. It turns out that a CSP has bounded width if, and only if, the collection of satisfiable instances has bounded WL dimension (see [4, 7, 18]). Thus, there is apparently a close relationship between the $k$-dimensional WL approximation of isomorphism and the $k$-local consistency method for approximating homomorphism. This relationship is made precise in the category theoretic framework we developed in [1].

It remains an open question whether we can similarly characterize *all* tractable CSP, i.e. those with a near-unanimity polymorphism by some (perhaps tractable) approximation of isomorphism. Indeed, it is conceivable that the invertible map equivalences serve this purpose. Could it be that for every such CSP there is a $k$ such that the collection of satisfiable instances is invariant under $\equiv^k_{\mathrm{IM}}$? Converesely, could it be that for every CSP that does not admit such a polymorphism (i.e. those that we know to be NP-complete) is not invariant under $\equiv^k_{\mathrm{IM}}$ for any $k$?

## Conclusion

We have discovered that symmetry arises in many forms in the analysis of computation, and is an important property of structured data, such as graphs and also of algorithms that work on this data. Symmetry in algorithms arises naturally when algorithms are automatically generated from high-level specifications. At the same time, symmetry-breaking can be an important method to improve efficiency of algorithmic procedures, and this is demonstrated by the unconditional lower bounds we have for symmetric algorithms for some problems, where efficient symmetry-breaking algorithms are known. The emerging theory of upper and lower bounds for symmetric computation pulls together a number of distinct strands within theoretical computer science, and draws in diverse mathematical methods with many promising directions to follow.

## References

1 S. Abramsky, A. Dawar, and P. Wang. The pebbling comonad in finite model theory. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017*, 2017. doi:10.1109/LICS.2017.8005129.

2 M. Anderson and A. Dawar. On symmetric circuits and fixed-point logics. *Theory Comput. Syst.*, 60(3):521–551, 2017. doi:10.1007/s00224-016-9692-2.

3 M. Anderson, A. Dawar, and B. Holm. Solving linear programs without breaking abstractions. *J. ACM*, 62, 2015.

4 A. Atserias, A. Bulatov, and A. Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Computer Science*, 410(18):1666–1683, 2009.

5 A. Atserias, A. Dawar, and J. Ochremiak. On the power of symmetric linear programs. *J. ACM*, 68:26:1–26:35, 2021. doi:10.1145/3456297.

6 A. Atserias and J. Fijalkow. Definable ellipsoid method, sums-of-squares proofs, and the graph isomorphism problem. *SIAM J. Comput.*, 52:1193–1229, 2023. doi:10.1137/20M1338435.

7 L. Barto and M. Kozik. Constraint satisfaction problems solvable by local consistency methods. *J. ACM*, 61:3:1–3:19, 2014.

8 A. A. Bulatov. A dichotomy theorem for nonuniform csps. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

9 A. A. Bulatov, P. Jeavons, and A. A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM J. Comput.*, 34(3):720–742, 2005.

10 J-Y. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992.

11 A. Dawar. The nature and power of fixed-point logic with counting. *ACM SIGLOG News*, 2(1):8–21, 2015.

12 A. Dawar, E. Grädel, B. Holm, E. Kopczynski, and W. Pakusa. Definability of linear equation systems over groups and rings. *Logical Methods in Computer Science*, 9, 2013. doi:10.2168/LMCS-9(4:12)2013.

13 A. Dawar, E. Grädel, and W. Pakusa. Approximations of isomorphism and logics with linear-algebraic operators. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 112:1–112:14, 2019. doi:10.4230/LIPIcs.ICALP.2019.112.

14 A. Dawar, M. Grohe, B. Holm, and B. Laubner. Logics with rank operators. In *2009 24th Annual IEEE Symposium on Logic In Computer Science*, pages 113–122. IEEE, 2009.

15 A. Dawar, E. Grädel, and M. Lichter. Limitations of the invertible-map equivalences. *Journal of Logic and Computation*, 33:961–969, 2023. doi:10.1093/logcom/exac058.

16 A. Dawar and B. Holm. Pebble games with algebraic rules. *Fundam. Informaticae*, 150:281–316, 2017. doi:10.3233/FI-2017-1471.

17 A. Dawar and K. Khan. Constructing hard examples for graph isomorphism. *J. Graph Algorithms Appl.*, 23:293–316, 2019. doi:10.7155/JGAA.00492.

18 A. Dawar and P. Wang. A definability dichotomy for finite valued CSPs. In *24th EACSL Annual Conference on Computer Science Logic, CSL 2015*, pages 60–77, 2015.

19 A. Dawar and P. Wang. Definability of semidefinite programming and Lasserre lower bounds for CSPs. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, 2017. doi:10.1109/LICS.2017.8005108.

20 A. Dawar and G. Wilsenach. Symmetric Arithmetic Circuits. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 36:1–36:18, 2020. doi:10.4230/LIPIcs.ICALP.2020.36.

21 A. Dawar and G. Wilsenach. Symmetric circuits for rank logic. *ACM Transactions on Computational Logic (TOCL)*, 23:1–35, 2021.

22 A. Dawar and G. Wilsenach. Lower bounds for symmetric circuits for the determinant. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 52:1–52:22, 2022. doi:10.4230/LIPIcs.ITCS.2022.52.

23 L. Denenberg, Y. Gurevich, and S. Shelah. Definability by constant-depth polynomial-size circuits. *Information and Control*, 70:216–240, 1986.

24 S. A. Fenner, L.ance Fortnow, and W. I. Gasarch. Complexity theory newsflash. *SIGACT News*, 27:126, 1996. doi:10.1145/235666.571629.

25 E. Grädel and W. Pakusa. Rank logic is dead, long live rank logic! *The Journal of Symbolic Logic*, 84, March 2019.

26 M. Grohe. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory*. Lecture Notes in Logic. Cambridge University Press, 2017.

27 N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.

28 M. Lichter. Separating rank logic from polynomial time. *J. ACM*, pages 14:1–14:53, 2023.

29 M. Otto. The logic of explicitly presentation-invariant circuits. In *Computer Science Logic, 10th International Workshop, CSL '96, Annual Conference of the EACSL*, pages 369–384, 1996.

30 T. Rothvoß. The matching polytope has exponential extension complexity. In *Symp. Theory of Computing, STOC 2014*, pages 263–272, 2014.

31 V. Vianu. Databases and finite-model theory. In N. Immerman and Ph. G. Kolaitis, editors, *Descriptive Complexity and Finite Models*, volume 31 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 97–148. AMS, 1996. doi:10.1090/DIMACS/031/04.

32 M. Yannakakis. Expressing combinatorial optimization problems by linear programs. *J. Comput. Syst. Sci.*, 43(3):441–466, 1991.

33 D. Zhuk. A proof of the CSP dichotomy conjecture. *Journal of the ACM*, 67:1–78, 2020.