



# Combining Cubic Dynamical Solvers with Make/Break Heuristics to Solve SAT

Anshujit Sharma 

Department of Electrical and Computer Engineering, University of Rochester, NY, USA

Matthew Burns 

Department of Electrical and Computer Engineering, University of Rochester, NY, USA

Michael C. Huang 

Department of Electrical and Computer Engineering, University of Rochester, NY, USA

---

## Abstract

Dynamical solvers for combinatorial optimization are usually based on 2<sup>nd</sup> degree polynomial interactions, such as the Ising model. These exhibit high success for problems that map naturally to their formulation. However, SAT requires higher degree of interactions. As such, these quadratic dynamical solvers (QDS) have shown poor solution quality due to excessive auxiliary variables and the resulting increase in search-space complexity. Thus recently, a series of cubic dynamical solver (CDS) models have been proposed for SAT and other problems. We show that such problem-agnostic CDS models still perform poorly on moderate to large problems, thus motivating the need to utilize SAT-specific heuristics. With this insight, our contributions can be summarized into three points. First, we demonstrate that existing *make-only* heuristics perform poorly on scale-free, industrial-like problems when integrated into CDS. This motivates us to utilize *break* counts as well. Second, we derive a relationship between *make/break* and the CDS formulation to efficiently recover *break* counts. Finally, we utilize this relationship to propose a new *make/break* heuristic and combine it with a state-of-the-art CDS which is projected to solve SAT problems several orders of magnitude faster than existing software solvers.

**2012 ACM Subject Classification** Mathematics of computing → Probabilistic algorithms; Computer systems organization → Analog computers

**Keywords and phrases** Satisfiability, Ising machines, Stochastic Heuristics, Natural Computation

**Digital Object Identifier** 10.4230/LIPIcs.SAT.2023.25

**Supplementary Material** *InteractiveResource (Experimental data, benchmarks and solver infrastructure)*: <https://ising.ece.rochester.edu/sat>

## 1 Introduction

The slowdown of general purpose computing has given rise to novel architectures to solve NP-Hard problems. One such approach is to go beyond the von Neumann paradigm and leverage systems whose evolution under physical laws carries out certain type of computation efficiently. This approach has shown potential for success at least in combinatorial optimization problems. In this regard, most of the literature has focused on quantum computing: specifically on Quantum Annealing (QA) [12, 21] and Adiabatic Quantum Computing (AQC) [2]. Recently, another non-von Neumann approach: Ising machines, has been gaining traction. The state-of-the-art Ising machines work completely in the classical regime relying on extremely fast dynamics of the system. Hence, these are less sensitive to noise when compared to quantum computers. Some notable examples are using coupled oscillators [52], capacitors in a resistive network [1, 46, 57], and modulated pulses of light [31].



© Anshujit Sharma, Matthew Burns, and Michael C. Huang;  
licensed under Creative Commons License CC-BY 4.0

26th International Conference on Theory and Applications of Satisfiability Testing (SAT 2023).

Editors: Meena Mahajan and Friedrich Slivovsky; Article No. 25; pp. 25:1–25:21



Leibniz International Proceedings in Informatics

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

These approaches have shown extraordinary performance on the weighted MaxCut problem when compared to software based approaches [1, 27, 46]. However, applications to SAT have been less successful. This is due to ① the lack of support for super-quadratic interactions; and ② the failure to leverage problem-specific information.

In this work, we examine and alleviate these shortcomings so as to revive the fast solution-finding capabilities of Ising machines. We focus on the Ising model of computation, both due to its successful implementation as fast dynamical hardware accelerators [1, 31, 32, 52] and as algorithms [27]. We limit ourselves to 3-SAT due to problem reducibility and simplicity of discussion. We will specifically base our analysis using simulations of an Ising machine proposed recently [57], as it represents a near-term achievable piece of hardware.

The novel contributions of this work can be summarized into three points:

1. Demonstrating the shortcomings of previous super-quadratic solvers and heuristic proposals on uniform random and scale-free problems.
2. Deriving a relation between cubic dynamical formulations and the *make/break* counts of variables in a 3-SAT formula.
3. Proposing novel *make/break* heuristics by leveraging the cubic formulation and demonstrating their viability by comparing a simulated dynamics-based solver against state-of-the-art software SAT solvers.

The rest of the paper is organized as follows. Section 2 provides a background on the Ising model and related work. Section 3 introduces a cubic formulation for 3-SAT and demonstrates the shortcomings of problem-agnostic dynamical solvers. Building on this insight, Section 4 demonstrates and analyzes the inadequacy of existing *make-only* heuristics in solving scale-free problems. This motivates us to also utilize *break* counts. We then derive a relationship that enable us to easily recover *break* counts from the cubic formulation itself. We utilize this relationship to propose a new heuristic using both *makes* and *breaks*. In Section 5, we combine this heuristic with a state-of-the-art cubic dynamical system and compare it against existing CDCL and SLS solvers. Finally, we conclude our findings in Section 6 as well as propose future directions for research.

## 2 Background

### 2.1 Preliminaries

When discussing SAT formulas, we use the notation introduced in “The Handbook of Satisfiability” [13]. Unless otherwise specified, all problems discussed are in 3-SAT form.  $N$  and  $M$  refer to the number of variables and clauses respectively.  $x_n \in \{0, 1\}$  is used to denote an arbitrary variable, and  $\mathbf{a} \in \{0, 1\}^N$  is the full assignment vector. Uniform and scale-free problems used for testing are generated using the methodology described in a work by Ansótegui et al. [4, 5].

### 2.2 Quadratic Models: Ising and QUBO

The Ising model was originally formulated by Wilhelm Lenz and solved in a simplified form by his student Ernst Ising [38]. It describes a system of magnetic *spins* ( $s_i$ ), expressed in one dimension. Each spin takes the values,  $s_i = \pm 1$ . Each pair of spins ( $s_i, s_j$ ) is “coupled” with some coefficient  $J_{ij}$ . An external field  $h_i$  can also exist, which imposes some linear coefficient to the spins. The overall energy of an  $N$ -spin system is expressed via the *Hamiltonian*  $H$ :

$$H(s, J, h) = - \sum_{i < j}^N J_{ij} s_i s_j - \sum_i^N h_i s_i = -s^T J s - h^T s \quad (1)$$

An Ising system will seek a state  $s$  such that  $H(s)$  is minimized. Hence,  $J_{ij} > 0$  implies spin affinity, known as *ferromagnetism*:  $s_i$  will tend to equal  $s_j$ .  $J_{ij} < 0$  implies spin repulsion, known as *antiferromagnetism*:  $s_i$  will tend to equal  $-s_j$ .

When referring to Ising formulation parameters,  $s$  describes the complete spin state vector,  $J$  the complete coupling matrix. Subscripts will be added to indicate a single element, for instance  $s_i$  or  $J_{ij}$ . A state vector  $s^*$  for which  $H(s^*) = 0$  is referred to as a *ground state* of  $H$ . We can express any given state vector for a SAT encoded Hamiltonian as an assignment to the original CNF problem, where  $s_i = 1 \rightarrow x_i$ , and  $s_i = -1 \rightarrow \bar{x}_i$ .

Another equivalent formulation is the Quadratic Unconstrained Binary Optimization (QUBO) form, where variables  $x_i$  take values in  $\{0, 1\}$ . An Ising formula can be trivially transformed into a QUBO formula using the following replacement rule for spins:

$$s = 2x - 1 \tag{2}$$

Formulating SAT as a QUBO problem is cleaner than its equivalent Ising formulation. Hence, in our discussions, we will use QUBO formulas, where  $x_i = 1$  indicates “true” and  $x_i = 0$  indicates “false”, and  $\mathbf{a} \in \{0, 1\}^N$  denotes the variable assignment vector. For convenience, we will refer to a dynamics-based QUBO/Ising solver as a *quadratic dynamical solver* (QDS). A QDS can be implemented in a wide variety of mediums [31, 32, 52], but we will focus on a CMOS-compatible hardware proposed by Afoakwa et al. [1] called Bistable Resistively-coupled Ising Machine (BRIM). There are some compelling reasons to use BRIM as the baseline for comparison:

1. *CMOS compatibility*: Unlike quantum systems and many other Ising machines, BRIM is electronic-based and can be fabricated using today’s CMOS technology. This allows for easier extension of its design and integration with other heuristics to improve its performance. The proposed design is also more feasible and energy efficient in the near term as discussed in previous work [1].
2. *All-to-all connectivity*: Many Ising machine implementations have limited connectivity between spins which greatly limits its true capacity. To solve problems on such machines, one needs to transform the input graph into another (much bigger) graph that the hardware can map; a process called *embedding*. Embedding is NP-Hard in itself [20, 51]. BRIM supports all-to-all connectivity and thus, doesn’t suffer from this problem.
3. *Extremely fast dynamics*: Unlike variants of *Coherent Ising Machines* (CIM) [31] which rely on FPGA computation to *emulate* coupling, the time evolution of BRIM is completely done naturally based on physics. Thus, BRIM can achieve good solutions very quickly as is established in previous works [1, 46].

Specifics on the BRIM model used for simulation is explained later in Section 4.3.1 and the pseudocode can be found in Appendix A. Throughout this work, we assume a variant of BRIM model with quantized nodal interactions [57].

## 2.3 Related Work

### Physics-Based Optimization for SAT

Optimization literature has previously utilized physical computational methods. Myriad dynamical systems have been proposed which implement the quadratic Ising model to solve NP-Hard optimization problems, including time-evolving quantum systems [2, 24, 32], modulated optical pulses [31], coupled electronic oscillators [52], and resistively coupled capacitors [1]. These approaches have shown success in natively quadratic problems such as

graph MaxCut, however their application to SAT has been largely unsuccessful as we will see. As an example, in one work [26], the authors proposed to optimize the Maximal Independent Set (MIS) reduction of 3-SAT with quantum annealing. In another work [15], the authors demonstrated the theoretical feasibility of gate-based quantum computing using noise-free QAOA simulations. In the near-term, quantum computers suffer from noise-induced errors which severely limits its performance and scalability [7, 11, 47, 53].

Optimization software algorithms loosely inspired by physical dynamics have also been proposed. The best known is simulated annealing (SA) [35]. SA minimizes a given cost function by taking inspiration from metallurgical processes with gradually decreasing temperature. Other notable physics-inspired examples are evolutionary algorithms [6].

Algorithms which directly simulate physical phenomena for optimization have also been proposed. Among these, simulated bifurcation (SB) [27] and continuous-time dynamical system (CTDS) [23] are primary examples. The former simulates chaotically bifurcating Ising models, the latter a chaotic system with exponentially growing factors. A GPU implementation of CTDS was shown to outperform MiniSAT in certain large problems [40].

### Hardware SAT Solvers

Hardware acceleration of SAT algorithms broadly falls into two categories: total solvers (implementing an algorithm in its entirety) and subset accelerators (implementing specific operations of a SAT algorithm). The former generally implement SLS algorithms [30, 41, 48] due to their simpler heuristics. In one work, the authors proposed a novel combination of naturally stochastic “p-bit” hardware to accelerate a simplified variant of ProbsAT [48] (see Section 4.1). There are examples of total CDCL hardware [28], however complex heuristics preclude efficient implementation. Subset solvers are commonly Boolean constraint propagation (BCP) accelerators [22]. BCP is particularly computationally demanding in CDCL (upwards of 90% of computation time [22, 50]). The fixed proportion means BCP accelerators can *only* provide an 8-10× speedup by Amdahl’s law. Other novel solvers include an analog circuit proposed [56] and demonstrated [17] by implementing the aforementioned CTDS algorithm [23]. Interested readers are referred to an in-depth survey for a more thorough treatment of SAT hardware acceleration [50].

### High-degree Dynamical Solvers

The inability of classical quadratic models to generalize to higher-degree polynomial interactions has motivated a recent surge in Ising-like high-degree models<sup>1</sup>. Formulations of  $k$ -SAT and NAE-SAT have been proposed and simulated in various dynamical systems [9, 16]. The SB algorithm has been extended to support high-degree polynomial interactions [33] with potential applications to highly parallel SAT solvers. These models have been shown to significantly outperform the traditional quadratic reductions described in literature [36, 38, 44, 51]. However, as we will discuss in Section 3.2, models implementing problem-agnostic local minima escaping heuristics demonstrate poor solution quality for moderate to large problems.

---

<sup>1</sup> Note that, all the citations refer to high-degree polynomial interactions as “high-order”, not to be confused with *high-order logic*.

### 3 High-degree Polynomial Formulation

#### 3.1 Reformulating SAT

Finding the globally minimum energy of a QDS model is proven to be NP-Hard, allowing for a large number of problems to be reformulated as QDS Hamiltonians [38]. An initial proposal to reduce 3-SAT to a QUBO model was to formulate it as a “Maximal Independent Subset” (MIS) problem [38]. While MIS can be more naturally encoded into the QUBO model, encoding an  $M$  clause CNF requires  $3M$  QUBO spins, an unacceptable level of search-space bloat. Instead, we propose a cubic Hamiltonian  $H_C$  as a “natural” expression of 3-SAT. Thus, for a given  $M$ -clause CNF 3-SAT formula:

$$f = \bigwedge_{i=1}^M (\ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3}) \quad (3)$$

the proposed Hamiltonian  $H_C$  is given by:

$$H_C = \sum_{i=1}^M g(\ell_{i,1})g(\ell_{i,2})g(\ell_{i,3}) \quad (4)$$

where for each clause literal  $\ell_{i,j}$

$$g(\ell_{i,j}) = \begin{cases} (1 - x_n) & \ell_{i,j} = x_n \\ x_n & \ell_{i,j} = \bar{x}_n \end{cases} \quad (5)$$

Here  $x_n$  is the variable corresponding to the  $j^{\text{th}}$  literal of the  $i^{\text{th}}$  clause,  $n \in \{1, \dots, N\}$ . Each variable in a problem only has one associated spin, hence this formulation requires  $N$  spins to represent the complete CNF.  $k$ -SAT problems for  $k > 3$  have a natural analogue with higher-degree terms, as discussed in a recent work [16]. However, we limit our focus to 3-SAT.

We observe that  $g(\ell_{i,j}) \in \{0, 1\}$ , with  $g(\ell_{i,j}) = 0$  corresponding to true  $\ell_{i,j}$  and  $g(\ell_{i,j}) = 1$  corresponding to false  $\ell_{i,j}$ . A clause  $C_i = (\ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3})$  is satisfied if and only if at least one of its literals is true. Therefore, we observe:

► **Proposition 1** (Clause Satisfaction). *A clause  $C_i = (\ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3})$  is satisfied by the state vector  $\mathbf{a}$  if and only if  $g(\ell_{i,1})g(\ell_{i,2})g(\ell_{i,3}) = 0$ .*

Since the Hamiltonian  $H_C$  is a sum over such clause-derived products, we conclude:

► **Lemma 2** (Ground States are SAT).  *$\mathbf{a}$  is a ground state of  $H_C$  ( $H_C(\mathbf{a}) = 0$ ) if and only if  $\mathbf{a}$  satisfies the underlying CNF formula  $f$ .*

The logical equivalence between  $H_C$  and a given CNF formula makes  $H_C$  a desirable Hamiltonian for QDS implementation. However, the QDS model is limited to quadratic and linear terms only, and it lacks support for 3<sup>rd</sup>-degree polynomial (cubic) interactions which is required in  $H_C$ . Therefore, to implement  $H_C$  in a QDS format, one must “quadratize” all cubic terms by introducing extra auxiliary spins and extra terms to penalize undesirable states [25, 36, 44]<sup>2</sup>.

<sup>2</sup> We do note that, the MAX-2-SAT reduction of 3-SAT [54] does map naturally to the QDS format. However, mathematically, it is equivalent to the one proposed by Kolmogorov et al. [36] and thus, suffer from the same problems.

In a recent work [16], the authors have demonstrated a drop in solution quality after such “quadratization”. This motivated them to propose an Ising-formulated  $\{+1, -1\}$  high-degree polynomial model. With experimental analysis for uniform random problems, the authors showed its superior performance compared to QDS.

Throughout our discussion, we limit our scope to solvers with support for cubic interactions and refer to these as *cubic dynamical solvers* (CDS).

### 3.2 Problem Agnostic Heuristics

As with naive gradient descent, dynamics-based solvers (QDS and CDS) can be trapped in local minima in the energy landscape. Classical optimization algorithms such as simulated annealing (SA) [35] overcome such local traps by introducing stochastic behavior (occasionally accept a “bad” state). It has been proven that given enough time, SA can converge to the global optimum [39]. Thus, a common approach for hardware solvers is to mimic the behavior of the SA algorithm by introducing random noise to perturb the system. Example noise sources include varying external oscillations [52], random polarity flips [1], and transverse quantum field fluctuations [32]. The strength of these perturbations decrease over the course of the run according to an *annealing schedule*. The primary motivation of such a schedule is to broadly explore the search-space initially and then (hopefully) settle into some global optimum towards the end. In general, these perturbations are applied randomly across problem variables, taking into account no knowledge of problem type or structure.

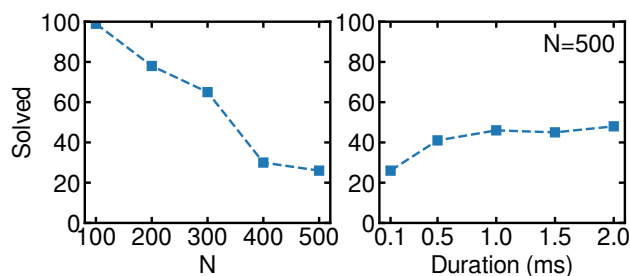
A recently proposed CDS used such problem agnostic perturbations [16], as have previous works proposing dynamics-based solvers [1, 26]. Here, we refer to this as the **Anneal** heuristic. Just like prior works on BRIM [1, 46], we use a linear annealing schedule to induce artificial spin flips. Let  $p_0$  and  $p_1$  be the start and end probabilities respectively. Then, at any given time  $t$  and a total run time of  $T$ , the instantaneous probability  $p$  to flip a spin is given by:  $p = p_0 - \frac{t(p_0 - p_1)}{T}$ . Thus, each spin is treated as an i.i.d. Bernoulli random variable with flip probability  $p$ .

Fig. 1 shows the performance of CDS with **Anneal** heuristic on 5 sets of 100 generated uniform random problems with  $M/N = 4.25$  and  $N \in \{100, 200, 300, 400, 500\}$ . The left figure shows the number of instances solved as  $N$  increases with a constant run time of 0.1 ms. The system easily finds solutions to 99% of the 100-variable problems, however the solved proportion drops quickly to 26% for  $N = 500$ . We also consider the possibility that the larger instances require longer system evolution times. Fig. 1 [right] shows the number of solved instances for the 500-variable set as run time is increased up to 2 ms ( $20\times$  longer). We observe only a minor increase in solved proportion to 48%. Parameter sweeps of  $p_0/p_1$  are also unable to improve solution quality for larger instances.

Theory suggests that all problems would be solved given a sufficiently long run time with a conservative, ergodicity-preserving schedule [39]. However, these results suggest that such schedules/anneal times would not represent a substantial speedup (if any) over other existing solutions. To give some context, WalkSAT is capable of solving the same 500 variable instances in as little as 5 ms. Therefore, we believe that such a problem-agnostic or “blind” annealing is insufficient.

## 4 Integrating SLS Heuristics

The reason for the comparatively poor performance of the problem-agnostic **Anneal** heuristic lies largely in the existence of this conference. SAT is a well-studied problem whose peculiarities are exploited by solver heuristics. CDCL-style clause learning uses Boolean



■ **Figure 1** Evaluation of CDS with a problem-agnostic `Anneal` heuristic. Left: Number of uniform random instances solved out of 100 with a constant run time of 0.1 ms as problem size increases. Right: Number of 500 variable instances solved as we increase the system run time.

resolution to actively store information on the problem at hand, and VSIDs have been shown to exploit large scale problem structures [37]. SLS algorithms such as WalkSAT [34, 45] include a veritable buffet of different heuristics for variable selection based on various scoring and ranking mechanisms.

Problem-agnostic dynamical systems have yet to adopt such advanced heuristics to exploit the existing body of SAT-specific knowledge. Prior implementations of BRIM used the `Anneal` heuristic. While sufficient for more unstructured problems (e.g. MaxCut [1, 32]), such annealing schedule falls short for a structured problem like 3-SAT requires. The results shown in Fig. 1 demonstrate the shortcomings of this uniform random heuristic. This motivates us to propose ProbSAT-like stochastic heuristics to increase search-space efficiency of CDS. We first discuss why it is a good idea to start with the ProbSAT heuristic, then move onto discussion of a hardware friendly variant and further improvements.

#### 4.1 The ProbSAT heuristic and its variant

ProbSAT [8] defines a probability distribution for all problem variables based upon an internally determined score. The score in turn depends on the variable’s *make* and *break* counts. We use standard definitions for these terms, restated here for clarity.

► **Definition 3 (Make Clause).** For a given assignment  $\mathbf{a}$ , the **make** count of variable  $x_n$  is the number of newly satisfied clauses after flipping  $x_n$ . We denote it as  $\mathcal{M}_n(\mathbf{a})$ .

► **Definition 4 (Break Clause).** For a given assignment  $\mathbf{a}$ , the **break** count of variable  $x_n$  is the number of newly unsatisfied clauses after flipping  $x_n$ . We denote it as  $\mathcal{B}_n(\mathbf{a})$ .

We observe that the net change in SAT clauses by flipping variable  $x_n$  with current assignment  $\mathbf{a}$  is  $\mathcal{M}_n(\mathbf{a}) - \mathcal{B}_n(\mathbf{a})$ . For simplicity, we omit the argument  $\mathbf{a}$  from subsequent uses.

Note that for a particular variable  $x_n$ , any currently UNSAT clause in which it appears is necessarily a *make* clause, while *break* clauses are a subset of currently SAT clauses. The original ProbSAT paper examined two candidate distribution functions: polynomial and exponential. For both classes of function, the authors found that using  $\mathcal{B}_n$  alone provided a more effective heuristic.

ProbSAT explores the search-space solely based upon the probability distribution, hence the logic is relatively simple compared to other SLS algorithms. Accordingly, a ProbSAT-style algorithm is suitable for hardware implementations [48, 49]. The original algorithm works sequentially, choosing random UNSAT clauses and sampling from its variables based on the

defined distribution. However, sequential flips are both impractical and undesirable for CDS integration, as they require complicated synchronization and information propagation across the system. Instead, it is easier to allow variables to flip in parallel (distributed manner).

Building on this motivation, researchers proposed a *Biased Random Walk* (BRW) heuristic as a modification to ProbSAT [48]. The approach allows parallel flips and relies solely on  $\mathcal{M}_n$ . The authors proposed two different update rules for the flipping probability, depending on  $\mathcal{M}_n$ : *linear* and *nonlinear*. The *linear* rule probability,  $p_l$  is parameterized by  $p_{step}$ , and for a particular variable  $x_n$ ,  $p_l(x_n) = \mathcal{M}_n \times p_{step}$ . The *nonlinear* rule has two parameters:  $p_{init}$  and  $p_{step}$ . The first *make* clause of a variable  $x_n$  sets  $p_{nl}(x_n) = p_{init}$  and every subsequent *make* contributes  $p_{step}$ . Our simulations of both rules using uniform random problems verified that the *nonlinear* rule produced much better results. Hence, we use this rule whenever we refer to BRW<sup>3</sup>.

## 4.2 Combining BRW with CDS

In the original paper [48], the authors implemented the BRW algorithm in hardware using stochastic Magnetic Tunnel Junction (MTJ) devices to represent the variables. It turns out,  $\mathcal{M}_n$  are easy to calculate in such a parallel system. The authors proposed a digital circuit which couples the states of individual variables to determine clause state. Instead of designing a stand-alone BRW hardware, we propose a CDS system, cBRIM (BRIM with support for cubic terms), so as to leverage its fast dynamics (spins could flip every 20 ps [1]), along with a BRW-like heuristic as its annealing algorithm. We refer to this solver as cBRIM-BRW. The details of cBRIM are explained later in Section 4.3.1.

Experimental analyses demonstrated the superior performance of an MTJ-based BRW solver over WalkSAT and ProbSAT [48]. However, the experiments were only done with uniform random problems. To consider problems of a heterogeneous structure, we consider the variable interaction graph (VIG) of the formula. The problem variables form the VIG vertex set, with edges connecting variables co-appearing in a clause<sup>4</sup>. The VIG was originally introduced as a means of reasoning about CDCL resolution and algorithm behavior [43], but has since become a standard means of reasoning about problem structure [3]. It has been established that industrial problem VIGs exhibit *scale-free* structure [4,5]. Specifically, the number of clauses each variable appears in (and consequently the vertex degree in the VIG) follows a power-law distribution. In such a model, the probability of any given node to have degree  $d$  has the form:  $P(d) \propto d^{-\beta}$ . Industrial instances generally have  $\beta \in [2, 3]$  compared to  $\beta > 18$  for uniform random problems. This implies that there is a rapid fall in the probability of high-degree variables for uniform random problems and hence relatively low degree variance ( $< 0.3 \times$  the average degree) compared to that of industrial ( $> 1.5 \times$ ) [4].

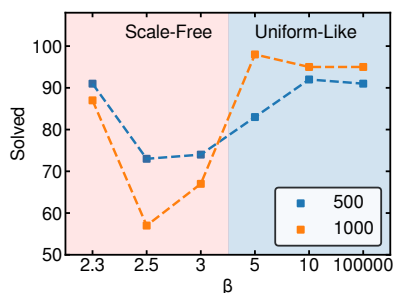
Consequently, real-world problems do not follow the Erdős-Renyi model used to generate the uniform random instances. Therefore, we simulate cBRIM-BRW and run custom generated power-law problems of 500 and 1000 variables with different values of  $\beta$  parameter [5]. For all the runs, we set  $p_{init} = 0.03$ ,  $p_{step} = 0.2$  for scale-free problems, and  $p_{init} = 0.07$ ,  $p_{step} = 0.9$  for uniform-like problems: both tuned for the best performance on 500 variable instances.

Fig. 2 shows how many problems cBRIM-BRW solved given different  $\beta$  values. We can observe that, as  $\beta$  increases (problems become more uniform-like), the performance of cBRIM-BRW improves generally. This is expected as BRW is demonstrated to work well

<sup>3</sup>  $p_l$  or  $p_{nl}$  can be greater than 1. The authors didn't elaborate, but presumably the probability is capped at 1.

<sup>4</sup> This differs slightly from the bipartite graph form described in another work [5], however the significance of vertex degree is equivalent.





■ **Figure 2** Number of problems solved by cBRIM-BRW out of 100 power-law distributed problems with 500 and 1000 variables for increasing  $\beta$ . We set cutoff run time of 0.11 ms and 1.1 ms for 500 and 1000 variables respectively.

for uniform random problems [48]. However, when  $\beta \in [2.5, 3]$ , the performance degrades. Moreover, the parameters tuned for 500 variable scale-free problems did not work well on larger problems: particularly in the case of  $\beta = 2.5$ . Thus, the optimal choice of parameters in cBRIM-BRW appears to be sensitive to the size of scale-free problems, far from ideal for practical use. These results lead us to conjecture that  $\mathcal{M}_n$  alone may not contain sufficient information to determine advantageous flips. Therefore, we propose to leverage both  $\mathcal{M}_n$  and  $\mathcal{B}_n$  in a stochastic heuristic.

### 4.3 Towards a better heuristic

A primary consideration for algorithms like BRW to use only  $\mathcal{M}_n$  is that  $\mathcal{B}_n$  is much harder to obtain in hardware. It relies both on problem structure and current variable assignments, imposing more hardware complexity. However, it turns out, the CDS formulation does provide some information that we can leverage to get  $\mathcal{B}_n$  without requiring complex hardware.

#### 4.3.1 Recovering Break Counts

Here, we discuss how to leverage inherent information from the CDS formulation to recover  $\mathcal{B}_n$ . Let us denote  $H_{C_n}$  as the Hamiltonian of all the clauses involving the variable  $x_n$ :

$$H_{C_n} = \sum_{x_n \in C_i} g(\ell_{i,1})g(\ell_{i,2})g(\ell_{i,3}) \quad (6)$$

where we denote  $\ell_{i,2}$  and  $\ell_{i,3}$  as the non- $x_n$  literals appearing in each clause and  $\ell_{i,1} = \{x_n, \bar{x}_n\}$  without loss of generality. From  $H_{C_n}$ , we now consider only the terms containing  $x_n$ . Recall from the definition of  $g(\ell_{i,j})$  in Eq. 5 that these terms will have the form  $\pm x_n g(\ell_{i,2})g(\ell_{i,3})$ .

► **Definition 5** ( $x_n$ -Relevant Form). *A clause  $C_i = (\ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3})$  where  $\ell_{i,1} \in \{x_n, \bar{x}_n\}$  has  $x_n$ -relevant form  $\mathcal{R}_{i|x_n} = \pm g(\ell_{i,2})g(\ell_{i,3})$*

We now define the state of a clause containing  $x_n$ , independent of  $x_n$ 's assignment.

► **Definition 6** ( $x_n$ -UNSAT Clause). *Let clause  $C_i = (\ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3})$  where  $\ell_{i,1} \in \{x_n, \bar{x}_n\}$ .  $C_i$  is  $x_n$ -UNSAT iff the subclause  $(\ell_{i,2} \vee \ell_{i,3})$  is UNSAT.*

Suppose  $C_i$  contains the literal  $x_n$ . Then  $\mathcal{R}_{i|x_n} = -g(\ell_{i,2})g(\ell_{i,3})$ . Furthermore, if  $C_i$  is  $x_n$ -UNSAT, then  $\mathcal{R}_{i|x_n} = -1$  and 0 otherwise. On the contrary, if  $C_i$  contains the negated literal  $\bar{x}_n$ , then  $\mathcal{R}_{i|x_n} = g(\ell_{i,2})g(\ell_{i,3})$ , with  $\mathcal{R}_{i|x_n} = 1$  in the case that  $C_i$  is  $x_n$ -UNSAT.

We now connect these definitions with previous notions of *make* and *break* clauses. Note that a clause can only be a *make* or *break* with respect to  $x_n$  if it is  $x_n$ -UNSAT.

► **Observation 7.** *If  $x_n = 1$ , then all clauses with  $\mathcal{R}_{i|x_n} = 1$  are makes, and all clauses with  $\mathcal{R}_{i|x_n} = -1$  are breaks. If  $x_n = 0$ , then all clauses with  $\mathcal{R}_{i|x_n} = -1$  are makes, and all clauses with  $\mathcal{R}_{i|x_n} = 1$  are breaks.*

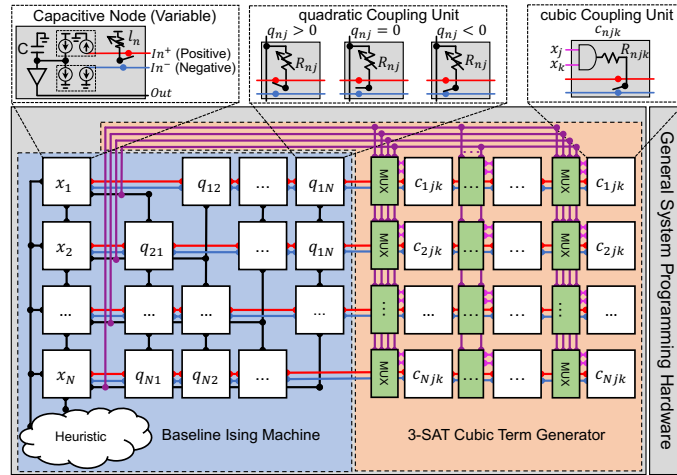
This leads us to the following Lemma:

► **Lemma 8.** *For some variable  $x_n$ ,*

$$\sum_{x_n \in C_i} \mathcal{R}_{i|x_n} = (1 - 2x_n) \times (\mathcal{B}_n - \mathcal{M}_n) = \begin{cases} \mathcal{M}_n - \mathcal{B}_n & x_n = 1 \\ \mathcal{B}_n - \mathcal{M}_n & x_n = 0 \end{cases}$$

Before stating the final theorem, we need to introduce just the basic principle of how our proposed CDS system, cBRIM, works.

### Cubic BRIM (cBRIM) design



■ **Figure 3** High level system overview of proposed cBRIM. The blue part is the baseline BRIM, while the orange part is new hardware to support cubic terms.

We extend BRIM [1, 46, 57] to support cubic terms for 3-SAT and refer to it as cubic BRIM or cBRIM in short. The full hardware details are beyond the scope of this paper and a number of variations in circuit design can achieve similar behavior at the system level. Here, we only discuss a high-level behavior model as shown in Fig. 3 and the pseudocode for simulation is shown in Appendix A. The general idea is to construct the hardware that naturally minimizes the cubic Hamiltonian  $H_C$ . The baseline system consists of an array of  $N$  bi-stable capacitive nodes whose voltages  $\in [0, 1]$  represent the variables.<sup>5</sup> Nodes are interconnected by programmable resistive coupling units  $q$ . The resistance of a coupling unit connecting node  $n$  and node  $j$ , is given by:

<sup>5</sup> Note that the original BRIM uses a virtual ground ( $\frac{V_{dd} + V_{ss}}{2}$ ). The voltage of the nodal capacitor ranges from  $V_{ss}$  to  $V_{dd}$ . Relative to the virtual ground, the polarity of the voltage serves as the spin representation ( $\pm 1$ ). In our current design, we do not use a virtual ground and the voltage quantizes to bit representation  $\{0, 1\}$ . This is because the bit representation is more convenient for SAT problems.

$$R_{nj} = \frac{R}{|q_{nj}|} \quad (7)$$

Where  $R$  is a constant resistance and  $q_{nj}$  is normalized to  $[-1, +1]$ . Thus, strong coupling means lower resistance. The sign of  $q_{nj}$  is implemented as follows: ① when  $q_{nj} > 0$ , the nodes are coupled in a parallel fashion (output of one node is connected to the positive input terminal of the other, via resistance  $R_{nj}$ :  $Out_n \rightarrow In_j^+$ ,  $Out_j \rightarrow In_n^+$ ); ② when  $q_{nj} < 0$ , they are coupled in an antiparallel fashion ( $Out_n \rightarrow In_j^-$ ,  $Out_j \rightarrow In_n^-$ ); ③ nodes are disconnected if  $q_{nj} = 0$ . Each node  $n$  can also have a linear bias  $l_n$ .

To support cubic terms, we add extra coupling units  $c$ . The idea is to pass voltages  $x_j$  and  $x_k$  through an AND gate to get  $x_{jk} = x_j \times x_k$ . Then we apply  $x_{jk}$  across a resistance of  $R_{njk} = (\frac{R}{|c_{njk}|})$  and feed the resulting current to node  $n$ . The inputs to the AND gate are also programmable via multiplexers (MUX) depending on the 3-SAT formula. Now, applying Kirchoff's current law, the rate of change of variable  $x_n$  is determined by the total incoming current via coupling resistors as described below:

$$\frac{dx_n}{dt} = \frac{1}{RC} \times (l_n + \sum_j q_{nj}x_j + \sum_{j<k} c_{njk}x_jx_k) \quad (8)$$

where  $C$  is the nodal capacitance. Moreover, as shown in Fig. 3, the variables can also be perturbed according to a selected heuristic of choice. If this perturbation is just a Gaussian white noise  $\sim \mathcal{N}(0, \sigma^2)$ , then the system is governed by Langevin dynamics (see Appendix B for more details). Now, consider the Hamiltonian  $H_C$  defined in Eq. 4. We can expand the summation and combine same degree polynomial terms together as shown below:

$$H_C = constant + \sum_n L_n x_n + \sum_{n<j} Q_{nj} x_n x_j + \sum_{n<j<k} C_{njk} x_n x_j x_k \quad (9)$$

where  $L_n$ ,  $Q_{nj}$  and  $C_{njk}$  are respectively the coefficients of linear, quadratic, and cubic terms. If we take the gradient of  $H_C$  with respect to  $x_n$ , we get,

$$\frac{\partial H_C}{\partial x_n} = L_n + \sum_j Q_{nj} x_j + \sum_{j<k} C_{njk} x_j x_k \quad (10)$$

Comparing Eq. 8 and Eq. 10, if we set  $l_n = -L_n$ ,  $q_{nj} = -Q_{nj}$  and  $c_{njk} = -C_{njk}$ , then we can write,

$$\frac{dx_n}{dt} = -\alpha \frac{\partial H_C}{\partial x_n} \quad (11)$$

where  $\alpha = \frac{1}{RC}$  is a known constant. Such a system is Lyapunov locally asymptotically stable [1] because it satisfies the criterion,  $\nabla H_C \cdot \frac{dx}{dt} < 0$ , unless  $x$  is a local minimum. Therefore, cBRIM naturally seeks a local minimum of  $H_C$  and stays there when found (unless perturbed by some heuristic). Moreover, using Definition 5 and the fact that  $x_n$ -Relevant forms are derived only from terms containing  $x_n$ , we can further write:

$$\frac{dx_n}{dt} = -\alpha \frac{\partial H_C}{\partial x_n} = -\alpha \sum_{x_n \in C_i} \mathcal{R}_{i|x_n} \quad (12)$$

Thus, the time dependent trajectory of each variable  $x_n$  is determined by the sum of all  $x_n$ -Relevant forms. Now, we can state the theorem that connects the system evolution of cBRIM with  $\mathcal{M}_n/\mathcal{B}_n$  of each variable.

► **Theorem 9.** Using Lemma 8 in Eq. 12,  $\frac{dx_n}{dt} = \alpha(1 - 2x_n) \times (\mathcal{M}_n - \mathcal{B}_n)$

From Eq. 8, cBRIM provides us the quantity  $(\mathcal{M}_n - \mathcal{B}_n)$  in Theorem 9 as the total incoming current to each variable  $x_n$ . Thus, coupled with the hardware to generate  $\mathcal{M}_n$  as in cBRIM-BRW, we can recover  $\mathcal{B}_n$  information without explicitly considering other variable states. This simplifies the hardware and takes advantage of the dynamical interactions between nodes. We now propose a new heuristic to leverage this.

### 4.3.2 The tanh-make-break (TMB) heuristic

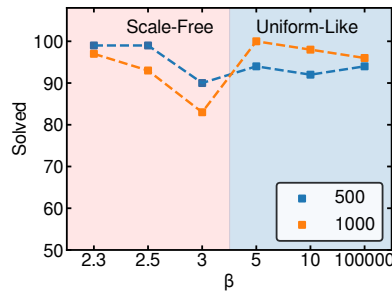
Motivated by the shortcomings of BRW as discussed in Section 4.2, we propose a heuristic that utilizes both  $\mathcal{M}_n$  and  $\mathcal{B}_n$ . The probability to flip a variable  $x_n$  is given by Eq. 13 using two nonlinear functions  $f$  and  $h$ :

$$p(x_n) = f(x_n) \cdot h(x_n) \quad (13)$$

In our testing, we selected  $f(x_n) = \tanh(c_m \cdot \mathcal{M}_n)$  and  $h(x_n) = 1 - \tanh(c_b \cdot \mathcal{B}_n)$ . Here,  $c_m > 0$  and  $c_b > 0$  are the *make* and *break* coefficients respectively. A number of nonlinear functions could work, and we leave exhaustive exploration as future work. The motivation behind the choice of tanh is threefold:

- Convenience: Given the range  $(\tanh(y) \in [0, 1])$  for  $y \geq 0$ , the product  $f(x_n) \cdot h(x_n)$  is mathematically a convenient probability without any need for normalization. Physically, it is easy to design a simple circuit with tanh-like behavior.
- SAT preservation: If all clauses are SAT, then  $\tanh(c_m \cdot \mathcal{M}_n) = 0$  for all  $x_n$ . Therefore, the system will not be perturbed once a satisfying state is reached.
- Nonlinearity: Previous heuristics using  $\mathcal{M}_n$  and  $\mathcal{B}_n$  utilized nonlinear functions with a steep slope [8, 48]. It was shown to significantly outperform linear functions. Therefore, we seek to preserve this characteristic.

We combine this *tanh-make-break* (TMB) heuristic with cBRIM and refer to this solver as cBRIM-TMB. Empirical observations suggest TMB heuristic is not overly sensitive to parameters. For instance,  $c_m = 0.9$ ,  $c_b = 0.4$  works best for the tested scale-free problems while for uniform random problems,  $c_m = 0.9$ ,  $c_b = 0.6$  works the best. Moreover, we found that the same  $c_m$  and  $c_b$  parameters worked for both 500- and 1000-variable problems.



■ **Figure 4** The performance of cBRIM-TMB tested on the same problems as in Fig. 2 and the same cutoff times. The plot shows the number of problems solved (out of 100) for varying values of  $\beta$ .

Fig. 4 demonstrates the performance of cBRIM-TMB which can be directly compared against that of cBRIM-BRW shown in Fig. 2. Unlike cBRIM-BRW, cBRIM-TMB is able to solve most of the industrial-like scale-free problems as well as uniform random problems with only

a minor dip in performance when  $\beta \approx 3$ . Moreover, it is able to maintain high success rates using the same parameters for both 500 and 1000 variable problems, indicating a weaker dependence on problem size for parameter tuning unlike `cBRIM-BRW`.

We leave it as future work to analyze the time dependent solver behavior, and to relate the performance of heuristics with problem structure. In the next section, we compare `cBRIM-TMB` with state-of-the-art CDCL and SLS solvers on generated scale-free and uniform random problems to estimate its real-world efficacy.

## 5 Evaluation

We now compare the performance of `cBRIM-BRW` and our proposed `cBRIM-TMB` against three state-of-the-art solvers: one CDCL based: KISSAT [14] and two SLS based: WalkSAT [34] and ProbsAT [8]. CDCL solvers have consistently outperformed other paradigms in SAT competitions, particularly in large industrial benchmarks. KISSAT and its variants are paradigmatic of high performance CDCL, placing top in recent SAT competitions [10, 14]. WalkSAT is representative of flip-efficient SLS heuristics, both in standalone solvers and those integrated into hybrid CDCL-SLS approaches [34]. Comparing against both algorithms will provide further context in which to evaluate `cBRIM-TMB`'s performance. We also choose to compare with the original ProbsAT solver since our heuristic is derived from it.

### 5.1 Methodology

Results of all software solvers: KISSAT [14], WalkSAT [34], and ProbsAT [8] are collected on an Intel Xeon Platinum 8268 CPU running at 2.90GHz. Each process is granted 8 GB of memory. KISSAT is allowed to run with a timeout of 10,000 seconds. WalkSAT is run with a 5 million flips cutoff with 20 retries using the `-best` heuristic (otherwise known as “SKC”) [45]. ProbsAT is used with its default configuration, with no cutoff specified. `cBRIM-TMB` and `cBRIM-BRW` are simulated by solving differential equations that describe the system dynamics. Each problem is simulated 20 times with different initial conditions. We start with 1 ms simulation time, up to 200 ms for unsolved instances.

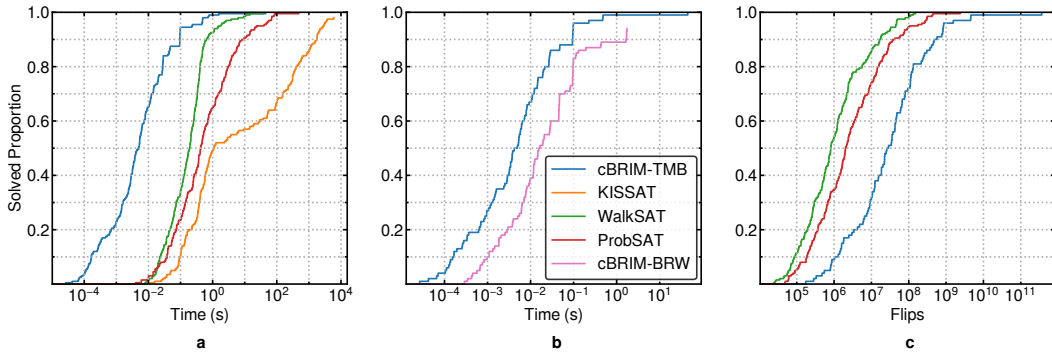
A 200-problem suite of 1000-variable instances were generated using a tool developed by Ansótegui et al. [4]. 100 are uniform random with  $M/N = 4.25$  (4250 clauses) and 100 are scale-free with  $M/N = 3.4$  (3400 clauses) and  $\beta = 2.935$ , which is observed to produce satisfiable problems about 50% of the time, with many being “hard” instances. All problems are verified as satisfiable using either WalkSAT or KISSAT. Instances are limited to 1000 variables and `cBRIM` run times are kept short due to the large simulation overhead of CDS (about  $10^5 \times$  slower than real hardware time).

For the reported solver time and flips, we use a “metric-to-solution” (MTS) formula, which is a generalization of “time-to-solution” [29] and is given by:

$$MTS = \begin{cases} m \times \frac{\log_{10}(0.01)}{\log_{10}(1 - S)} & S < 0.99 \\ m & \text{otherwise} \end{cases} \quad (14)$$

where,  $m$  is the average value of the metric of interest (time or flips) and  $S$  is the success probability of finding a satisfying solution for a single problem. Thus, MTS tells us the estimated metric value needed to find a solution with 99% probability, consequently penalizing solvers finding solutions with low success rate. The reported flips and times for WalkSAT, `cBRIM-TMB`, and `cBRIM-BRW` are then “flips-to-solution”(FTS) and “time-to-solution” (TTS) respectively. The flip counts used to calculate FTS for `cBRIM-TMB` and `cBRIM-BRW` are the sum of “heuristic flips” and “natural flips”. The latter are purely due to system dynamics, excluding that of the added heuristic.

## 5.2 Results



■ **Figure 5** Solver performance comparisons on 200 randomly generated 1000-variable problems: 100 scale-free with  $M = 3400$  and 100 uniform random with  $M = 4250$ . (a) The proportion of problem instances solved by each solver as time increases. Reported times for cBRIM-TMB and WalkSAT are calculated using the MTS formula in Eq. 14. (b) Comparing the performance of cBRIM-TMB and cBRIM-BRW on the 100 scale-free problems as time increases. (c) The same plot but with increasing flip count for the SLS solvers and cBRIM-TMB. Flip counts are similarly scaled according to Eq. 14.

First, we tested cBRIM with a problem-agnostic annealing schedule which could only solve 7% of the 200 problems with a TTS of 0.59 s in the worst case. Due to this poor performance and the requirement of ever longer run times, we have decided to remove it from the main analysis. Fig. 5a shows the proportion of problems solved by cBRIM-TMB and the software solvers versus execution time. cBRIM-TMB and the SLS solvers were able to solve all 200 problems. In contrast, KISSAT is able to solve all scale-free problems quickly, but is unable to solve 4 uniform random problems before the cutoff. Both SLS solvers outperformed KISSAT in the majority of the instances. However, 23 of the 100 scale-free problems are solved faster by KISSAT, with a geometric mean speedup of  $2\times$ . This indicates that some aspects of industrial problem structure is captured in the benchmark suite which KISSAT is able to exploit.

We also observe that for all the solvers, the curves flatten as we increase the run time. For cBRIM-TMB, 199 of the 200 problems could be solved in just 1.56 s in the worst-case, while the remaining scale-free problem took  $28.8\times$  longer. WalkSAT and ProbSAT have a similar trend in terms of run time increase, but it is not as dramatic. For instance, 199 problems took 14.53 seconds and 86.84 seconds respectively, while the last problem took  $2.6\times$  and  $5.6\times$  longer respectively. Like cBRIM-TMB, ProbSAT struggled with a scale-free problem while a uniform random problem proved difficult for WalkSAT. KISSAT could solve 50% of the instances within 1 s with majority of them being scale-free. The uniform random problems that it could solve, took orders of magnitude longer in the worst-case scenario. It is worth noting that larger scale-free problems likely favor CDCL solvers over SLS counterparts [10]. Therefore, the performance picture of CDS such as cBRIM-TMB for large real industrial instances is incomplete. Nevertheless, the results in Fig. 5a portray CDS as a promising avenue for execution time reduction.

Next, we compare the two versions of cBRIM. We find that both cBRIM-TMB and cBRIM-BRW perform similarly well for uniform random problems (except that cBRIM-BRW could not solve 1 problem instance). Therefore, Fig. 5b only shows the performance on the 100 scale-free problems with increasing run time. We can observe that cBRIM-TMB is consistently better than cBRIM-BRW by about  $4\times$  and moreover, the latter could not solve 6 out of the 100 scale-free problems. This emphasizes the importance of considering  $\mathcal{B}_n$  for industrial-like problems.

Another interesting analysis is to look at how efficiently these solvers arrive at a solution. Fig. 5c shows the proportion of problems solved as flip counts increase for **cBRIM-TMB** and the SLS solvers. Both WalkSAT and ProbSAT do outperform **cBRIM-TMB**. WalkSAT in particular stands out as the most “flip-efficient”. This is to be expected: a system doing parallel flips can effect many more state changes than sequential algorithms. Several variables can flip in a single move, compared to just one for WalkSAT or ProbSAT. The redundancy of parallel flip schemes with respect to  $\mathcal{M}_n/\mathcal{B}_n$  is another area for future study and optimization. In any case, the extremely fast dynamics of **cBRIM-TMB** more than compensates for higher flip counts which results in its observed speedup. This will become even clearer in the following analysis.

■ **Table 1** Summary of the results shown in Figure 5 for each problem type. The time and flip values are the geometric mean of all successful solutions for each solver scaled by the MTS formula in Eq. 14.

Solver	Dist	Solved	Time (s)	Flips	Flip-rate (Flips/ $\mu$ s)
<b>cBRIM-TMB</b>	Scale-Free	100/100	4.06e-03	2.78e+07	6853.00
	Uniform	100/100	5.41e-03	9.34e+06	1726.00
<b>cBRIM-BRW</b>	Scale-Free	94/100	1.58e-02	4.00e+06	252.58
	Uniform	99/100	6.86e-03	8.62e+06	1255.88
KISSAT	Scale-Free	100/100	2.50e-01	N/A	N/A
	Uniform	96/100	1.62e+02	N/A	N/A
WalkSAT	Scale-Free	100/100	1.03e-01	5.23e+05	5.23
	Uniform	100/100	2.59e-01	1.96e+06	7.54
ProbSAT	Scale-Free	100/100	6.01e-01	3.2e+06	5.32
	Uniform	100/100	4.30e-01	2.05e+06	4.77

Table 1 summarizes the results of each solver’s performance separately on scale-free and uniform random problems. The reported times and flips are the geometric mean of the respective metric for all successful runs scaled by the MTS formula in Eq. 14. As an example, let us take the case of scale-free problems. **cBRIM-TMB** requires about  $53\times$  and  $8.7\times$  more flips than WalkSAT and ProbSAT respectively. However, it flips over  $550\times$  faster than both the SLS solvers in the geometric average case (see “Flip-rate” in table) which results in **cBRIM-TMB** having a net speedup of  $25\times$  and  $147\times$  over WalkSAT and ProbSAT respectively. Fast hardware dynamics enable a high “Flip-rate” for **cBRIM-TMB** which takes about 0.5 ns to flip a variable, compared to numerous instructions per flip for software solvers [46].

KISSAT outperforms ProbSAT by  $2.4\times$  in run time for scale-free problems, but falls behind both **cBRIM-TMB** and WalkSAT. As expected, uniform random problems proved particularly difficult for KISSAT. Such problems lack structure that a CDCL solver like KISSAT can exploit [18, 37].

## 6 Conclusions and future work

Non-von Neumann computing using dynamical systems show potential for extreme speedup and energy efficiencies for difficult optimization problems. Most of these solvers operate on quadratic models like Ising/QUBO. Recently, cubic dynamical solvers (CDS) have also been proposed for problems like SAT. In this paper, we show that such CDS systems that rely on problem-agnostic heuristics do not seem to offer any tangible benefit over software SAT solvers. This motivates us to utilize SAT-specific heuristics. The combination of a *make-only* heuristic with CDS does perform well for uniform random problems but underperforms for

industrial-like, scale-free instances. Our analysis lead us to the opinion that  $\mathcal{M}_n$  alone might be insufficient for such skewed variable distribution in problems and we need to take into account  $\mathcal{B}_n$  as well. In the process, we derive a novel relationship between the time derivative of each variable in CDS to its  $(\mathcal{M}_n - \mathcal{B}_n)$  quantity. Given that  $\mathcal{M}_n$  can be computed easily, one can recover  $\mathcal{B}_n$  using this relationship. This allows us to propose a new heuristic using a nonlinear tanh-based function of  $\mathcal{M}_n$  and  $\mathcal{B}_n$  which we combine with CDS. With simulation results, our proposed solver, called cBRIM-TMB, is projected to perform orders of magnitude faster than software SAT solvers both in scale-free and uniform random problems. The speedup can be attributed to the extremely fast dynamics of CDS-based systems that give rise to a high flip-rate.

We believe that cBRIM-TMB is just a first attempt at combining a *make/break* heuristic with a CDS system. Further design space exploration is likely to come up with more efficient solution systems. Testing on real SAT instances is also needed to judge the true effectiveness of our solver especially because a reduction from generic  $k$ -SAT to 3-SAT is required. Moreover, since real SAT instances are huge (containing hundreds of thousands of variables), efficient software-hardware co-design approach is necessary owing to the limited capacity of hardware. We hope that our proposed solver can make CDS a promising paradigm to solve SAT and encourage more people to contribute to it.

---

## References

- 1 Richard Afoakwa, Yiqiao Zhang, Uday Kumar Reddy Vengalam, Zeljko Ignjatovic, and Michael Huang. Brim: Bistable resistively-coupled Ising machine. *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 749–760, 2021. doi:10.1109/HPCA51647.2021.00068.
- 2 Tameem Albash and Daniel A. Lidar. Adiabatic quantum computation. *Rev. Mod. Phys.*, 90:015002, January 2018. doi:10.1103/RevModPhys.90.015002.
- 3 Tasniem Nasser Alyahya, Mohamed El Bachir Menai, and Hassan Mathkour. On the structure of the boolean satisfiability problem: A survey. *ACM Computing Surveys (CSUR)*, 55(3):1–34, 2022.
- 4 Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. On the structure of industrial SAT instances. In *Principles and Practice of Constraint Programming-CP 2009: 15th International Conference, CP 2009 Lisbon, Portugal, September 20-24, 2009 Proceedings 15*, pages 127–141. Springer, 2009.
- 5 Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. Towards industrial-like random SAT instances. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI'09*, pages 387–392, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.
- 6 Masashi Aono, Makoto Naruse, Song-Ju Kim, Masamitsu Wakabayashi, Hirokazu Hori, Motoichi Ohtsu, and Masahiko Hara. Amoeba-inspired nanoarchitectonic computing: Solving intractable computational problems using nanoscale photoexcitation transfer dynamics. *Langmuir : the ACS journal of surfaces and colloids*, 29, April 2013. doi:10.1021/la400301p.
- 7 Daniel Azses, Maxime Dupont, Bram Evert, Matthew J. Reagor, and Emanuele G. Dalla Torre. Navigating the noise-depth tradeoff in adiabatic quantum circuits, 2022. doi:10.48550/arXiv.2209.11245.
- 8 Adrian Balint and Uwe Schöning. Choosing probability distributions for stochastic local search and the role of make versus break. In *SAT*, 2012.
- 9 Mohammad Khairul Bashar, Antik Mallick, Avik W. Ghosh, and Nikhil Shukla. Dynamical system-based computational models for solving combinatorial optimization on hypergraphs. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, pages 1–1, 2023. doi:10.1109/JXCDC.2023.3235113.



- 10 Tomas Baylo, Marijn J.H. Heule, Markus Iser, Matti Järvisalo, and Martin Suda. Proceedings of SAT competition 2022: Solver and benchmark descriptions. In *Proceedings of SAT Competition 2022 : Solver and Benchmark Descriptions*, volume B-2022-1 of *Department of Computer Science Series of Publications B*, Helsinki, 2022. Department of Computer Science, University of Helsinki. URL: <http://hdl.handle.net/10138/347211>.
- 11 Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermann Heimonen, Jakob S. Kottmann, Tim Menke, Wai-Keong Mok, Sukin Sim, Leong-Chuan Kwek, and Alán Aspuru-Guzik. Noisy intermediate-scale quantum algorithms. *Rev. Mod. Phys.*, 94:015004, February 2022. doi:10.1103/RevModPhys.94.015004.
- 12 Zhengbing Bian, Fabian Chudak, William Macready, Aidan Roy, Roberto Sebastiani, and Stefano Vartotti. Solving sat (and maxsat) with a quantum annealer: Foundations, encodings, and preliminary results. *Information and Computation*, 275:104609, 2020. doi:10.1016/j.ic.2020.104609.
- 13 A. Biere, M. Heule, H. van Maaren, and T. Walsh. *Handbook of Satisfiability: Volume 185 Frontiers in Artificial Intelligence and Applications*. IOS Press, NLD, 2009.
- 14 Armin Biere and Mathias Fleury. Gimsatul, IsaSAT, Kissat entering the SAT competition 2022. In *Proceedings of SAT Competition 2022 : Solver and Benchmark Descriptions*, 2022.
- 15 Sami Boulebnane and Ashley Montanaro. Solving boolean satisfiability problems with the quantum approximate optimization algorithm, 2022. doi:10.48550/arXiv.2208.06909.
- 16 Connor Bybee, Denis Kleyko, Dmitri E Nikonov, Amir Khosrowshahi, Bruno A Olshausen, and Friedrich T Sommer. Efficient optimization with higher-order Ising machines. *arXiv preprint*, 2022. arXiv:2212.03426.
- 17 Muya Chang, Xunzhao Yin, Zoltan Toroczkai, Xiaobo Hu, and Arijit Raychowdhury. An analog clock-free compute fabric base on continuous-time dynamical system for solving combinatorial optimization problems. In *2022 IEEE Custom Integrated Circuits Conference (CICC)*, pages 1–2, 2022. doi:10.1109/CICC53496.2022.9772850.
- 18 Mohamed Sami Cherif, Djamel Habet, and Cyril Terrioux. Combining vsids and chb using restarts in SAT. In *27th International Conference on Principles and Practice of Constraint Programming (CP 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.
- 19 Tzuu-Shuh Chiang, Chii-Ruey Hwang, and Shuenn Jyi Sheu. Diffusion for global optimization in  $\mathbb{R}^n$ . *SIAM Journal on Control and Optimization*, 25(3):737–753, 1987. doi:10.1137/0325042.
- 20 D-WAVE. minorminer, 2014. URL: <https://github.com/dwavesystems/minorminer>.
- 21 D-WAVE. Operation and timing, 2022. URL: [https://docs.dwavesys.com/docs/latest/c\\_qpu\\_timing.html](https://docs.dwavesys.com/docs/latest/c_qpu_timing.html).
- 22 John D. Davis, Zhangxi Tan, Fang Yu, and Lintao Zhang. A practical reconfigurable hardware accelerator for boolean satisfiability solvers. In *Proceedings of the 45th Annual Design Automation Conference, DAC '08*, pages 780–785, New York, NY, USA, 2008. Association for Computing Machinery. doi:10.1145/1391469.1391669.
- 23 Mária Ercsey-Ravasz and Zoltán Toroczkai. Optimization hardness as transient chaos in an analog approach to constraint satisfaction. *Nature Physics*, 7(12):966–970, 2011.
- 24 Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014. doi:10.48550/arXiv.1411.4028.
- 25 D. Freedman and P. Drineas. Energy minimization via graph cuts: settling what is possible. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 939–946 vol. 2, 2005. doi:10.1109/CVPR.2005.143.
- 26 Thomas Gabor, Sebastian Zielinski, Sebastian Feld, Christoph Roch, Christian Seidel, Florian Neukart, Isabella Galter, Wolfgang Mauerer, and Claudia Linnhoff-Popien. Assessing solution quality of 3SAT on a quantum annealing platform, 2019. doi:10.48550/arXiv.1902.04703.
- 27 Hayato Goto, Kosuke Tatsumura, and Alexander R Dixon. Combinatorial optimization by simulating adiabatic bifurcations in nonlinear hamiltonian systems. *Science advances*, 5(4):eaav2372, 2019.

- 28 Kanupriya Gulati, Suganth Paul, Sunil P. Khatri, Srinivas Patil, and Abhijit Jas. Fpga-based hardware acceleration for boolean satisfiability. *ACM Trans. Des. Autom. Electron. Syst.*, 14(2), April 2009. doi:10.1145/1497561.1497576.
- 29 Ryan Hamerly, Takahiro Inagaki, Peter L. McMahon, Davide Venturelli, Alireza Marandi, Tatsuhiro Onodera, Edwin Ng, Carsten Langrock, Kensuke Inaba, Toshimori Honjo, Koji Enbutsu, Takeshi Umeki, Ryoichi Kasahara, Shoko Utsunomiya, Satoshi Kako, Ken ichi Kawarabayashi, Robert L. Byer, Martin M. Fejer, Hideo Mabuchi, Dirk Englund, Eleanor Rieffel, Hiroki Takesue, and Yoshihisa Yamamoto. Experimental investigation of performance differences between coherent Ising machines and a quantum annealer. *Science Advances*, 5(5):eaau0823, 2019. doi:10.1126/sciadv.aau0823.
- 30 Kazuaki Hara, Naoki Takeuchi, Masashi Aono, and Yuko Hara-Azumi. Amoeba-inspired stochastic hardware sat solver. In *20th International Symposium on Quality Electronic Design (ISQED)*, pages 151–156, 2019. doi:10.1109/ISQED.2019.8697729.
- 31 Takahiro Inagaki, Yoshitaka Haribara, Koji Igarashi, Tomohiro Sonobe, Shuhei Tamate, Toshimori Honjo, Alireza Marandi, Peter L McMahon, Takeshi Umeki, Koji Enbutsu, et al. A coherent Ising machine for 2000-node optimization problems. *Science*, 354(6312):603–606, 2016.
- 32 Tadashi Kadowaki and Hidetoshi Nishimori. Quantum annealing in the transverse Ising model. *Physical Review E*, 58(5):5355–5363, November 1998. doi:10.1103/physreve.58.5355.
- 33 Taro Kanao and Hayato Goto. Simulated bifurcation for higher-order cost functions. *Applied Physics Express*, 16(1):014501, 2022.
- 34 Henry Kautz. Walksat, 2018. URL: <https://gitlab.com/HenryKautz/Walksat>.
- 35 Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.
- 36 V. Kolmogorov and R. Zabini. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004. doi:10.1109/TPAMI.2004.1262177.
- 37 Jia Hui Liang, Vijay Ganesh, Ed Zulkoski, Atulan Zaman, and Krzysztof Czarnecki. Understanding vsids branching heuristics in conflict-driven clause-learning SAT solvers. In *Hardware and Software: Verification and Testing: 11th International Haifa Verification Conference, HVC 2015, Haifa, Israel, November 17-19, 2015, Proceedings 11*, pages 225–241. Springer, 2015.
- 38 Andrew Lucas. Ising formulations of many NP problems. *Frontiers in Physics*, 2:5, 2014.
- 39 Debasis Mitra, Fabio Romeo, and Alberto Sangiovanni-Vincentelli. Convergence and finite-time behavior of simulated annealing. In *1985 24th IEEE Conference on Decision and Control*, pages 761–767, 1985. doi:10.1109/CDC.1985.268600.
- 40 Ferenc Molnár, Shubha R Kharel, Xiaobo Sharon Hu, and Zoltán Toroczkai. Accelerating a continuous-time analog SAT solver using gpus. *Computer Physics Communications*, 256:107469, 2020.
- 41 Anh Hoang Ngoc Nguyen, Masashi Aono, and Yuko Hara-Azumi. Amoeba-inspired hardware sat solver with effective feedback control. In *2019 International Conference on Field-Programmable Technology (ICFPT)*, pages 243–246, 2019. doi:10.1109/ICFPT47387.2019.00038.
- 42 Tiemo Pedergnana and Nicolas Noiray. Exact potentials in multivariate langevin equations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(12):123146, 2022. doi:10.1063/5.0124031.
- 43 Irina Rish and Rina Dechter. Resolution versus search: Two strategies for SAT. *Journal of Automated Reasoning*, 24(1-2):225–275, 2000.
- 44 I. G. Rosenberg. Reduction of bivalent maximization to the quadratic case. *Cahiers du Centre d’Etudes de Recherche Operationnelle*, 17:71–74, 1975.
- 45 Bart Selman, Henry A Kautz, Bram Cohen, et al. Local search strategies for satisfiability testing. *Cliques, coloring, and satisfiability*, 26:521–532, 1993.

- 46 Anshujit Sharma, Richard Afoakwa, Zeljko Ignjatovic, and Michael Huang. Increasing Ising machine capacity with multi-chip architectures. In *Proceedings of the 49th Annual International Symposium on Computer Architecture, ISCA '22*, pages 508–521, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3470496.3527414.
- 47 Anshujit Sharma, Matthew Burns, Andrew Hahn, and Michael Huang. Augmented electronic ising machine as an effective sat solver, 2023. arXiv:2305.01623.
- 48 Yong Shim, Abhronil Sengupta, and Kaushik Roy. Biased random walk using stochastic switching of nanomagnets: Application to SAT solver. *IEEE Transactions on Electron Devices*, 65(4):1617–1624, 2018. doi:10.1109/TED.2018.2808232.
- 49 Ali Asgar Sohangpurwala and Peter Athanas. An effective probability distribution SAT solver on reconfigurable hardware. In *2016 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, pages 1–6. IEEE, 2016.
- 50 Ali Asgar Sohangpurwala, Mohamed W. Hassan, and Peter M. Athanas. Hardware accelerated SAT solvers - a survey. *J. Parallel Distributed Comput.*, 106:170–184, 2017.
- 51 Juexiao Su, Tianheng Tu, and Lei He. A quantum annealing approach for boolean satisfiability problem. In *Proceedings of the 53rd Annual Design Automation Conference*, pages 1–6, 2016.
- 52 Tianshi Wang, Leon Wu, and Jaijeet Roychowdhury. New computational results and hardware prototypes for oscillator-based Ising machines. In *Proceedings of the 56th Annual Design Automation Conference 2019, DAC '19*, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3316781.3322473.
- 53 Yulun Wang and Predrag S. Krstic. Prospect of using grover's search in the noisy-intermediate-scale quantum-computer era. *Physical Review A*, 102(4), October 2020. doi:10.1103/physreva.102.042609.
- 54 Ryan Williams. *Maximum Two-Satisfiability*, pages 507–510. Springer US, Boston, MA, 2008. doi:10.1007/978-0-387-30162-4\_227.
- 55 Pan Xu, Jinghui Chen, Difan Zou, and Quanquan Gu. Global convergence of langevin dynamics based algorithms for nonconvex optimization. *Advances in Neural Information Processing Systems*, 31, 2018.
- 56 Xunzhao Yin, Behnam Sedighi, Melinda Varga, Maria Ercsey-Ravasz, Zoltan Toroczkai, and Xiaobo Sharon Hu. Efficient analog circuits for boolean satisfiability. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 26(1):155–167, 2017.
- 57 Yiqiao Zhang, Uday Kumar Reddy Vengalam, Anshujit Sharma, Michael Huang, and Zeljko Ignjatovic. Qubrim: A cmos compatible resistively-coupled Ising machine with quantized nodal interactions. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design, ICCAD '22*, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3508352.3549443.

## A Cubic BRIM (cBRIM) Simulation

For simulation purposes, we consider the product voltage  $x_{jk}$  as an auxiliary variable and incorporate all the coefficients  $l_n$ ,  $q_{nj}$  and  $c_{njk}$  into one matrix  $J$ . We simulate cBRIM using the following pseudocode:

```
// f: the 3SAT formula
// simtime: the total anneal time,
// tstep: the simulation step size
procedure cBRIM (f, simtime, tstep) {
  // Initialize the coupling matrix and variable
  // based on the problem
  J_matrix, x_vector = initialize_problem(f);

  // Update the auxiliary variable, x_jk = x_j * x_k
  update_auxiliary(x_vector);
```

```

// Initialize time to 0
time = 0;

// Main simulation loop
while(time < simtime) {
    // Get the quantized variables in {0,1}
    qx_vector = Quant(x_vector);

    // Compute the derivatives of variables
    // by matrix vector multiplication
    dxdt = (J_matrix * qx_vector)/(R * C);

    // Select heuristic spin flips
    hflips = heuristic(f, time, qx_vector);

    // Assign opposing currents to perform heuristic flips
    // hfR is heuristic flip resistance (1K0hm)
    for (var_id in hflips) {
        dxdt(var_id) =
            (!hflips(var_id) - x_vector(var_id)) / (hfR * C);
    }

    // Update the voltages of variables
    x_vector += dxdt * timestep;

    // Limit voltages in [0, 1]
    limit(x_vector);

    // Update the auxiliary variables
    update_auxiliary(x_vector);

    // Increment time
    time += timestep;
}

// return quantized state, 1 for true and 0 for false
return Quant(x_vector);
}

```

## B Langevin cBRIM

Here, we give a theoretical analysis of using a simple Gaussian white noise perturbation  $\eta \sim \mathcal{N}(0, \sigma^2)$  in cBRIM. We can rewrite Eq. 8 as follows:

$$\frac{dx_n}{dt} = \frac{1}{RC} \times (l_n + \sum_j q_{nj}x_j + \sum_{j < k} c_{njk}x_jx_k) + \eta \quad (15)$$

Following a similar procedure as in Eq. 9 to Eq. 11, we arrive at the following expression:

$$\frac{dx_n}{dt} = -\alpha \frac{\partial H_C}{\partial x_n} + \eta \quad (16)$$

where  $\alpha = \frac{1}{RC}$ . Such a system is governed by Langevin dynamics and the joint probability density  $P(x, t)$  for some state  $x$  evolves according to the Fokker-Planck equation [42]:

$$\frac{\partial P(x, t)}{\partial t} = \nabla \cdot \left[ \alpha P(x, t) \nabla H_C + \frac{\sigma^2}{2} \nabla P(x, t) \right] \quad (17)$$

We can derive the stationary distribution when  $t \rightarrow \infty$  by setting  $\frac{\partial P(x, t)}{\partial t} = 0$  in Eq. 17. This results in the Boltzmann distribution as intended [19, 55]:

$$P_\infty(x) = \frac{1}{Z} \exp \left[ \frac{-2\alpha H_C(x)}{\sigma^2} \right] \quad (18)$$

where  $Z$  is the normalization constant. Eq. 18 implies that when the system settles into equilibrium, the ground state(s) have higher probability than other states.