Journal Pre-proof

Popularity-Similarity random SAT formulas

Jesús Giráldez-Cru and Jordi Levy

PII: S0004-3702(21)00088-6

DOI: <https://doi.org/10.1016/j.artint.2021.103537>

Reference: ARTINT 103537

To appear in: *Artificial Intelligence*

Received date: 24 January 2020

Revised date: 2 December 2020

Accepted date: 21 May 2021

Please cite this article as: J. Giráldez-Cru and J. Levy, Popularity-Similarity random SAT formulas, *Artificial Intelligence*, 103537, doi: <https://doi.org/10.1016/j.artint.2021.103537>.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2021 Published by Elsevier.

Popularity-Similarity Random SAT Formulas

Jesús Giráldez-Cru

Andalusian Research Institute DaSCI "Data Science and Computational Intelligence", University of Granada (UGR), Spain

Jordi Levy

Artificial Intelligence Research Institute (IIIA-CSIC), Campus UAB, Bellaterra, Spain

Abstract

In the last decades, we have witnessed a remarkable success of algorithms solving the Boolean Satisfiability problem (SAT) on instances encoding application or real-world problems arising from a very diverse number of domains, such as hardware and software verification, planning or cryptography. These algorithms are the so known Conflict-Driven Clause Learning (CDCL) SAT solvers. Interestingly enough, the reasons for the success of these solvers on this diverse range of problems are not completely understood yet.

A common issue when facing this open challenge is the heterogeneity of this set of benchmarks. Another problem is the limited number of existing instances. In this context, random models of SAT formulas capturing features shared by the majority of these application benchmarks become crucial, for both theoretical and practical purposes. On the one hand, it is undoubtedly necessary to have random models where theoretical properties, like hardness, can be studied. Therefore, *realistic* random SAT models may contribute to explain the success of these solvers on these industrial problems. On the other hand, the limited number of benchmarks and their hardness in practice makes the evaluation of new solving techniques a costly task. Therefore, these *realistic* random SAT generators can provide an unlimited number of *pseudo-industrial* random SAT instances with some desired properties.

In this work, we present a random SAT instances generator based on the notion of *locality*. This notion is complementary to the *popularity* of variables, which is present in the scale-free structure, observable in actual application problems and achievable by previous generators. Our random SAT model combines both locality and popularity, and we show that they are two decisive dimensions of attractiveness among the variables of a formula, and how CDCL SAT solvers take advantage of them. Locality is closely related to the community structure, another important feature of application SAT benchmarks, which is indirectly achieved by this model. To the best of our knowledge, this is the first random SAT model that generates both scale-free structure and community structure at once.

Keywords: Satisfiability, SAT generator, Random SAT, Scale-free structure, Community structure

NOTE: In this second submission, in order to facilitate the reviewing process, the major changes with respect to the previous version are marked in red (this is also clarified in the cover letter).

1. Introduction

It is well known that most real-world SAT instances used in SAT competitions have a great variability in the number of occurrences of variables. Ansótegui *et al.* [2] showed that, in most of these application SAT benchmarks, the number of occurrences k of a randomly selected variable follows a power-law distribution $P(k) \sim k^{-\delta}$ where δ is between 2 and 3. In many cases, the clauses size also shows this kind of distribution. This means that a solver that would assign preferably those very frequent or *popular* variables would decrease the size of the formula very quickly. Variants of this strategy have been used in the past [37], in the Bohm's heuristic, Maximum Occurrences on clauses of

Email addresses: jgiralde@ugr.es (Jesús Giráldez-Cru), levy@iiia.csic.es (Jordi Levy)

URL: <http://www.ugr.es/~jgiralde> (Jesús Giráldez-Cru), <http://www.iiia.csic.es/~levy> (Jordi Levy)

Minimum Size (MOMS) [18] and Jeroslow-Wang heuristic [30], for instance. However, nowadays most modern SAT solvers use the *Variable State Independent Decaying Sum* (VSIDS) heuristic [38] and the *Learning Rate Branching* (LRB) heuristic [36]. The intuition to explain the success of these heuristics is that they make the solver *focus* the search on some *local area* of the formula. In large formulas, sometimes, it is preferable to assign variables *closer* to other recently assigned variables, than to assign popular but *distant* variables [31]. In this direction, Giráldez-Cru and Levy [24, 25] define a model of random SAT instances with high modularity [3, 5, 6], and show that VSIDS tends to focus on *communities* of variables closely connected by clauses. Ansótegui *et al.* [7, 8] define another model of random formulas with *scale-free structure*, i.e. where the number of occurrences of variables follows a power-law distribution, as observed in real-world SAT instances. However, there is no model of random SAT formulas that captures both properties at once: a power-law distribution in the number of variable occurrences and a notion of *locality*. This is the main objective of this work.

We can represent SAT instances as graphs; either representing variables as nodes, and the coexistence of two variables in a clause as an edge; or variables and clauses both as nodes, and the occurrence of a variable in a clause as an edge between them. This approach has allowed to apply many ideas from complex networks to SAT, e.g., the study of the fractal dimension of these formulas [4].

Most real-world networks (or graphs), such as Internet, the Web, or many social and metabolic networks, have the same *scale-free* structure observed in industrial SAT instances. This means that the degree (number of connections) of a random node follows a power-law distribution, where a few popular nodes attract a significant fraction of the edges. *Preferential attachment* [10] is a model where the probability of a node to get new connections is proportional to its *popularity* (defined as the number of connections it already has). It has been proposed as the force that governs the growth of scale-free networks. However, it has also been observed that most real-world networks have also high clustering factor¹ and modularity, whereas graphs generated with the preferential attachment model have a low clustering factor and a modularity that tends to zero as the number of nodes tends to infinity.

Papadopoulos *et al.* [41] show that, although *popularity* and preferential attachment is one of the forces ruling real-world networks, another force is *similarity*. In other words, new nodes tend to connect to *popular* nodes (that already have a lot of connections), and also to *similar* nodes (that are close with respect to some metric). They define a model where at every time instant $t \in \{1, \dots, n\}$ a new node with index $i = t$ is created. The probability of an older node $j < i$ to get a connection from the new node i depends on some *energy* or *distance* $e_{ij} = r_i + r_j + \log(\theta_{ij}/2)$, where $r_i = \log(i)$ is the popularity of node i (the smaller the value of r_i , the greater the popularity of node i), and θ_{ij} is the similarity between i and j . To model similarity, they assign a random angle θ_i to each node i , and compute θ_{ij} as the minimum distance between angles θ_i and θ_j , i.e., $\theta_{ij} = \pi - |\pi - |\theta_i - \theta_j||$. Preferential attachment generates scale-free graphs with node degrees following a power-law distribution $P(k) \sim k^{-\delta}$, where $\delta = 2$. In order to obtain greater values of δ , they assume a *popularity fading*, where popularity of node i depends on the time t as $r_i(t) = \beta \log(i) + (1 - \beta) \log(t)$, and $\beta \in [0, 1]$ is smaller as fading is faster. Finally, as edges in a graph are exclusive (we cannot have more than one edge between two nodes), they assume that edges behave as *fermions*.² Therefore, they use the Fermi-Dirac probability distribution for the expected number of edges n in a given energy level e_{ij} :

$$E[n_{ij}] = \frac{1}{1 + e^{\frac{e_{ij} - \mu}{kT}}}$$

where μ is the total chemical potential, k is the Boltzmann's constant, and T is the temperature.³ Later, Krioukov *et al.* [35] and Boguñá *et al.* [13] describe what they call an *Einsteinian* version of this model, where nodes are generated at once in a hyperbolic circle and the energy between two nodes is described by the hyperbolic distance between them.

In [26], a preliminary version of this work, we propose a new model of random SAT instances that captures the notion of *locality* whereas it preserves the power-law distribution in the number of variable occurrences. Additionally, this model also allows the lengths of the clauses to follow another power-law distribution. We adapt many ideas of Papadopoulos *et al.* [41], but one of the problems is that the *growing* process imposes some limitations on the graph. For instance, a minimum degree on nodes. Thus, we also adapt some ideas of Ansótegui *et al.* [7], where

¹The clustering factor is the fraction of neighbors of a node that are connected, i.e., the conditional probability $P(a \leftrightarrow b \mid \exists c. a \leftrightarrow c \wedge b \leftrightarrow c)$.

²Fermions are particles that obey the Pauli exclusion principle. Therefore, we can not have two fermions in exactly the same physical state.

³In fact, they integrate the Boltzmann's constant inside the definition of temperature.

instead of a growing process as in preferential attachment, all nodes are created at once, and a distinct probability is used for every possible edge, ensuring that the expected node degree is $E[k_i] \sim i^{-\beta}$. This produces a power-law distribution of degrees on random nodes $P(k) \sim k^{-\delta}$, where $\delta = 1 + 1/\beta$. Moreover, we generate bipartite graphs with nodes representing either variables or clauses, where the degree of a variable-node is the number of occurrences of such a variable, and the degree of a clause-node is the size of that clause. Although in most real-world industrial instances both values follow power-law distributions, their exponent δ is different in each case. In the preliminary model presented in [26], the similarity between two nodes is given by the angle difference θ_{ij} , where θ_i and θ_j are random angles with uniform probability in $[0, 2\pi]$. However, the popularity of node i computed as $\beta \log(i)$ is deterministic and only depends on its index (or age) i and a parameter β . In this work, we select a random radius r_i for every node i , with uniform probability in $[0, 1]$. This differs from the Einsteinian model of [35, 13], where the radius is selected randomly in the range $[0, R]$ with probability $P(r) \sim r^{\alpha r}$. In Section 4.4, we comment on these differences.

Also, we show that CDCL SAT solvers take advantage of both popularity and similarity. In particular, we show that when formulas have these two properties, the most typical heuristic used in these solvers (i.e., VSIDS) quickly focuses its decisions on both popular variables, and on variables similar to them. We also show that CDCL solvers perform better in formulas with popularity and similarity, than other solvers *specialized* in classical random formulas. On the contrary, the absence of these two properties makes these last solvers more efficient than CDCL techniques. This suggests that both popularity and similarity are two crucial properties to understand the success of CDCL on real-world SAT benchmarks.

Finally, we empirically analyze the SAT-UNSAT phase transition phenomenon of this model. Our experimental results suggest that our model behaves very similarly to the classical random SAT model in the absence of locality (i.e., when there is neither popularity nor similarity). On the contrary, the presence of these structural components modifies the phase transition, which seems to tend to a fixed clause/variable ratio $m/n = 0$. It remains unclear whether this phase transition also depends on the exponent δ of the scale-free structure of the formula.

In [26] we presented a preliminary version of this work. In this new version, the definition of the model has been revised and slightly modified as explained above. Also, in this new version we introduce the discussion of the PS model for graphs, extended in more details in Section 4.4. Additionally, we have included a new evaluation of different CDCL SAT solvers on PS random instances in Section 6.3. Finally, the experimental analysis of the SAT-UNSAT phase transition phenomenon of the PS model presented in Section 7 is completely new. The implementation of the Popularity-Similarity random model for SAT instances is publicly available at <https://github.com/jgirald/ps-sat>.

The rest of the paper is organized as follows. In Section 2 we describe some related works on random SAT instances generation. Preliminary concepts used in the manuscript are defined in Section 3. The Popularity-Similarity model for random SAT instances is described in Section 4. Section 5 describes an experiment of a simple random model with distinct clause lengths, whose results justifies our choices in our experimental evaluation. An analysis of the relation between locality and CDCL SAT solving techniques is presented in 6, whereas the empirical study of the SAT-UNSAT phase transition phenomenon of the PS model is described in Section 7. Finally, conclusions are presented in Section 8.

2. Related Work

The problem of generating random SAT formulas which realistically capture the computational properties of industrial instances has been stated as one of the most challenging problems in propositional search [42, 32, 33, 16].

Ansótegui *et al.* [7] propose the *Random Scale-free model*, where the number of variable occurrences follow a power-law distribution $P(k) \sim k^{-\delta}$. However, this model is unable to generate formulas with high clustering or community structure. Recently, Friedrich *et al.* [22] and Ansótegui *et al.* [8] have proved that the phase transition point of 2-CNF random scale-free formulas depends on the exponent δ of the power-law distribution as $m/n = \frac{(\delta-1)(\delta-3)}{(\delta-2)^2}$. Experimental results suggest that this is also the case in k -CNF formulas with $k > 2$ [22]. Friedrich *et al.* [21], Friedrich and Rothenberger [19, 20], Cooper *et al.* [15] and Omelchenko and Bulatov [40] study the properties of the phase transition of 2-CNF scale-free and similar models of random formulas.

Giráldez-Cru and Levy [24, 25] propose the *Community Attachment model* for random SAT formulas, which generates formulas with clear community structure. In this model, there is no high variability in the number of

occurrences of variables. Although these formulas are experimentally *easy*, Mull *et al.* [39] prove that in theory they are NP-hard.

There are also other works focused on random generators for particular problem domains. Gomes and Selman [27] propose a model that introduces random perturbations into instances encoding the partial Latin square completion problem. Achlioptas *et al.* [1] refine the previous model to guarantee the existence of satisfying assignments in the generated instances. Gent *et al.* [23] presented *morphing*, a method that generates problems with small-world topology. Järvisalo *et al.* [29] propose an instance generator based on Boolean functions and their optimal circuits.

A detailed comparison between our PS model for SAT instances and the original PS model for graphs [41, 35, 13] is described in Section 4.4.

3. Preliminaries

The well-known *Boolean Satisfiability problem* (SAT) is the problem of deciding whether there exists a satisfying assignment for a given propositional formula, i.e., an assignment of its variables in such a way that the formula is evaluated as `true`. Without loss of generality, we consider propositional formulas in Conjunctive Normal Form (CNF).⁴ A *literal* is either a variable or its negation, a *clause* is a disjunction of literals, and a formula in CNF is a conjunction of clauses. A *k*-CNF is a CNF in which all clauses have exactly *k* literals.

A graph $G(V, E)$ is composed by a set of nodes V , and a set of edges $E \subseteq V \times V$. In a bipartite graph $G(V_1, V_2, E)$, nodes belong either to V_1 or to V_2 , and edges are restricted to connect nodes of these two partitions: $E \subseteq V_1 \times V_2$.

4. The Popularity-Similarity SAT Model

In this section we define the Popularity-Similarity random model for SAT instances (PS for short), adapting some ideas from Papadopoulos *et al.* [41] and Ansótegui *et al.* [7].

Definition 1 (Popularity-Similarity Random SAT Instance). *In order to generate a random SAT instance with locality over n variables with m clauses of average size k , we first assign a random radius $r_i \in [0, 1]$ and a random angle $\theta_i \in [0, 2\pi]$ to every variable $i \in \{1, \dots, n\}$, and a random radius $r'_j \in [0, 1]$ and a random angle $\theta'_j \in [0, 2\pi]$ to every clause $j \in \{1, \dots, m\}$, with uniform probability distributions.*

Then, we construct a bipartite random graph with n variable-nodes and m clause-nodes, where every possible edge $i \leftrightarrow j$ between a variable-node i and a clause-node j , is selected with probability:

$$P(i \leftrightarrow j) = \frac{1}{1 + \left(\frac{r_i^\beta \cdot r'^{\beta'}_j \cdot \theta_{ij}}{R} \right)^{1/T}} \quad (1)$$

where $\beta, \beta' \in (0, 1)$ and T are parameters of the model, θ_{ij} is the minimal distance between angles θ_i and θ'_j :

$$\theta_{ij} = \pi - |\pi - |\theta_i - \theta'_j||$$

and R is the normalizing constant ensuring that, on average, the number of selected edges is $k \cdot m$. In other words, we consider β, β', k, T, n and m as parameters of the model, and define R as the value (depending on those parameters) that satisfies:

$$\sum_{i=1}^n \sum_{j=1}^m P(i \leftrightarrow j) = k \cdot m \quad (2)$$

Finally, we construct a random SAT formula from the graph as follows. For every edge $i \leftrightarrow j$ in the bipartite graph, we add to the clause C_j the literal x_i with probability $1/2$,⁵ or the literal $\neg x_i$ otherwise.

⁴Any propositional formula can be written in CNF in polynomial time.

⁵This is the common choice in random SAT models. In fact, analyzing application SAT benchmarks, we observe that most of the variables have a positive/negative ratio of occurrences close to $1/2$.

Figure 1: Representation of $P(i \leftrightarrow j)$ for a fixed j as a function of r_i and θ_{ij} , for $\beta = 0.8$ and $T = 0.3$. We observe that the probability of connection is higher when both the angle θ_{ij} and the radius r_i are close to zero.

This model corresponds to defining the *energy* of the edge $i \leftrightarrow j$ as $e_{ij} = \beta \cdot \log(r_i) + \beta' \cdot \log(r'_j) + \log(\theta_{ij})$, the chemical potential as $\mu = \log R$, and using the Fermi-Dirac probability distribution. The first term in the energy e_{ij} represents the popularity of variable i , the second the popularity of clause j and the third the *similarity* between them. The probability of selecting an edge could be any function of this energy $P(i \leftrightarrow j) = f(e_{ij})$. However, following the same argument of Papadopoulos *et al.* [41], we decide to use the Fermi-Dirac probability distribution, and a simplification of it described in Subsection 4.3. Figure 1 shows a representation of the probability defined in (1).

The effect of temperature T is regulating the entropy of the resulting formula. In the limit $T \rightarrow 0$, when we approach the absolute zero temperature, the number of fermions in state i is one when $e_i < \mu$, and zero when $e_i > \mu$. For other temperatures, the average number of fermions is between zero and one. Therefore, at $T = 0$, the total chemical potential is equal to the energy of the n -th less energetic state, where n is the number of particles. In our case, for $T = 0$, the value R is the minimum value such that

$$|\{i \leftrightarrow j \mid e_{ij} < \log(R)\}| = k \cdot m$$

Therefore, for $T = 0$, the generated bipartite graph consists of the $k \cdot m$ edges with smallest energy. At $T = \infty$, we get the graph with maximal entropy, which corresponds to the classical bipartite random $G(n, p)$, i.e., classical random SAT formulas. The chemical potential μ , or in our case R , depends on the temperature. In Subsection 4.2 we describe how R can be approximated when $T > 0$.

For $T > 0$, the model generates SAT instances with $k \cdot m$ literals *on average*. If we want to generate instances of *exactly* $k \cdot m$ literals, we have two alternatives. We can select a random variable i and clause j with uniform probability, and if the edge $i \leftrightarrow j$ has not been previously added, add it with probability given by Equation 1, repeating the process until we get the desired number of edges. Alternatively, we can select a random edge with (non-uniform) probability $P(i \leftrightarrow j)$, adding this edge if it has not been added previously, and repeating the process until we have $k \cdot m$ edges. In both cases, we can either use rejection sampling or avoid it by setting $P(i \leftrightarrow j) = 0$ for the already added edges, normalizing R after every addition. In all these cases, the same result is obtained, and the differences only affects the efficiency of each generator.

4.1. Generating Scale-free Instances

Ansótegui *et al.* [7] proved that, for scale-free random SAT instances, if the probability of a clause is given by $P(x_{i_1} \vee \dots \vee x_{i_l}) \sim \prod_{j=1}^l (i_j)^{-\beta}$, then the number of occurrences of a variable chosen at random follows a power-law distribution $P(k) \sim k^{-\delta}$, where $\delta = 1/\beta + 1$. Here we prove a similar result:

Lemma 2. *For Popularity-Similarity random SAT instances, if the probability of an edge is $P(i \leftrightarrow j) = f(r_i^\beta r_j^{\beta'} \theta_{ij})$, and the function f decreases fast enough,⁶ then the resulting SAT instance is scale-free with a number of variable*

⁶Formally, when $r_i^{-\beta} \int_0^1 r^{-\beta'} \int_{r_i^\beta r_j^{\beta'} \pi}^\infty f(x) dx dr = o(1)$, for any r_i .

occurrences distributed as $P(k) \sim k^{-\delta}$, where $\delta = 1 + 1/\beta$. Similarly, clauses size also follows a power-law distribution with exponent $\delta' = 1 + 1/\beta'$.

PROOF: The expected degree of a fixed node i can be computed integrating $P(i \leftrightarrow j)$, for all possible values of j , and multiplying by m . Since θ_i and θ'_j are uniformly distributed in $[0, 2\pi]$, the minimal distance between both angles will be uniformly distributed in $[0, \pi]$. The radius r_j is also uniformly distributed in $[0, 1]$. The integral can be computed as:

$$\begin{aligned} E[k_i | r_i, \theta_i] &= \frac{m}{\pi} \int_0^1 \int_0^\pi f(r_i^\beta r_j^{\beta'} \theta) d\theta dr \\ &= \frac{m}{\pi} \int_0^1 r_i^{-\beta} r^{-\beta'} \int_0^{r_i^\beta r_j^{\beta'} \pi} f(x) dx dr \\ &\approx \frac{m}{\pi} \int_0^1 r_i^{-\beta} r^{-\beta'} \int_0^\infty f(x) dx dr \\ &= \frac{m r_i^{-\beta}}{\pi(1-\beta')} \int_0^\infty f(x) dx \end{aligned}$$

The approximation is correct when $r_i^{-\beta} \int_0^1 r^{-\beta'} \int_{r_i^\beta r_j^{\beta'} \pi}^\infty f(x) dx dr$ is negligible.⁷

Now, we basically have to reproduce the proof in [7, 8], to prove that $E[k_i] \sim r_i^{-\beta}$ implies $P(k) \sim k^{-\delta}$ with $\delta = 1 + 1/\beta$. Similarly, we prove that the expected size of clause j is $E[k_j] \sim r_j'^{-\beta'}$. From this, we deduce that clause size also follows a power-law distribution.

In our case the approximation above is correct for small values of T . For instance, for $T = 0$, the function f defined by the Fermi-Dirac distribution returns one for the smallest km values of energy, and zero elsewhere. Therefore, $\int_{r_i^\beta r_j^{\beta'} \pi}^\infty f(x) dx$ is exactly zero, because there are less edges than possible levels of energy. For small values of T , there is some error. ■

It would be interesting to investigate, for the particular form of the function f that we propose in our models, what values of the parameters allow us to ensure that Lemma 2 is applicable and the resulting graph is scale-free.

4.2. Normalizing the Probability Distribution

In order to normalize the probability distribution to ensure that the sum of probabilities is equal to the number of edges, as stated in Equation (2), we have to integrate it, which corresponds to sampling the coordinates of the nodes. As the distance θ_{ij} between angles is uniformly distributed, we can compute the probability of an edge between nodes i and j , conditioned to their radius r_i and r'_j , as:

$$\begin{aligned} P(i \leftrightarrow j | r_i, r'_j) &= \frac{1}{\pi} \int_0^\pi \frac{1}{1 + \left(\frac{r_i^\beta r_j'^{\beta'} \theta}{R} \right)^{1/T}} d\theta \\ &= \frac{R}{\pi} r_i^{-\beta} r_j'^{-\beta'} \int_0^{\frac{r_i^\beta r_j'^{\beta'} \pi}{R}} \frac{1}{1 + x^{1/T}} dx \\ &\approx \frac{R}{\pi} r_i^{-\beta} r_j'^{-\beta'} \int_0^\infty \frac{1}{1 + x^{1/T}} dx = \frac{R}{\pi} \frac{T\pi}{\sin(T\pi)} r_i^{-\beta} r_j'^{-\beta'} \end{aligned}$$

where we are using the same approximation as in Lemma 2, and the equality $\int_0^\infty \frac{1}{1+x^n} = \frac{\pi/n}{\sin(\pi/n)}$ that holds for any $n > 1$. Integrating w.r.t. r_i and r'_j and imposing the condition of Equation 2, we get:

$$R \approx \frac{k(1-\beta)(1-\beta') \sin(T\pi)}{nT}$$

⁷Formally, when we write $f(n) \approx g(n)$, we mean $f(n) = g(n)(1 + o(1))$, in little-o notation. When $g(n) = \Theta(1)$, like in this case, this is equivalent to $f(n) = g(n) + o(1)$. When we say that the integral is *negligible*, we mean that it is equal to $o(1)$.

which is a good approximation of R for small values of T .

Unfortunately, we have to compute the value of R for values $T > 1$, and we do not know how to do this analytically. Therefore, we use in our algorithm the Newton-Raphson method. Let $F(R) = \sum_{i=1}^n \sum_{j=1}^m P(i \leftrightarrow j)$, where we make explicit that the probability depends on the value of R . We want to find the value of R such that $F(R) = k m$. Therefore, we compute:

$$\frac{\partial F}{\partial R} = \sum_{i=1}^n \sum_{j=1}^m \frac{\left(\frac{r_i^\beta r_j^{\beta'} \theta_{ij}}{R} \right)^{1/T}}{R T \left(1 + \left(\frac{r_i^\beta r_j^{\beta'} \theta_{ij}}{R} \right)^{1/T} \right)^2}$$

and compute R as the limit of the series:

$$R_{l+1} = R_l + \frac{k m - F}{\frac{\partial F}{\partial R}} \Big|_{R=R_l}$$

Notice that, in the analytical calculation of R , we sample the values of r_i and r_j' . However, in the Newton-Raphson method we use the particular values of the r 's. Experimentally, we observe that the approximation done in the analytical calculation of R makes the formula only usable when T is very close to zero. Experimentally, we also observe that, using the Newton-Raphson method, after 4 or 5 iterations, the value of F is $k m \pm 0.1$. Therefore, we have done all the experiments using this second method.

4.3. Simplifying the Model

In the previous subsection we have seen that computing the value of R is a difficult task. Given an unnormalized probability distribution $P(x) \sim f(x)$ where $x = 1, \dots, n$, a trivial way to normalize it is defining $P(x) = f(x) / \sum_{x=1}^n f(x)$, which ensures that $\sum_{x=1}^n P(x) = 1$. However, in the case of (1), the function is already normalized (its value is always in the range $[0, 1]$), and (2) is used to compute R , not as a proper probability normalization. However, if we slightly modify (1), we can transform the computation of R in a proper probability normalization.

The expected number of bosons⁸ in an energy state e_i is given by the Bose-Einstein probability distribution:

$$E[n_i(e_i)] = \frac{1}{e^{\frac{e_i - \mu}{kT}} - 1}$$

Compared to the Fermi-Dirac distribution, we observe that a “plus one” has changed to “minus one”. In our case, we propose to simply remove the “plus one”. With this change, if we replace $R^{1/T}$ by simply R , and we impose an upper bound of 1 to the probability, in order to get a single edge for every pair of nodes, we get:

$$P(i \leftrightarrow j) = \min \left\{ 1, \frac{R}{(r_i^\beta r_j^{\beta'} \theta_{ij})^{1/T}} \right\} \quad (3)$$

Now, R is *almost* a normalizing constant, since it multiplies one of the cases in the min. This probability distribution is quite similar to (1), and it results into SAT formulas quite similar to the ones obtained with the other model. This new model is quite similar to the GIRGs model, recently proposed in [14].

Since the probability is a function of $r_i^{-\beta} r_j^{-\beta'} \theta_{ij}$, adapting Lemma 2, we can see that the expected degree of node i is also $P(k_i) \sim r_i^{-\beta}$ and the generated formulas are also scale-free.

⁸Bosons are particles that are not restricted by the Pauli's exclusion principle. Thus, we may have several bosons in the same state.

The computation of R is also quite simple. We have to compute a set S of pairs of nodes whose probability is 1, and iterate the computation of R as follows. Initially, we set:

$$S_0 = \emptyset$$

$$R_0 = \frac{k m}{\sum_{i=1}^n \sum_{j=1}^m (r_i^\beta r_j^{\beta'} \theta_{ij})^{-1/T}}$$

and, at every iteration, compute

$$S_{l+1} = \{(i, j) \mid R_i / (r_i^\beta r_j^{\beta'} \theta_{ij})^{1/T} \geq 1\}$$

$$R_{l+1} = \frac{(k m - |S_{l+1}|)}{\sum_{\substack{i \in \{1, \dots, n\}, j \in \{1, \dots, m\} \\ (i, j) \notin S_{l+1}}} (r_i^\beta r_j^{\beta'} \theta_{ij})^{-1/T}}$$

until $S_l = S_{l+1}$ and the series converges.⁹ In practice, it converges in just 2 or 3 steps. At this point, Equation (2) holds.

This simplified model is the one used in the experiments. Let us finally recall the parameters of this model. The number of variables and clauses are n and m . K is the minimum clause size, and $(K + k)$ is the average clause size. The exponent of the power-law distributions of number of variable occurrences and clause size are respectively $\delta = 1 + 1/\beta$ and $\delta' = 1 + 1/\beta'$. When β and β' are very close to 0 (resp. to 1), there is almost no (resp. a high) variability. Finally, T is the temperature. When T is small, variables in each clause are chosen according to their popularity and similarity. In contrast, when the temperature is high enough, all variables have almost the same probability to be chosen at every step, and hence, the model behaves very similar to the classical random model.

4.4. Comparison to the PS model for graphs

The original Popularity-Similarity model deserves a special comparison to our model. Papadopoulos *et al.* [41] presented the first version of the model where, like in the *preferential attachment model*, new nodes are added to the network and connected to old nodes. Later, Krioukov *et al.* [35] (see also [13]) described a new version where, like in the *configuration model*, all nodes are added at once, and connections are defined by means of a connection probability function. Informally, they call the old model *Newtonian* and the new model *Einsteinian*. Our model is in essence a configuration model, therefore we will compare it with the Einsteinian version of the popularity-similarity model.

To synthesise a network with a desired number of nodes N , with power-law distribution of node degrees $P(k) \sim k^{-\delta}$, and average node degree $\bar{k}$ and clustering $\bar{C}$, they start with a circle of radius R in the hyperbolic plane with curvature $K = -\zeta^2$. Over this circle, they randomly distribute N points. In this hyperbolic geometry, the perimeter of a circle of radius r is $\frac{2\pi}{\zeta} \sinh(\zeta r)$ and the surface $\frac{2\pi}{\zeta} (\cosh(\zeta r) - 1)$. Both of them grow as $e^{\zeta r}$. Therefore, to obtain a uniform distribution of points one should assign to each point a random angle θ_i with uniform probability and a random radius r_i with probability density $\rho(r) \sim e^{\zeta r}$. If we connect every point with all other points at distance smaller than R , we get a scale-free network with exponent $P(k) \sim k^{-3}$. In order to obtain scale-free networks with other exponents $P(k) \sim k^{-\delta}$, we have to concentrate more points on the center or in the perimeter of the circle, using a probability density $\rho(r) \sim e^{\alpha r}$, with $\alpha \neq \zeta$. Then, the resulting exponent is $\delta = \max\{2, 2\alpha/\zeta + 1\}$. Since the exponent only depends on the fraction α/ζ , we can fix one of these parameters without loss of generality,¹⁰ so we will take $\zeta = 1$. In order to relax the model, they introduce a temperature T and use the Fermi-Dirac probability distribution $P(x) = \frac{1}{1 + e^{-\frac{\zeta(x-R)}{2T}}}$ to define the probability of connection between two points at distance x . They also approximate

⁹Notice that $R_{l+1} \leq R_l$ and $S_{l+1} \supseteq S_l$, for any l . Therefore, convergence is ensured.

¹⁰In [35], they fix $\alpha = 1/2$, whereas, in [13], they fix $\zeta = 1$.

Figure 2: Fraction of unsatisfiable formulas for the variable clause length model (left) and number of conflicts required by Minisat to solve them (right), for $n = 10^4$ variables. Every point is the mean for 100 instances.

the hyperbolic distance x_{ij} between two points with coordinates (r_i, θ_i) and (r_j, θ_j) as $x_{ij} = r_i + r_j + \frac{2}{\zeta} \log(\frac{\theta_{ij}}{2})$. With this probability distribution the exponent of the power-law is $\delta = \max\{2, 2\alpha/\zeta + 1, 2\alpha T/\zeta + 1\}$.

In our model, we take (basically) the same decisions, but we have to extend it to bi-partite graphs. Moreover, we are more focused on the computational aspects of how to generate a random instance rather than in the physical interpretation of the model. Therefore, we start by using a distinct definition of the radius, which takes values in $[0, 1]$ with uniform probability distribution. We also renounce to have an explicit (analytical) equation to compute the value of R . Instead, we will propose some methods to compute R that result into the desired number of edges in our random graph (hence in the size of our random SAT formula). In other words, R is for us nothing else than a kind of normalizing constant in our random distribution that we have to approximate. If we want to transform the radius r with values in $[0, R]$ and probability density $\rho(r) \sim e^{\alpha r}$ into a random variable $\hat{r}$ with values in $[0, 1]$ and uniform probability distribution, we have to apply the variable transformation $\hat{r} = \frac{e^{\alpha r} - 1}{e^{\alpha R} - 1}$. Since in the original model they assume $R \gg 1$ we have $\hat{r} \approx e^{\alpha(r-R)} - e^{-\alpha R}$. We go a step further and define $\hat{r} = e^{\alpha(r-R)}$, which of course results into a slightly different model, because the term $e^{-\alpha R}$ is not negligible for r close to zero. Now, replacing $x_{ij} = r_i + r_j + 2 \log(\frac{\theta_{ij}}{2})$ and $r = \frac{\log \hat{r}}{\alpha} + R$ into $P(x_{ij})$ we get:

$$P(i \leftrightarrow j) = \frac{1}{1 + e^{\frac{r_i + r_j + 2 \log(\frac{\theta_{ij}}{2}) - R}{2T}}} = \frac{1}{1 + e^{\frac{\frac{\log \hat{r}_i}{\alpha} + R + \frac{\log \hat{r}_j}{\alpha} + R + 2 \log(\frac{\theta_{ij}}{2}) - R}{2T}}} = \frac{1}{1 + \left(\frac{\hat{r}_i^{1/2\alpha} \hat{r}_j^{1/2\alpha} \theta_{ij}}{2e^{-R/2}} \right)^{1/T}}$$

Considering that for $T < 1$ we have $\delta = 2\alpha + 1$, and that in Lemma 2 we prove that $\delta = 1 + 1/\beta$, we can replace $\frac{1}{2\alpha}$ by β . Replacing $2e^{-R/2}$ by $\hat{R}$, we get

$$P(i \leftrightarrow j) = \frac{1}{1 + \left(\frac{\hat{r}_i^\beta \hat{r}_j^\beta \theta_{ij}}{\hat{R}} \right)^{1/T}}$$

This expression can be generalized, introducing a second exponent β' for bi-partite graphs, which is the expression that we introduced in Definition 1, where the hats are removed, understanding that the R and r values that we use are different from the ones used in the original Einsteinian model.

5. Models with Clauses of Distinct Lengths

The Popularity-Similarity model we have described generates clauses of distinct sizes, following a power-law distribution with exponent $\delta' = 1 + 1/\beta'$, as it has been observed in real-world industrial instances. However, the

satisfiability of formulas is very sensitive to the size of the clauses. In particular, if the formula contains (many) small –unary or binary– clauses, it likely becomes trivially unsatisfiable. Moreover, one of the properties we appreciate in the classical random model is the existence of a SAT-UNSAT phase transition phenomenon. This allows to generate formulas with 50% probability of being satisfiable, and hence, more adequate for competitions. However, it is difficult to define a model with both properties: distinct clause sizes and a sharp SAT-UNSAT phase transition at a constant clause/variable ratio.

In order to exemplify this fact, we are going to study what we can consider a *classical model* of random formulas with distinct sizes. This is the simplest model with maximal entropy –hence, generating the most difficult instances– that we can define. Even though, we show that the existence of small clauses in this model makes formulas trivial.

Given a number of variables n , number of clauses m and average clause size k , we can generate a random formula choosing a random variable $i \in \{1, \dots, n\}$ with uniform probability, a random clause $j \in \{1, \dots, m\}$ with uniform probability, and a random sign $s \in \{1, -1\}$ with probability $1/2$, adding $s \cdot i$ in j , and repeating this process km times. Finally, we remove empty clauses, if there is any.

We have studied the phase transition and the hardness of formulas for this simple model, for distinct values of k , and clause/variable ratios m/n . We show the results in Figure 2 for $n = 10^4$ variables. We observe that the transition between SAT and UNSAT is not so sharp as in regular random k -SAT formulas. The clause/variable ratio at the transition point increases with k , like the phase transition ratio for regular k -SAT. However, the value is not exactly the same. Moreover, if we repeat the experiment with $n = 10^5$ variables, we observe that almost all formulas in the represented ranges are trivially unsatisfiable. This means that the clause/variable ratio at the transition point is not constant. Finally, if we observe the number of conflicts needed to solve the formula, we see that, although the hardest formulas are located in the phase transition region, like in classical random formulas, they are trivially solvable for a modern SAT solver. The explanation of this phenomenon is the following. In this model, there is a constant fraction of unit clauses. The same variable will be chosen twice in unit clauses after $\Theta(\sqrt{n})$ choices, according to the birthday paradox. Therefore, it is likely to have two conflicting unit clauses.

The only way to avoid all these problems is not only removing empty clauses, but also unary (and probably binary) clauses. Alternatively, we can also force clauses to contain a minimum of K literals (now k and K will be parameters of the model). In this second case, we would assign K literals to each clause like in the classical model, and then we would add km more literals shared among all clauses, following the above method. This way, the average clause size would be $k + K$. This is exactly what we do in our model. However, in order to assign the K literals to each clause, instead of the classical method, we use the following method:

For every clause, and for 1 to K , we select a random variable i with probability $P(i) = 1/(1 + (r_i^\beta \theta_{ij}/R)^{1/T})$ and a random sign with probability $1/2$. This is exactly the same probability distribution described in Equation (1) removing the popularity of j . After that, we generate the km additional literals as described in Section 4, avoiding repetitions of variables in clauses.

6. Locality and CDCL SAT Solving

In this section, we first analyze how VSIDS, one of the most common heuristic in CDCL SAT solvers, takes advantage of both the popularity and the similarity of variables of PS random formulas. Second, we compare the performance of CDCL SAT solving techniques to Look-ahead SAT solvers, a family of techniques that shows a better performance on (classical) random k -SAT. Finally, we evaluate several state-of-the-art CDCL SAT solvers on PS random formulas.

We recall that the parameter of the PS model used in our experiments are described at the end of Sect. 4.3. Since in this section we are just interested in analyzing the effects of popularity and similarity of variables in SAT solving performance –and not the effects of different clause sizes–, we limit our experimentation to the case $K = 3$ and $k = 0$ (thus $\beta' = 0$), i.e., all clauses have exactly 3 literals.

In Figure 3, we present the graph representing a PS instance generated with $n = 100$, $m = 425$, $K = 3$, $k = 0$, $\beta = 0.8$, and $T = 0$. In this bipartite graph, red nodes represent variables and blue nodes represent clauses, whereas an edge between a variable-node and a clause-node indicates that such a clause contains that variable. The position (x, y) of each node represents respectively its radial and angular coordinates. Radial coordinates are represented in logarithmic scale. Notice that clauses mostly contain either very popular variables (variables with small radius) or variables similar to them (variables with a similar angle).

Figure 3: Radial and angular coordinates of variables (red circles) and clauses (blue squares) nodes of a PS instance generated with $n = 100$, $m = 425$, $K = 3$, $k = 0$, $\beta = 0.8$, and $T = 0$.

6.1. VSIDS on PS random formulas

In a first experiment, we show how VSIDS exploits both the popularity and the similarity of variables by analyzing the variables selected by the branching heuristic of the well-known CDCL SAT solver Glucose [9]. VSIDS rewards the variables occurring in the last conflicts, and hence they are more likely to be selected in the next branching steps. We generate some extreme cases of our model to show these results.

In Figure 4, we represent the variables chosen by VSIDS along the search, for some formulas of the PS model generated with $n = 5000$ and $m/n = 4.25$. Each point (i, j) represents that variable j was decided between the i -th and the $(i + 1)$ -st conflicts. We represent three cases. First, a formula with high popularity ($\beta = 0.8$) and small temperature ($T = 1.50$); variables indexes ordered by popularity (the lower the index of a variable, the more popular it is). Second, a formula with high similarity ($\beta = 0.1$) and small temperature ($T = 0.75$); variables indexes ordered by similarity (variables with close indexes have close angular coordinates, thus the first variables have similar angles than the last ones). Finally, a third formula with high temperature ($T = 100$). In this last case, $\beta = 0.1$ and variables are ordered by similarity; similar results occur with other β or if ordered by popularity. Although we only represent the first 30,000 conflicts, the same behavior can be observed during the whole execution.

We can observe that in the formula with high popularity, the solver decides in popular variables (low variable indexes). Ansótegui *et al.* [2] already showed that VSIDS tends to branch on popular variables. Interestingly, in the formula with high similarity, we observe that the solver is focused on a certain area of the angular space, i.e., similar variables. This suggests that the solver is indeed also taking advantage of the notion of locality in the formula. Finally, if the formula has neither popularity nor similarity –because the temperature is too high–, decisions affects all variables; the solver does not seem to be focused on any particular subset of variables at any part of the search. This

Figure 4: Analysis of the branching variables decisions performed by Glucose in some formulas of the PS model, with high popularity (top), high similarity (center), and none of them (bottom). Vertical lines represent restarts.

is expected because this formula is more similar to a classical random CNF.

When a SAT formula has popularity and similarity, the CDCL solver is able to exploit these two structures, finding a good balance between them. A possible explanation is that the solver is finding conflicts faster, and these conflicts relate fewer variables. In the formula with high popularity, the average clause size of the learned clauses is 26.59 literals, and conflicts occur at a rate of 627 conflicts per second; in the formula with high similarity 17.08 literals and 1053 conflicts per second. On the other hand, in the formula with high temperature, it is observed that after each restart (vertical lines), the solver makes many decisions, suggesting that it is harder to find conflicts in this formula (as it happens in random SAT instances). In fact, the average clause size for the learned clauses is 103.71 literals, and the learning rate is 223 conflicts per second.

Finally, it is interesting to mention that the restarts seems not to have any impact on this behavior of VSIDS. In the same experiment carried out with MiniSat, we obtained the same results. Notice that, in contrast to MiniSat, which implements non-adaptive restarts (based on Luby), Glucose uses adaptive restarts based on the LBD of the learned clauses. This difference can be seen in the vertical lines of the three plots in this figure, which represent the restarts and are positioned at different moments of the search. It is interesting to see that the frequency of the restarts does not alter the general behavior of the branching heuristic VSIDS in Glucose or in MiniSat. We emphasize that achieving the same results with both solvers reinforces our hypothesis about the exploitation of both popularity and similarity of variables by CDCL SAT solvers, and more specifically, by VSIDS.

6.2. CDCL vs Look-Ahead on PS random formulas

In a second experiment, we show that CDCL SAT solvers indeed outperform other solving techniques in PS random formulas. To this purpose, we evaluate the performances of the CDCL SAT solver Glucose [9] and the Look-ahead SAT solver March [28] (which is more efficient in classical random SAT formulas), on some families of random PS formulas with different temperatures T . All formulas are generated in the SAT-UNSAT phase transition region with $m/n = 4.25$. The size of the formulas is $n = 5000$ when $T < 2$, and $n = 300$ otherwise. Notice that actual

Figure 5: Scatter plot of runtimes of Glucose and March on different PS instances, for different temperatures T .

industrial formulas have millions of variables, but $n = 5000$ is enough to show the effects of popularity and similarity. On the contrary, when the temperature is high, the formulas are similar to random CNF. No complete solver could even solve a hard random instance with $n = 5000$ variables (in a reasonable timeout). So, we need to decrease the formula size to analyze these cases.

In Figure 5, we plot the results. It can be seen that when the temperature is low (see $T = 1.40$), all formulas are very easy, and both solvers solve them in a few seconds. Interestingly, as we increase the temperature T , formulas become harder (as expected), but the CDCL solver is much more efficient solving most of them (see $T = 1.50$). This happens until the temperature is very high, when both solvers fail to solve any formula (see $T = 1.55$). Finally, if the temperature keeps growing, the formulas become easier for March (see $T = 10$), as expected.

6.3. CDCL SAT solvers on PS random formulas

In a third experiment, we evaluate a set of CDCL SAT solvers on a set of challenging PS random SAT instances. The solvers are: (i) MiniSat [17], which is possibly the most well-known CDCL SAT solver, (ii) Glucose [9] and (iii) Lingeling [11], two of the best CDCL SAT solvers in the last years, (iv) MapleLCMDiscChronoBT-DL-v3 [34] (from now, we refer this solver as MapleLCM for short), which was the winner of the SAT Race 2019 (in fact, the three versions of this solver ranked from first to third positions of this competition), and (v) CaDiCaL [12], which ranked in fourth position in that competition.

In order to evaluate these CDCL SAT solvers, we generate a set of challenging PS random SAT instances, which can be found in the repository of the generator.¹¹ This set contains a total of 960 SAT formulas, both satisfiable and unsatisfiable. All the instances are 3-CNF (i.e., $K = 3$, $k = 0$, and $\beta' = 0$), have $n = 1000$ variables, and the number of clauses ranges from $m = 3500$ to $m = 5000$. The scale-free structure of the variable occurrences and the temperature is set to: (i) $T = \{1.2, 1.3\}$ when $\beta = 1/2$, (ii) $T = \{1.6, 1.7\}$ when $\beta = 2/3$, and (iii) $T = \{1.9, 2.0\}$ when $\beta = 5/6$. This results into a set of challenging SAT benchmarks, which can be usually solved by state-of-the-art CDCL SAT solver in the order of seconds or minutes.

¹¹<https://github.com/jgiraldd/ps-sat>

Figure 6: Cactus plot of CDCL SAT solvers on PS random SAT instances.

We solve all these formulas in an Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz with 16GB of RAM memory. All experiments were executed with a timeout of 600 seconds.

In Figure 6, it is depicted the cactus plot of the five CDCL SAT solvers aforementioned on the set of challenging random PS instances. We also report a summary of the results of this experiment in Table 1, where we distinguish between SAT and UNSAT formulas. In Appendix A, we report the detailed results of this experiment for each subfamily of PS formulas.

It can be observed that MiniSat is the solver showing the worst performance, as expected. On the other hand, MapleLCM is clearly the solver showing the best performance. This solver implements an heuristic to handle duplicate learned clauses. This seems to be beneficial to solve PS formulas, where the concept of locality may cause such duplicate learned clauses. We also observe that Glucose, Lingeling, and CaDiCaL show a very similar performance in the timeout of 600 seconds. However, with a shorter timeout, Lingeling clearly shows a worse performance than the other two solvers. In this case, the inprocessing techniques to detect variable dependencies implemented in Lingeling seems to hurt its performance on PS formulas, especially on unsatisfiable ones. On the contrary, the performance of CaDiCaL at this shorter timeouts is very similar to MapleLCM. CaDiCaL also implements inprocessing techniques but its design focus on improving its performance on satisfiable formulas. Our experimental results suggests that these modifications help to improve the performance of CDCL SAT solvers on PS formulas.

7. Empirical Analysis of the SAT-UNSAT Phase Transition

In this section we present an empirical analysis of the phase transition SAT-UNSAT of the Popularity-Similarity model for random SAT instances. The aim of this analysis is to provide empirical data about the satisfiability of PS instances in a varied set-up of its parameters.

We present experimental results on three kind of experimental settings. First, we analyze whether the temperature T affects the phase transition point (PTP). To this purpose, we generate families of PS formulas with different

Solver	SAT	UNSAT	Total
MiniSat	145	629	774
Glucose	128	666	794
Lingeling	139	654	793
MapleLCM	154	685	839
CaDiCaL	180	622	802

Table 1: Summary of performance of CDCL SAT solvers on PS random SAT instances. The best results for each subset of formulas are marked in bold.

clause/variable ratios m/n and analyze the PTP for different temperatures T . We perform this study for 2-CNF and 3-CNF, i.e., $K = 2$ and $K = 3$, for four representative values of β . In these experiments, the number of variables n must be desirably fixed as high as possible. However, this also increases the solving time of the formula. Therefore, we need to find a reasonable trade-off between both.¹²

Second, we study the PTP with respect to the formula size. In particular, we want to study whether empirical results suggest that the PTP tends to a particular point as the formula size increases. To this end, we generate families of PS instances with an increasing number of variables n , and analyze the PTP for different number of clauses m . In this experiment, we use a fixed temperature $T = 0$, which is possibly the most interesting case from a theoretical point of view. We also analyze this behavior for both 2-CNF and 3-CNF.

Finally, we analyze the relation between the PTP and the lengths of clauses in the formulas. In particular, we generate families of PS random problems with the same average clause length (i.e., $K + k$), differing in the minimum clause length (i.e., K). In particular, we generate three cases where $K + k = 5$, with (a) $K = 5, k = 0$, (b) $K = 4, k = 1$, and (c) $K = 3, k = 2$. In cases (b) and (c), where $k > 0$, we also generate several families of PS formulas varying the variability of clauses lengths (parameter β').

The following subsections analyze in details these three experiments.

7.1. Clause/Variable Ratio Versus Temperature

Let us start by the simplest case of 2-CNF SAT formulas. This is, fixing $K = 2$ and $k = 0$. In Figure 7, we represent the percentage of unsatisfiable formulas, varying the temperature $T \in [0, 4]$ (step 0.1) w.r.t. the clause/variable ratio $m/n \in [0.05, 1.45]$ (step 0.05), for four families of PS formulas differing in the variability of variables occurrences, with $\beta = \{0.25, 0.50, 0.75, 1.00\}$. The percentage of unsatisfiable formulas is computed among 10 random PS SAT instances generated in each point (x, y) of the plots. In all the formulas, the number of variables is $n = 2000$. Notice that green (resp. red) points represent families of only satisfiable (resp. only unsatisfiable) formulas, whereas black points represent formulas in the phase transition point.

A first observation is that in all the four cases analyzed (differing in β), the PTP at high temperatures is close to $m/n = 1$, which is the PTP of 2-CNF in the classical random model. As seen in the previous section, this indicates that the PS model at high temperatures behaves similarly to the classical random model. However, for higher values of β , this behavior starts to occur at higher temperatures. For instance, the PTP for $\beta = 0.25$ is $m/n \approx 1$ at $T > 1$, whilst for $\beta = 1$ it occurs at $T > 3.5$.

We also observe in the four plots that at low temperatures, the PTP is located at lower clause/variables ratios m/n . See, for instance, that at $T = 0$ the PTP for $\beta = 0.25$ is located at $m/n \approx 0.35$. Interestingly, this PTP also decreases for higher values of β . For $\beta = 1$ at $T = 0$, this PTP is $m/n \approx 0.05$.

This suggests that the PTP depends on the clause/variable ratio (as expected), but it also depends on the temperature and the exponent of the scale-free structure β . We conjecture that for large enough formulas (i.e., large enough formula size), the PTP of the PS model at high temperatures is the same than the PTP of the classical random model. However, at low temperatures, it depends on β . In particular, we conjecture that if β is high enough, the PTP tends to $m/n = 0$ (for low temperatures).

Now, we analyze the same case for 3-CNF, i.e., with $K = 3$ and $k = 0$. In this case, $m/n \in [0.1, 4.9]$ (step 0.1), $T \in [0, 2]$ (step 0.1), $n = 1000$, and $\beta = \{0.50, 0.67, 0.83, 1.00\}$. In Figure 8, we represent the percentage

¹²In order to generate all the plots in this section, around one million SAT formulas were solved.

Figure 7: Phase Transition of 2-CNF PS SAT formulas varying the temperature T w.r.t. the clause/variable ratio m/n , for distinct values of $\beta = \{0.25, 0.50, 0.75, 1.00\}$.

of unsatisfiable formulas in this set. Notice that the *white* points in these heatmaps represent families of formulas unsolved in the timeout. In Figure 9, we represent the solving time of these formulas.

As in the previous experiment, the higher the temperature, the higher the clause/variable ratio of the PTP. As conjectured before, the PTP of the PS model at high temperatures may be the same as the PTP of the classical random model, located at $m/n \approx 4.26$. Unfortunately, this is hard to check empirically, since formulas become very hard (see Fig. 9).

We can also observe that at higher temperatures, the phase transition region seems to be narrower than at lower temperatures. This is, the transition between satisfiable and unsatisfiable formulas occurs faster. We conjecture that if the temperature is low enough (i.e., when the value of β makes any effect in the model), the PTP is not sharp. See for instance the cases for $\beta = 0.83$ and $\beta = 1.00$ in Figure 8. In the top right corner of the plots, this is, for high temperatures and high clause/variable ratios, there are few dark points, i.e., most of the points contain families of either only satisfiable instances or only unsatisfiable ones. In contrast, in the bottom left corner of the plots, this is, for low temperatures and low clause/variable ratios, there are many dark points, and hence it is not clear where the PTP is exactly located.

The previous conjectures for 2-CNF seem to hold for 3-CNF as well.

7.2. Number of Variable Versus Number of Clauses

Now, we analyze whether the PTP tends to a particular clause/variable ratio m/n for an increasing number of variables n . As we have seen before, this PTP also depends on the parameters β and T of the model. In the case of β , we again evaluate four representative values. For the temperature, we have showed that the PS model with high

Figure 8: Phase Transition of 3-CNF PS SAT formulas varying the temperature T w.r.t. the clause/variable ratio m/n , for distinct values of $\beta = \{0.50, 0.67, 0.83, 1.00\}$.

Figure 9: Hardness (solving time) of the same formulas represented in Figure 8.

Figure 10: Phase Transition of 2-CNF PS SAT formulas varying the clause/variable ratio m/n w.r.t. the number of variables n at temperature $T = 0$, for distinct values of $\beta = \{0.25, 0.50, 0.75, 1.00\}$.

temperature T behaves similarly to the classical random model, and hence, it seems more interesting to analyze those cases with low temperatures. In particular, we focus on the extreme case with $T = 0$.

First, we carry out an experiment in 2-CNF PS random formulas, generated with $\beta = \{0.25, 0.50, 0.75, 1.00\}$. We generate families of formulas with $n \in [100, 10000]$ (step 100) and $m/n \in [0.05, 1]$ (step 0.05). We represent these results in Figure 10.

We can observe in the four plots that the PTP seems to tend to a particular value of m/n , but this value differs of each β . In particular, the lower the value of β is, the higher the PTP becomes. See, for instance, the PTP with $\beta = 0.25$ in the largest formulas is around $m/n \approx 0.25$, whereas this point for $\beta = 0.75$ is around $m/n \approx 0.1$. In fact, for $\beta = 1$, it seems the PTP is located at $m/n = 0$, confirming our previous conjecture.

We repeat this experiment for 3-CNF PS instances. We generate these formulas with $\beta = \{0.50, 0.67, 0.83, 1.00\}$, $n \in [100, 4900]$ (step 100) and $m/n \in [0.1, 3]$ (step 0.1). We represent these results in Figure 11. As observed before, with low values of β , the PTP seems to be located at a clause/variable ratio m/n greater than zero (e.g., $m/n \approx 0.25$ for $\beta = 0.5$ in 2-CNF). In contrast, with high β , the phase transition point seems to tend to $m/n = 0$.

7.3. Variable Clause Length

Finally, we analyze the effect of clause length variations on the PTP. To this purpose, we generate several families of PS formulas, all having the average clause size $K + k = 5$. In particular, we study the cases: (i) $K = 5, k = 0$, (ii) $K = 4, k = 1$, and (iii) $K = 3, k = 2$. Notice that in all these cases, the minimum clause length is 3, in order to avoid trivially unsatisfiable formulas (as explained in Section 5). When $k > 0$, we analyze three cases with low, medium and high variability of clauses lengths, i.e., $\beta' = \{0.5, 0.75, 1\}$. In all cases, we fix $n = 500$ and $\beta = 0.75$.

Figure 11: Phase Transition of 3-CNF PS SAT formulas varying the clause/variable ratio m/n w.r.t. the number of variables n at temperature $T = 0$, for distinct values of $\beta = \{0.50, 0.67, 0.83, 1.00\}$.

The formulas are generated with $m/n \in [0.5, 20]$ (step 0.1), and $T \in [0, 1]$ (step 0.1). In Figure 12, we represent the percentage of unsatisfiable formulas of this study.

A first observation is that the existence of variations in the clause lengths makes, in general, the formulas easier (not showed in the plots). This happens for all temperatures and all clause/variable ratios (not showed in the figure), although the differences specially increase for high temperatures.

It is interesting to observe that the PTP is located at lower clause/variable ratios m/n for higher values of k and higher values of β' . In other words, when there is more variability in the clause length. Therefore, we conjecture that the PTP also tends to $m/n = 0$ if the formula size is large enough, while T and β' are low enough.

8. Conclusions

In this paper, we present the Popularity-Similarity model for random SAT instances. It generates formulas with a power-law distribution in the number of variable occurrences (popularity), and high clustering between them (similarity). To the best of our knowledge, this is the first model that combines these two typical properties of real-world SAT instances. Additionally, it also generates formulas with variable clause size, which is also a common feature of real-world SAT benchmarks.

We show that popularity and similarity are two orthogonal forces exploited by VSIDS. Also, we show that the performance of CDCL SAT solvers (w.r.t. other solving techniques) is related to the existence of popularity and similarity in the SAT formulas. These observations suggest that both popularity and similarity are two crucial properties to understand the success of CDCL on application SAT benchmarks.

Figure 12: Phase Transition of 5-CNF PS SAT formulas varying the temperature T w.r.t. the clause/variable ratio m/n , for $K = 5, k = 0$ (top), $K = 4, k = 1$ (center), and $K = 3, k = 2$ (bottom), and distinct values of $\beta' = \{0.50, 0.75, 1.00\}$ (when $k > 0$).

Finally, we have empirically analyzed the SAT-UNSAT phase transition phenomenon. Based on this empirical analysis, we conjecture: (a) if the temperature of the model is high enough, the model behaves as the classical random SAT model, and (b) if the temperature is low enough (see Fig. 10 and 11), on the existence of scale-free structure (i.e., high popularity) the phase transition point seems to tend to the clause/variable ratio $m/n = 0$, when β is close to one, and to a non-zero value, for values of β closer to zero. Similarly, in scale-free random SAT instances [8], the transition is at $m/n = 0$, when $\beta > 1 - 1/k$, and at $m/n > 0$, when $\beta < 1 - 1/k$.

Acknowledgements

This work is partially supported by the EU H2020 Research and Innovation Programme under the LOGISTAR project (Grant Agreement No. 769142), and by the Spanish Ministry of Science, Innovation and Universities and the Spanish National Agency of Research (AEI) under the projects RASO (TIN2015-71799-C2-1-P) and EXASOCO (PGC2018-101216-B-I00), and by the Andalusian Government and the University of Granada under project AIMAR (A-TIC-284-UGR18), including European Regional Development Funds (ERDF). The first author is also supported by a MICINN Juan de la Cierva fellowship (grant FJCI-2017-32420).

References

References

- [1] Dimitris Achlioptas, Carla Gomes, Henry Kautz, and Bart Selman. Generating satisfiable problem instances. In *Proc. of the 17th National Conf. on Artificial Intelligence (AAAI'00)*, pages 256–261, 2000.
- [2] Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. On the structure of industrial SAT instances. In *Proc. of the 15th Int. Conf. on Principles and Practice of Constraint Programming (CP'09)*, pages 127–141, 2009.
- [3] Carlos Ansótegui, Jesús Giráldez-Cru, and Jordi Levy. The community structure of SAT formulas. In *Proc. of the 15th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'12)*, pages 410–423, 2012.
- [4] Carlos Ansótegui, Maria Luisa Bonet, Jesús Giráldez-Cru, and Jordi Levy. The fractal dimension of SAT formulas. In *Proc. of the 7th Int. Joint Conf. on Automated Reasoning (IJCAR'14)*, pages 107–121, 2014.
- [5] Carlos Ansótegui, Jesús Giráldez-Cru, Jordi Levy, and Laurent Simon. Using community structure to detect relevant learnt clauses. In *Proc. of the 18th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'15)*, pages 238–254, 2015.
- [6] Carlos Ansótegui, Maria Luisa Bonet, Jesús Giráldez-Cru, Jordi Levy, and Laurent Simon. Community structure in industrial SAT instances. *Journal of Artificial Intelligence Research*, 66:443–472, 2019.
- [7] Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. Towards industrial-like random SAT instances. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI'09)*, pages 387–392, 2009.
- [8] Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. Scale-free random SAT instances. *CoRR*, abs/1708.06805, 2017.
- [9] Gilles Audemard and Laurent Simon. Predicting learnt clauses quality in modern SAT solvers. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI'09)*, pages 399–404, 2009.
- [10] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
- [11] Armin Biere. CaDiCaL, Lingeling, Plingeling, Treengeling, YalSAT entering the SAT Competition 2017. In *Proceedings of SAT Competition 2017: Solver and Benchmark Descriptions*, pages 14–15, 2017.
- [12] Armin Biere. CaDiCaL at the SAT Race 2019. In *Proceedings of SAT Race 2019 : Solver and Benchmark Descriptions*, pages 8–9, 2019.
- [13] M. Boguñá, F. Papadopoulos, and D. Krioukov. Sustaining the Internet with Hyperbolic Mapping. *Nature Communications*, 1(62), Oct 2010.
- [14] Karl Bringmann, Ralph Keusch, and Johannes Lengler. Geometric inhomogeneous random graphs. *Theoretical Computer Science*, 760:35 – 54, 2019.
- [15] Colin Cooper, Alan Frieze, and Gregory B. Sorkin. Random 2-SAT with prescribed literal degrees. *Algorithmica*, 48(3):249–265, July 2007.
- [16] Rina Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.
- [17] Niklas Eén and Niklas Sörensson. An extensible SAT-solver. In *Proc. of the 6th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'03)*, pages 502–518, 2003.
- [18] J. W. Freeman. *Improvements to Propositional Satisfiability Search Algorithms*. PhD thesis, University of Pennsylvania, Philadelphia, 1995.
- [19] Tobias Friedrich and Ralf Rothenberger. Sharpness of the satisfiability threshold for non-uniform random k-SAT. In *Theory and Applications of Satisfiability Testing (SAT'18)*, pages 273–291, 2018. Best Paper Award.
- [20] Tobias Friedrich and Ralf Rothenberger. The satisfiability threshold for non-uniform random 2-SAT. In *International Colloquium on Automata, Languages and Programming (ICALP'19)*, 2019.
- [21] Tobias Friedrich, Anton Krohmer, Ralf Rothenberger, Thomas Sauerwald, and Andrew M. Sutton. Bounds on the satisfiability threshold for power law distributed random SAT. In *European Symposium on Algorithms (ESA'17)*, volume 87 of *LIPIcs*, pages 37:1–37:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, sep 2017.
- [22] Tobias Friedrich, Anton Krohmer, Ralf Rothenberger, and Andrew M. Sutton. Phase transition for sclae-free SAT formulas. In *Proc. of the 31st National Conf. on Artificial Intelligence (AAAI'17)*, 2017.
- [23] Ian P. Gent, Holger H. Hoos, Patrick Prosser, and Toby Walsh. Morphing: Combining structure and randomness. In *Proc. of the 16th National Conf. on Artificial Intelligence (AAAI'99)*, pages 654–660, 1999.
- [24] Jesús Giráldez-Cru and Jordi Levy. A modularity-based random SAT instances generator. In *Proc. of the 24th Int. Joint Conf. on Artificial Intelligence (IJCAI'15)*, pages 1952–1958, 2015.
- [25] Jesús Giráldez-Cru and Jordi Levy. Generating SAT instances with community structure. *Artif. Intell.*, 238:119–134, 2016.
- [26] Jesús Giráldez-Cru and Jordi Levy. Locality in random SAT instances. In *Proc. of the 26th Int. Joint Conf. on Artificial Intelligence (IJCAI'17)*, pages 638–644, 2017.
- [27] Carla P. Gomes and Bart Selman. Problem structure in the presence of perturbations. In *Proc. of the 14th National Conf. on Artificial Intelligence (AAAI'97)*, pages 221–226, 1997.
- [28] Marijn J. H. Heule, Joris E. van Zwieten, Mark Dufour, and Hans van Maaren. March.eq: Implementing additional reasoning into an efficient Look-ahead SAT solver. In *Proc. of the 7th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'04)*, pages 345–359, 2004.
- [29] Matti Järvisalo, Petteri Kaski, Mikko Koivisto, and Janne H. Korhonen. Finding efficient circuits for ensemble computation. In *Proc. of the 15th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'12)*, pages 369–382, 2012.
- [30] Robert G. Jeroslow and Jinchang Wang. Solving propositional satisfiability problems. *Annals of Mathematics and Artificial Intelligence*, 1(1):167–187, 1990.
- [31] George Katsirelos and Laurent Simon. Eigenvector centrality in industrial SAT instances. In *Proc. of the 18th Int. Conf. on Principles and Practice of Constraint Programming (CP'12)*, pages 348–356, 2012.
- [32] Henry A. Kautz and Bart Selman. Ten challenges redux: Recent progress in propositional reasoning and search. In *Proc. of the 9th Int. Conf. on Principles and Practice of Constraint Programming (CP'03)*, pages 1–18, 2003.
- [33] Henry A. Kautz and Bart Selman. The state of SAT. *Discrete Applied Mathematics*, 155(12):1514–1524, 2007.
- [34] Stepan Kochemazov, Oleg Zaikin, Victor Kondratiev, and Alexander Semenov. MapleLCMDistChronoBT-DL, duplicate learnts heuristic-aided solvers at the SAT Race 2019. In *Proceedings of SAT Race 2019 : Solver and Benchmark Descriptions*, page 24, 2019.

- [35] Dmitri Krioukov, Fragkiskos Papadopoulos, Amin Vahdat, and Marián Boguñá. Curvature and temperature of complex networks. *Phys. Rev. E*, 80:035101, Sep 2009.
- [36] Jia Hui Liang, Vijay Ganesh, Pascal Poupart, and Krzysztof Czarnecki. Learning rate based branching heuristic for SAT solvers. In Nadia Creignou and Daniel Le Berre, editors, *Proc. of the 19th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'16)*, volume 9710 of *Lecture Notes in Computer Science*, pages 123–140. Springer, 2016.
- [37] Joao P. Marques-Silva. The impact of branching heuristics in propositional satisfiability algorithms. In *Proc. of the 9th Portuguese Conf. on Artificial Intelligence: Progress in Artificial Intelligence (EPIA'99)*, volume 1695 of *LNCS*, pages 62–74, 1999.
- [38] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient sat solver. In *Proc. of the 38th Annual Design Automation Conf. (DAC'01)*, pages 530–535, 2001.
- [39] Nathan Mull, Daniel J. Fremont, and Sanjit A. Seshia. On the hardness of SAT with community structure. In *Proc. of the 19th Int. Conf. on Theory and Applications of Satisfiability Testing (SAT'16)*, pages 141–159, 2016.
- [40] Oleksii Omelchenko and Andrei A. Bulatov. Satisfiability threshold for power law random 2-SAT in configuration model. *CoRR*, abs/1905.04827, 2019.
- [41] Fragkiskos Papadopoulos, Maksim Kitsak, M. Ángeles Serrano, Marián Boguñá, and Dimitri Krioukov. Popularity versus similarity in growing networks. *Nature*, 489:537–540, 2012.
- [42] Bart Selman, Henry A. Kautz, and David A. McAllester. Ten challenges in propositional reasoning and search. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI'97)*, pages 50–54, 1997.

Appendix A. Detailed results of CDCL SAT solvers on PS random formulas

In this appendix, we include the detailed results of the five CDCL SAT solvers evaluated on solving the set of challenging PS random SAT instances described in Section 6.3. In particular, the results are reported in Tables A.2 and A.3, distinguishing between satisfiable and unsatisfiable instances.

m	β	T	MiniSat	Glucose	Lingeling	MapleLCM	CaDiCal
3500	0.50	1.2	10 - 0	10 - 0	10 - 0	10 - 0	10 - 0
		1.3	10 - 0	10 - 0	10 - 0	10 - 0	10 - 0
	0.67	1.6	10 - 0	10 - 0	10 - 0	10 - 0	10 - 0
		1.7	10 - 0	10 - 0	10 - 0	10 - 0	10 - 0
	0.83	1.9	10 - 0	10 - 0	10 - 0	10 - 0	10 - 0
		2.0	9 - 1	9 - 1	9 - 1	9 - 1	9 - 1
3600	0.50	1.2	8 - 1	8 - 1	7 - 1	9 - 1	9 - 1
		1.3	10 - 0	9 - 0	10 - 0	10 - 0	10 - 0
	0.67	1.6	10 - 0	8 - 0	9 - 0	10 - 0	10 - 0
		1.7	10 - 0	10 - 0	10 - 0	10 - 0	10 - 0
	0.83	1.9	7 - 3	5 - 3	5 - 3	7 - 3	7 - 3
		2.0	9 - 1	8 - 1	9 - 1	9 - 1	9 - 1
3700	0.50	1.2	6 - 2	4 - 2	7 - 2	7 - 2	8 - 2
		1.3	6 - 0	4 - 0	4 - 0	8 - 0	10 - 0
	0.67	1.6	2 - 5	2 - 5	4 - 5	3 - 5	5 - 5
		1.7	6 - 1	2 - 1	3 - 1	6 - 1	9 - 1
	0.83	1.9	2 - 5	1 - 6	1 - 6	2 - 6	2 - 5
		2.0	4 - 4	3 - 4	5 - 4	5 - 4	6 - 4
3800	0.50	1.2	1 - 4	1 - 4	1 - 4	1 - 5	3 - 4
		1.3	1 - 0	1 - 0	1 - 0	5 - 0	8 - 0
	0.67	1.6	1 - 5	2 - 5	1 - 5	0 - 6	1 - 5
		1.7	1 - 1	1 - 2	0 - 2	2 - 2	6 - 1
	0.83	1.9	0 - 9	0 - 9	1 - 9	0 - 9	1 - 9
		2.0	1 - 5	0 - 5	2 - 5	1 - 5	4 - 5
3900	0.50	1.2	0 - 6	0 - 6	0 - 6	0 - 6	0 - 6
		1.3	1 - 1	0 - 1	0 - 1	0 - 1	2 - 1
	0.67	1.6	0 - 7	0 - 8	0 - 8	0 - 8	0 - 7
		1.7	0 - 4	0 - 4	0 - 4	0 - 4	1 - 4
	0.83	1.9	0 - 9	0 - 9	0 - 9	0 - 10	0 - 9
		2.0	0 - 6	0 - 8	0 - 7	0 - 9	0 - 6
4000	0.50	1.2	0 - 2	0 - 4	0 - 4	0 - 4	0 - 2
		1.3	0 - 1	0 - 1	0 - 1	0 - 2	0 - 1
	0.67	1.6	0 - 9	0 - 9	0 - 9	0 - 9	0 - 9
		1.7	0 - 1	0 - 3	0 - 3	0 - 4	0 - 1
	0.83	1.9	0 - 9	0 - 9	0 - 9	0 - 9	0 - 9
		2.0	0 - 7	0 - 7	0 - 7	0 - 9	0 - 7
4100	0.50	1.2	0 - 5	0 - 7	0 - 6	0 - 7	0 - 5
		1.3	0 - 1	0 - 1	0 - 1	0 - 1	0 - 1
	0.67	1.6	0 - 9	0 - 9	0 - 9	0 - 9	0 - 9
		1.7	0 - 3	0 - 6	0 - 3	0 - 6	0 - 3
	0.83	1.9	0 - 9	0 - 10	0 - 10	0 - 10	0 - 9
		2.0	0 - 9	0 - 9	0 - 9	0 - 9	0 - 7
4200	0.50	1.2	0 - 8	0 - 9	0 - 9	0 - 9	0 - 8
		1.3	0 - 2	0 - 4	0 - 4	0 - 5	0 - 1
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 6	0 - 7	0 - 7	0 - 9	0 - 6
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 9	0 - 9	0 - 9	0 - 10	0 - 9

Table A.2: Detailed results of CDCL SAT solvers on PS random SAT instances (part I), distinguishing between SAT (left) and UNSAT (right) instances. The best results for each subset of formulas are marked in bold.

m	β	T	MiniSat	Glucose	Lingeling	MapleLCM	CaDiCal
4300	0.50	1.2	0 - 9	0 - 10	0 - 10	0 - 10	0 - 9
		1.3	0 - 2	0 - 5	0 - 3	0 - 7	0 - 2
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 8	0 - 9	0 - 8	0 - 10	0 - 8
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
4400	0.50	1.2	0 - 8	0 - 8	0 - 8	0 - 8	0 - 8
		1.3	0 - 5	0 - 6	0 - 5	0 - 8	0 - 4
	0.67	1.6	0 - 8	0 - 8	0 - 8	0 - 8	0 - 8
		1.7	0 - 8	0 - 8	0 - 8	0 - 8	0 - 6
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
4500	0.50	1.2	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.3	0 - 7	0 - 7	0 - 7	0 - 8	0 - 6
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 8	0 - 10	0 - 10	0 - 10	0 - 8
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
4600	0.50	1.2	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.3	0 - 7	0 - 9	0 - 9	0 - 9	0 - 7
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 10	0 - 10	0 - 10	0 - 10	0 - 9
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
4700	0.50	1.2	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.3	0 - 9	0 - 9	0 - 9	0 - 9	0 - 9
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
4800	0.50	1.2	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.3	0 - 7	0 - 10	0 - 9	0 - 10	0 - 7
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 8	0 - 9	0 - 8	0 - 10	0 - 8
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
4900	0.50	1.2	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.3	0 - 8	0 - 9	0 - 9	0 - 9	0 - 8
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 9	0 - 10	0 - 9	0 - 10	0 - 9
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
5000	0.50	1.2	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.3	0 - 9	0 - 10	0 - 10	0 - 10	0 - 10
	0.67	1.6	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		1.7	0 - 9	0 - 10	0 - 10	0 - 10	0 - 9
	0.83	1.9	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10
		2.0	0 - 10	0 - 10	0 - 10	0 - 10	0 - 10

Table A.3: Detailed results of CDCL SAT solvers on PS random SAT instances (part II), distinguishing between SAT (left) and UNSAT (right) instances. The best results for each subset of formulas are marked in bold.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

□