



On the Empirical Time Complexity of Scale-Free 3-SAT at the Phase Transition

Thomas Bläsius^{1(✉)}, Tobias Friedrich¹, and Andrew M. Sutton²

¹ Hasso Plattner Institute, Potsdam, Germany
thomas.blaesius@hpi.de

² University of Minnesota Duluth, Duluth, MN, USA

Abstract. The hardness of formulas at the solubility phase transition of random propositional satisfiability (SAT) has been intensely studied for decades both empirically and theoretically. Solvers based on stochastic local search (SLS) appear to scale very well at the critical threshold, while complete backtracking solvers exhibit exponential scaling. On industrial SAT instances, this phenomenon is inverted: backtracking solvers can tackle large industrial problems, where SLS-based solvers appear to stall. Industrial instances exhibit sharply different structure than uniform random instances. Among many other properties, they are often *heterogeneous* in the sense that some variables appear in many while others appear in only few clauses.

We conjecture that the *heterogeneity* of SAT formulas alone already contributes to the trade-off in performance between SLS solvers and complete backtracking solvers. We empirically determine how the run time of SLS vs. backtracking solvers depends on the heterogeneity of the input, which is controlled by drawing variables according to a scale-free distribution. Our experiments reveal that the efficiency of complete solvers at the phase transition is strongly related to the heterogeneity of the degree distribution. We report results that suggest the depth of satisfying assignments in complete search trees is influenced by the level of heterogeneity as measured by a power-law exponent. We also find that incomplete SLS solvers, which scale well on uniform instances, are not affected by heterogeneity. The main contribution of this paper utilizes the scale-free random 3-SAT model to isolate heterogeneity as an important factor in the scaling discrepancy between complete and SLS solvers at the uniform phase transition found in previous works.

1 Introduction

The worst-case time complexity of propositional satisfiability (SAT) entails that no known algorithm can solve it in polynomial time [12]. Nevertheless, many large industrial SAT instances can be solved efficiently in practice by modern solvers. So far, this discrepancy is not well-understood.

Studying random SAT instances provides a way to explain this discrepancy between theory and practice as it replaces the worst case with the average case.

© The Author(s) 2019

T. Vojnar and L. Zhang (Eds.): TACAS 2019, Part I, LNCS 11427, pp. 117–134, 2019.

https://doi.org/10.1007/978-3-030-17462-0_7

A large amount of both theoretical and experimental research effort focuses almost exclusively on the *uniform random* distribution. Uniform random SAT instances are generated by choosing, for each clause, the variables included in this clause uniformly at random among all variables. Uniform random formulas are easy to construct, and are comparatively more accessible to probabilistic analysis due to their uniformity and the stochastic independence of choices. The analysis of this model can provide valuable insights into the SAT problem in general and has led to the development of tools that are useful also in other areas. However, considering the average-case complexity of solving uniform random formulas cannot explain why SAT solvers work well in practice: in the interesting case that the clause-variable ratio is close to the satisfiability threshold (i.e., the formulas are not trivially satisfiable or trivially unsatisfiable), SAT solvers that perform well on industrial instances struggle to solve the randomly generated formulas fast and algorithms tuned for random formulas perform poorly on industrial instances [14, 27, 33].

The comparative efficiency of existing solvers on real-world SAT instances is somewhat surprising given not only worst-case complexity theoretic results, but also the apparent hardness of uniform random formulas sampled from the critically-constrained regime [8, 31]. Katsirelos and Simon [26] comment that even though the ingredients for building a good SAT solver are mostly known, we still currently cannot explain their strong performance on real-world problems.

This picture is further complicated by the fact that solvers based on stochastic local search (SLS) appear to scale *polynomially* in the critically constrained region of uniform random SAT, whereas complete backtracking solvers scale *exponentially* on these formulas [32]. We are interested in identifying structural aspects of formulas that do not occur in uniform random instances, but can somehow be exploited by solvers.

Industrial SAT instances are complex, and possess many structural characteristics. Among these are *modularity* [4], *heterogeneity* [2], *self-similarity* [1], and *locality* [22]. Modularity measures how well the formula (when modeled as a graph representing the inclusion relation between variables and clauses) can be separated into communities with many internal and few external connections. It is generally assumed that the high modularity of industrial instances is one of the main reasons for the good performance of SAT solvers. Though it is possible to develop models that generate formulas with high modularity [21, 35], there is, however, no established model with this property. Enforcing high modularity can lead to complicated stochastic dependencies, making analysis difficult.

Heterogeneity measures the imbalance in distribution of variables over the clauses of the formula. A highly heterogeneous formula contains only few variables that appear in many clauses and many variables appearing in few clauses. Many industrial instances, particularly from formal verification, exhibit a high heterogeneity. Ansótegui, Bonet, and Levy [3] proposed a number of non-uniform models that produce heterogeneous instances. One such model they introduced was the *scale-free* model. Often, the degree distribution (the *degree* of a variable is the number of clauses containing it as a literal) roughly follows a *power-law* [2], i.e., the number of variables of degree d is proportional to $d^{-\beta}$.

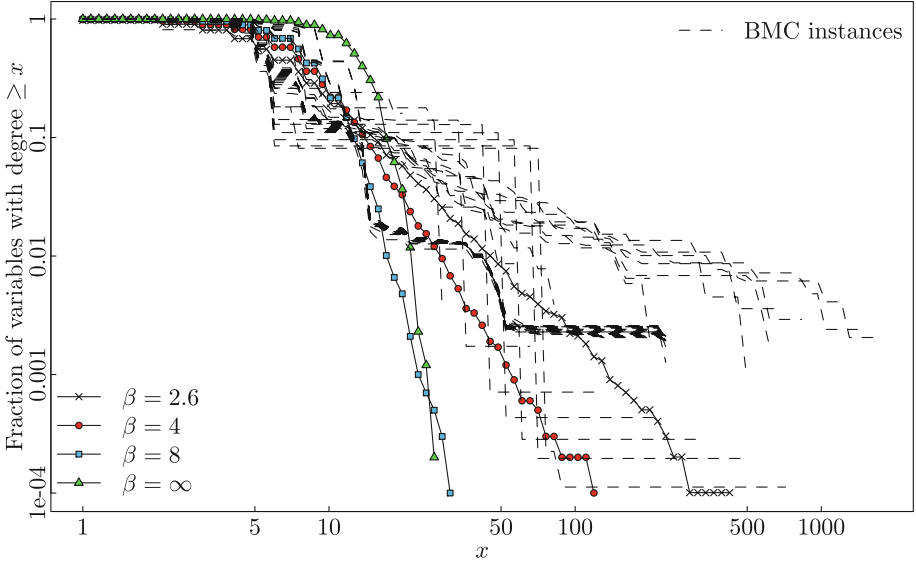


Fig. 1. Empirical cumulative degree distributions on power-law formulas for different β , $n = 10000$, and hardware model checking instances from SAT 2017 competition. Power-law distributions appear as a line on a log-log plot with slope determined by β . As β increases, we observe the power-law distributions converging to uniform ($\beta = \infty$).

Here β is the *power-law exponent*. Figure 1 illustrates a number of empirical cumulative degree distributions for both industrial and synthetic power-law formulas. The industrial formulas come from the SAT-encoded deep bound hardware model checking instance benchmarks submitted to the 2017 SAT competition [7] that had measured empirical power law exponents below 5.

A degree distribution that follows a power law is only a sufficient condition for heterogeneity. Nevertheless, we argue that the scale-free model allows for fine-tuned parametric control over the extent of heterogeneity in the form of the power-law exponent β (see Proposition 1).

No single one of the above properties (e.g., heterogeneity, modularity, locality, etc.) can completely characterize the scaling effects observed on industrial instances vs. uniform random instances. However, a first step toward explaining the performance of SAT solvers in different environments is to isolate these features and determine what kinds of structural properties do influence the run time of different solvers. Our goal in this paper is to empirically determine the impact of heterogeneity (as produced by a scale-free degree distribution) on the run time of SLS-based vs. complete backtracking solvers.¹

Though it seems natural that a more realistic model is better suited to explain the run times observed in practice, it is unclear whether the heterogeneity is

¹ A solver is *complete* if it always finds a solution or a proof that no solution exists. In contrast, SLS-based solvers only terminate if they find a solution.

actually even a relevant factor. It might as well be that other mentioned properties, such as modularity, or self-similarity lead to good run times independent of the degree distribution. The experiments Ansótegui et al. [3] performed on their different models indicate that the heterogeneity in fact helps solvers that also perform well on industrial instances. However, these experiments did not address the phenomenon observed by Mu and Hoos [32] where SLS solvers outperform complete solvers. Our goal is to demonstrate that heterogeneity of the degree distribution has a strong positive impact on the scaling of complete solvers, but not on SLS-based solvers.

To study this impact, we perform large-scale experiments on scale-free random 3-SAT instances with varying power-law exponents β . We note that small power-law exponents lead to heterogeneous degree distributions while increasing β makes the instances more and more homogeneous; see Fig. 1. In fact, for $\beta \rightarrow \infty$, scale-free random 3-SAT converges to the uniform random 3-SAT model. Thus, it can be seen as a generalization of the uniform model with a parameter β that directly adjusts the heterogeneity.

Our experiments clearly show a distinct crossover in performance with respect to a set of complete backtracking (CDCL- and DPLL-based) SAT solvers and a set of SLS-based solvers as we interpolate between highly heterogeneous instances and uniform random instances. Moreover, the performance of SLS-based solvers remain relatively unaffected by the degree distribution. These results might partially explain the effect observed on uniform random instances by Mu and Hoos [32]. In this case, complete backtracking solvers scale poorly on random instances with a homogeneous degree distribution, while SLS-based solvers perform best.

2 Scale-Free Formulas and Heterogeneity

A random 3-SAT formula Φ on n Boolean variables x_1, \dots, x_n and m clauses is a conjunction of m clauses $\Phi := C_1 \wedge C_2 \wedge \dots \wedge C_m$ where $C_i := (\ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3})$ is a disjunction of exactly three literals. A literal is a possibly negated variable. A formula Φ is satisfiable if there exists variable assignment for which Φ evaluates to true.

The canonical distribution for random 3-SAT formulas is the *uniform distribution*, which is the uniform measure taken over all formulas with n variables and m clauses. The uniform 3-SAT distribution is sampled by selecting uniformly m 3-sets of variables and negating each variable with probability 1/2 to form the clauses of the formula. The *scale-free* random 3-SAT distribution is similar, except the degree distribution of the variables is not homogeneous.

In the scale-free model introduced by Ansótegui, Bonet and Levy [3], a formula is constructed by sampling each clause independently at random. In contrast to the classical uniform random model, however, the variable probabilities $p_i := \Pr(X = x_i)$ to choose a variable x_i are non-uniform. In particular, a scale-free formula is generated by using a *power-law* distribution for the variable distribution. To this end, we assign each variable x_i a *weight* w_i and sample it with probability $p_i := \Pr(X = x_i) = \frac{w_i}{\sum_j w_j}$. To achieve a power-law distribution,

we use the following concrete sequence of weights.

$$w_i := \frac{\beta - 2}{\beta - 1} \left(\frac{n}{i}\right)^{\frac{1}{\beta-1}} \quad (1)$$

for $i = 1, 2, \dots, n$, which is a canonical choice for power-law weights, cf. [9]. This sequence guarantees $\sum_j w_j \rightarrow n$ for $n \rightarrow \infty$ and therefore

$$p_i \rightarrow \frac{1}{n} \frac{\beta - 2}{\beta - 1} \left(\frac{n}{i}\right)^{\frac{1}{\beta-1}}. \quad (2)$$

To draw Φ , we generate each clause C_i as follows. (1) Sample three variables independently at random according to the distribution p_i . Repeat until no variables coincide. (2) Negate each of the three variables independently at random with probability $1/2$.

Note that Ansótegui et al. [3] use α instead of β as the power-law exponent and define $\beta := 1/(\alpha - 1)$. We instead follow the notational convention of Chung and Lu, cf. [9].

As already noted in the introduction, the power-law exponent β can be seen as a measure of how heterogeneous the resulting formulas are. This can be formalized as follows.

Proposition 1. *For a fixed number of variables, scale-free random 3-SAT converges to uniform random 3-SAT as $\beta \rightarrow \infty$.*

Proof. First observe that, for any fixed n and $\beta \rightarrow \infty$, the weights w_i as defined in Eq. (1) converge to 1. When generating a scale-free random 3-SAT instance, variables are chosen to be included in a clause with probability proportional to w_i . Thus, for $\beta \rightarrow \infty$, each variable is chosen with the same probability $1/n$ as it is the case for uniform random 3-SAT. \square

We note that the model converges rather quickly: The difference between the weights is maximized for w_1 and w_n (with w_1 being the largest and w_n being the smallest). By choosing $\beta = c \log n$, the maximum weight difference $w_1 - w_n$ converges to the small constant $e^{1/c} - 1$ for growing n . Thus, when choosing $\beta \in \omega(\log n)$ (i.e., β grows asymptotically faster than $\log n$), this difference actually goes to 0 for $n \rightarrow \infty$, leading to the uniform model. This quick convergence can also be observed in Fig. 1 where the difference between $\beta = 8$ and the uniform case ($\beta = \infty$) is rather small.

2.1 The Solubility Phase Transition

The *constraint density* of a distribution of formulas on n variables and m clauses is measured as the ratio of clauses to variables m/n . A *phase transition* in a random satisfiability model is the phenomenon of a sharp transition as a function of constraint density between formulas that are almost surely satisfiable and formulas that are almost surely not satisfiable. The location of such a transition is called the *critical density* or *threshold*.

Threshold phenomena in the uniform random model have been studied for decades. The *satisfiability threshold conjecture* maintains that if Φ is a formula drawn uniformly at random from the set of all k -CNF formulas with n variables and m clauses, there exists a real number r_k such that

$$\lim_{n \rightarrow \infty} \Pr\{\Phi \text{ is satisfiable}\} = \begin{cases} 1 & m/n < r_k; \\ 0 & m/n > r_k. \end{cases}$$

This transition is sharp [18] in the sense that the probability of satisfiability as a function of constraint density m/n approaches a unit step function as $n \rightarrow \infty$. For $k = 2$, the location of the transition is known exactly to be $r_2 = 1$ [10]. For $k \geq 3$, bounds asymptotic in k [11] and exact results for large constant k [16] are now known.

The phenomenon of a sharp solubility transition is also interesting from the perspective of computational complexity and algorithm engineering, since it appears to coincide with a regime of formulas that are particularly difficult to solve by complete SAT solvers [31].

In the scale-free model, the location of the critical threshold $r_k(\beta)$ is a function of power-law exponent β . In the case of $k = 2$, it was proved that the critical density is bounded as $r_2(\beta) \leq \frac{(\beta-1)(\beta-3)}{(\beta-2)^2}$ [19]. Recently, Levy [28] proved this bound for $k = 2$ is tight. Similar to the uniform model, the critical density $r_k(\beta)$ for $k > 2$ seems to be more elusive.

2.2 Characterizing the Scale-Free Phase Transition

Ansótegui et al. [3] empirically located the phase transition of the scale-free 3-SAT model and noted that the critical density for very low β was small, and the threshold approaches the critical point of the uniform model at ≈ 4.26 as $\beta \rightarrow \infty$.² They report the critical threshold values as a function of β by testing 200 formulas at each value of β in the set $\{2, 7/3, 3, 5, \infty\}$.

Nevertheless, a number of details about the nature of the scale-free phase transition is still lacking from this picture. First, the sharpness of the phase transition as β evolves is not immediately clear. Furthermore, even though most previous work assumes the hardest instances are located at the phase transition region [3, Section 4], it is not obvious what the shape and extent of an easy-hard-easy transition (if it even exists) would be for scale-free formulas, nor is it known how the effect is influenced by β . Finally, previous works have so far not addressed the curious phenomenon of SLS solvers and complete solvers that seem to scale so differently on uniform random and industrial problems. We tackle this issue in the next section.

To better characterize the phase transition in the scale-free model, we generated formulas as follows. For any particular n and β , taking a sequence of 300 equidistant values $\alpha_i \in [2, 5]$ for $i \in \{1, \dots, 300\}$, we sample 100 formulas from

² We are translating the term they refer to as β to the term we refer to as β , as mentioned above.

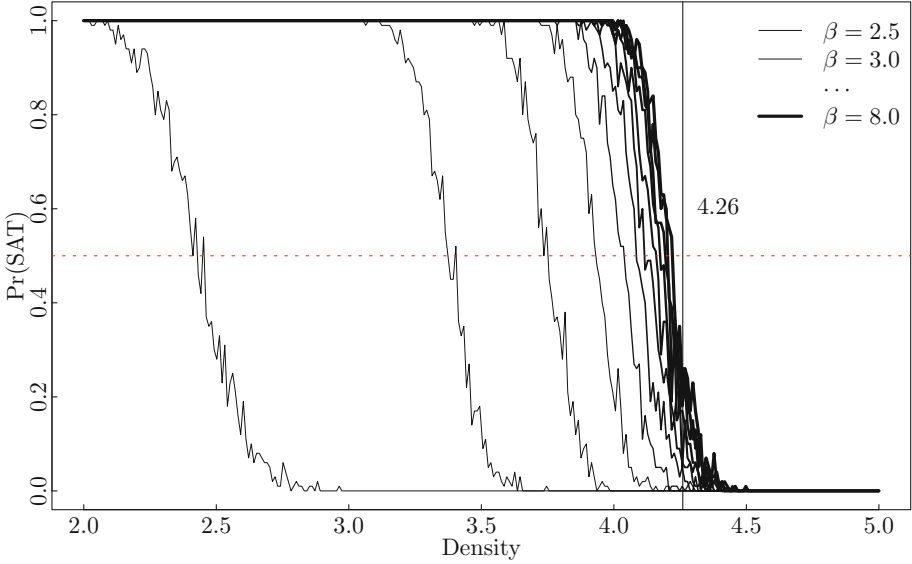


Fig. 2. Proportion of satisfiable formulas ($n = 500$) as a function of constraint density m/n for various power-law exponents β . Threshold approaches the critical density of the uniform random 3-SAT model: $r_3 \approx 4.26$.

the scale-free distribution with n variables, power-law exponent β , and density α_i (i.e., $m = \alpha_i n$). In other words, each value of n and β corresponds to 30000 formulas at various densities.

To estimate the location of the critical threshold, we decide the satisfiability of each formula with the DPLL-based solver `march_hi` [25]. We then find the density α_i yielding half the formulas satisfiable. This is the approach for locating the threshold in the uniform model [13]. However, in the case of the random scale-free model, the threshold depends on β .

Using this approach, we generated sets of formulas at the phase transition varying both n and β . For each $n \in \{500, 600, 700\}$, we generated formulas with $\beta \in \{2.5, 2.6, \dots, 5.0\}$. We find the run times of the complete solvers exhibit a strong positive correlation with β . This is consistent with complete solvers performing poorly on uniform random ($\beta = \infty$) problems of even modest sizes, but it unfortunately restricts us to more modest formula sizes. To determine the effect of very large power-law exponents, we also generated formulas for $n = 500$, $\beta \in \{5, 6, 7, 8\}$.

The sharpness of the transition appears to evolve with β . Figure 2 reports the proportion of satisfiable formulas as a function of constraint density with $n = 500$. As $\beta \rightarrow \infty$, the solubility transition shifts toward the supposed critical density of the uniform random 3-SAT model, i.e., $r_3 \approx 4.26$. This is consistent with previous work on this model, but we also can see from this that the transition becomes steeper with increasing β , despite the fact that n is held constant.

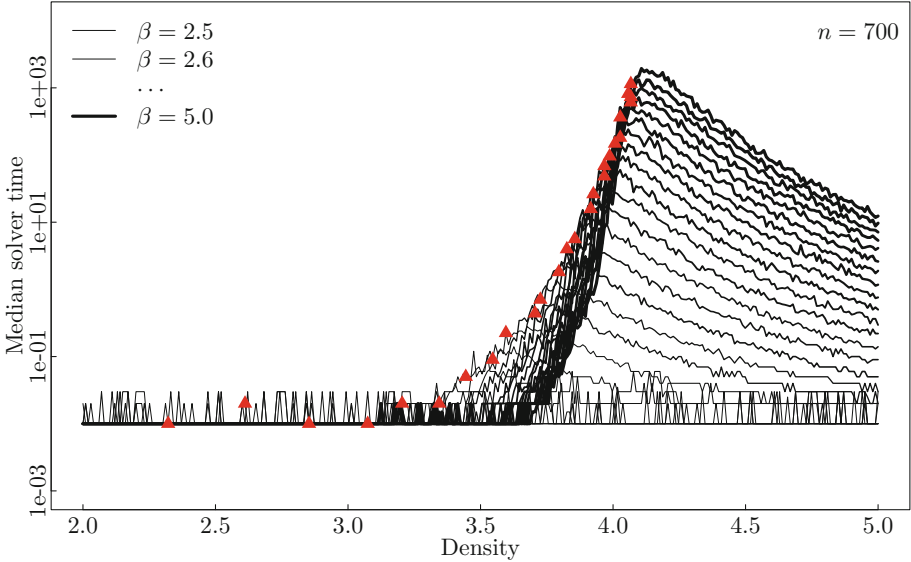


Fig. 3. Median solver times for `march_hi` on formulas of size $n = 700$ as β evolves toward the uniform distribution. Red triangles (\blacktriangle) mark the exact density at which empirical threshold was determined. (Color figure online)

On the uniform random model, the hardest formulas for complete backtracking solvers lie at the solubility phase transition because they produce deeper search trees [31]. We also observe this so-called easy-hard-easy pattern for the scale free model in the trends for median time for `march_hi` to solve formulas of size $n = 700$ in Fig. 3. Moreover, the power-law exponent β is strongly correlated with the height of the hardness peak and we conjecture that the complexity of the resulting search tree is strongly coupled to the heterogeneity of the degree distribution. The empirically determined critical point, indicated in the figure with a red triangle (\blacktriangle) tightly corresponds with the hardness peaks.

3 Scaling Across Solver Types

Our main goal is to understand the influence of the heterogeneity of the degree distribution at the phase transition on SLS-based solvers and complete backtracking solvers. The original paper by Mu and Hoos [32] investigated three DPLL-based SAT solvers: `kcdfs` [15], `march_hi` [25], and `march_br` [24]; and three SLS-based solvers: `WalkSAT/SKC` [34], `BalancedZ` [29], and `ProbSAT` [5]. They found the three DPLL-based solvers scaled exponentially at the uniform critical threshold and the SLS-based solvers did not. To investigate the role of heterogeneity in this context, we used the same solvers as the original Mu and Hoos paper.

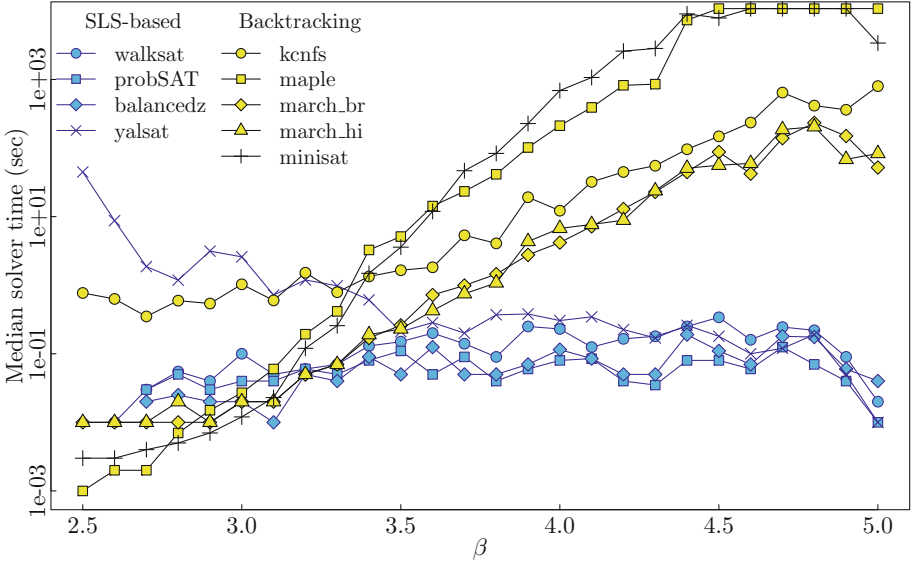


Fig. 4. Median solver time at the phase transition for all solvers on formulas with $n = 700$. Top of plot corresponds to one hour cutoff.

In addition to the DPLL solvers, we tested MiniSAT [17], and the MiniSAT-based CDCL solver MapleCOMSPS [30]. These choices are motivated by a number of reasons. First, MiniSAT has performed well in industrial categories in previous SAT competitions, as has MapleCOMSPS on the Applications Benchmark at the SAT 2016 competition and the Main Track and No Limit Track at the SAT 2017 competition. Second, we want to know if architectural decisions such as branching heuristic, look-ahead, backtracking strategy, or clause learning has any effect. We also supplemented the SLS-based solvers with YalSAT [6], which won first place in the Random Track of the SAT 2017 competition.

SLS-based solvers are *incomplete* in the sense that they can only decide satisfiability. Studies that involve such solvers need to be constrained to satisfiable formulas [20, 32, 36]. We use the standard rejection sampling approach to filtering for satisfiable formulas. For each n and β value, we filtered out the unsatisfiable formulas at the phase transition located as described above. For each of these formulas, we ran each of the above SLS-based and complete solvers to compare the required CPU time until a solution was determined. We imposed a solver cutoff time of one hour. In Fig. 4, we chart the median solution time on formulas of size $n = 700$ at the phase transition as a function of power law exponent β . For low β (highly heterogeneous) formulas, the complete solvers outpace the SLS-based solvers (though solution time for both is fast). We observe a crossover point around $\beta = 3.5$ where the required time for complete solvers begins to dominate the median time for the SLS techniques.

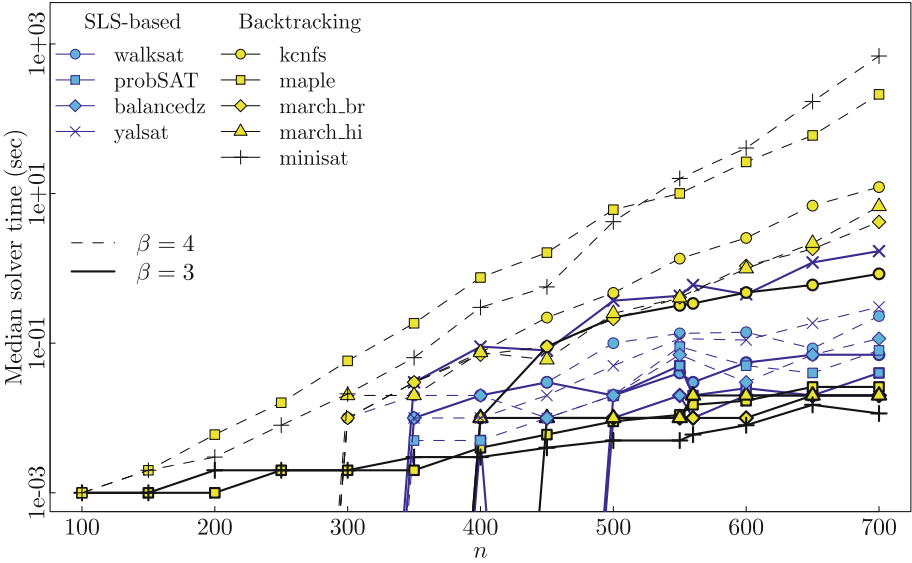


Fig. 5. Median solver time at the phase transition for all solvers on formulas with $\beta = 3$ (—) and $\beta = 4$ (- -).

Figure 4 reveals that the CDCL-based solvers MiniSAT and MapleCOMSPS had worse performance degradation with decreasing heterogeneity than the DPLL-based solvers. Moreover, YalSAT seems to perform worse on highly heterogeneous formulas. This is interesting behavior, as YalSAT implements variants of ProbSAT with a Luby restart strategy and is conjectured to be identical to ProbSAT on uniform random instances [6]. Our results confirm that YalSAT and ProbSAT are indistinguishable as β increases, but we have evidence that the restart schedule might somehow affect performance on heterogeneous formulas.

To better quantify the influence of scaling with n , we consider the median solver time as a function of n for two representative values of β (3 and 4) in Fig. 5. To obtain a clearer picture, we repeated the formula generation process for smaller formulas ($n = 100, 200, \dots$). The exponential scaling in complete solvers with large β (dashed lines) is easy to observe in this plot. On smaller β (solid lines), the complete solvers scale more efficiently.

To take variance into account, we compare solver performance in Fig. 6.³ Here we have removed the CDCL-based solvers and YalSAT for clarity. This is therefore the solver set originally investigated by Mu and Hoos [32]. We again can identify a distinct crossover point at $\beta = 3.5$. Figure 7 repeats the results for $n = 500$. For this size of formula, the small β regime is extremely easy, and the results are somewhat obscured. However, these formulas are small enough that

³ In all box blots, the boxes show the interquartile range, the bold line is the median, and the whiskers extend to $3/2 \cdot \text{IQR}$ below (respectively, above) the 25th (respectively, the 75th) percentile. All points beyond this are outliers.

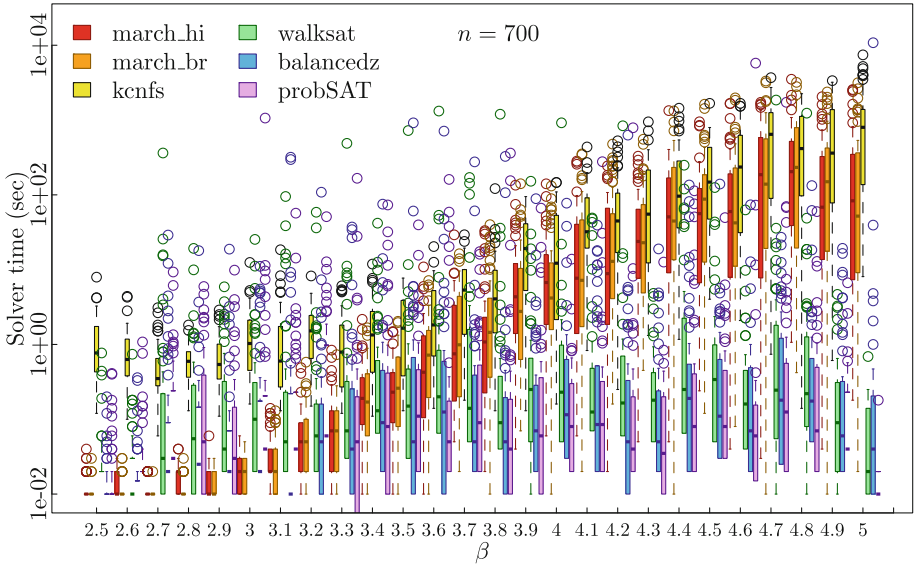


Fig. 6. CPU time to solve formulas at the scale-free phase transition with $n = 700$.

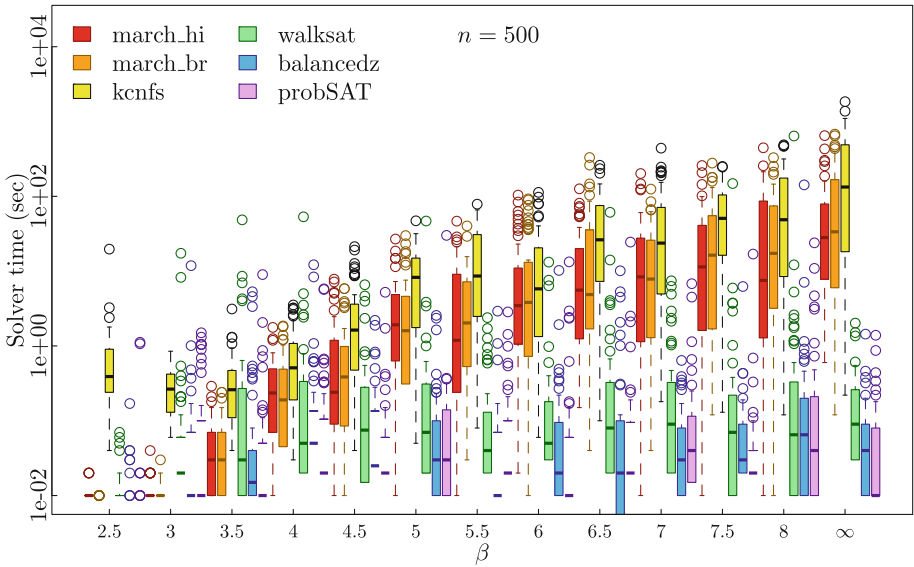


Fig. 7. CPU time to solve formulas at the scale-free phase transition with $n = 500$. Rightmost group ($\beta = \infty$) denotes satisfiable uniform random formulas with $n = 500$ and $m = 2131$.

we are able to consistently solve high β formulas, and we report the results up to $\beta = 8$. The rightmost group of the figure represents filtered uniform random formulas with $n = 500$ and $m = 2131$. To estimate the correct critical density for the uniform formulas, we used the parametric model from [32, Eq. (3)].

4 Effect of Heterogeneity on Search Trees

The discrepancy in solver efficiency across different levels of heterogeneity observed in the preceding section suggests that the degree distribution strongly affects the decisions of a wide range of solvers in a systematic way. We hypothesize that, for fixed n and m , heterogeneous 3-SAT formulas have more satisfying assignments on average, and these assignments tend to be more quickly reachable, because partial assignments tend to produce more implied literals.

Even for small formulas, it is infeasible to enumerate all satisfying assignments. Propositional model counting offers a technique in the form of an exhaustive extension to DPLL in which the branchpoint after a satisfying assignment is followed [23]. When a branch corresponding to a partial assignment of t fixed variables is marked as satisfiable at depth d , it is already possible to conclude that there are 2^{n-t} solutions at depth at least d . Using this technique, we can count the satisfying assignments to formulas generated by the scale-free model to measure the influence of heterogeneity on the solution count. The left panel of Fig. 8 reports the number of satisfying assignments found in satisfiable scale-free formulas generated at each β value.

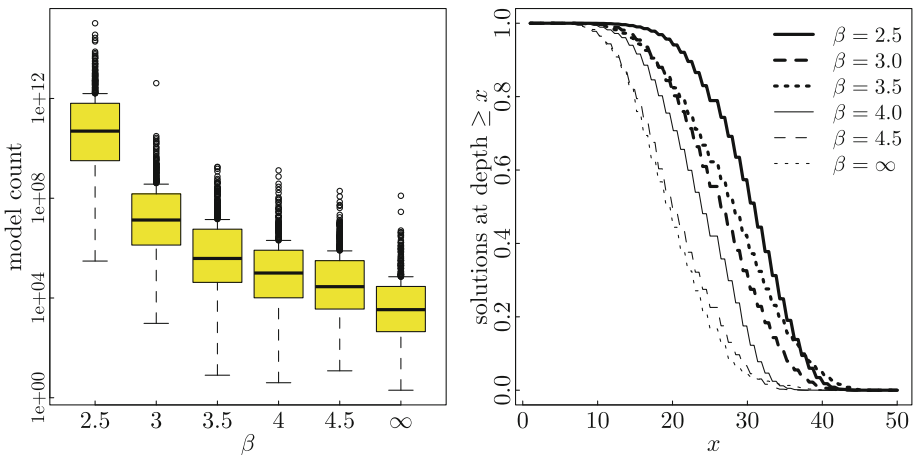


Fig. 8. Left: count of satisfying assignments on satisfiable scale-free formulas at each power-law exponent at the phase transition. For each $\beta \in \{2.5, 3.0, \dots, 4.5, \infty\}$, we filter from 1000 generated random formulas at the phase transition with $n = 100$. Right: empirical degree distributions reporting the proportion of solutions at depth equal or greater than x aggregated over all formulas at each β value.

To obtain a more detailed picture, we plot the empirical cumulative distribution functions of solution depth on the right panel of Fig. 8. The curves represent the proportion $P(x)$ of solutions at a depth equal or greater than x .

As mentioned above, shallower solutions arise from the abundance of implied literals. We find that highly heterogeneous formulas tend to have many more constraint propagations leading to either satisfying assignments or contradictions. We recorded this data and display it in Fig. 9.

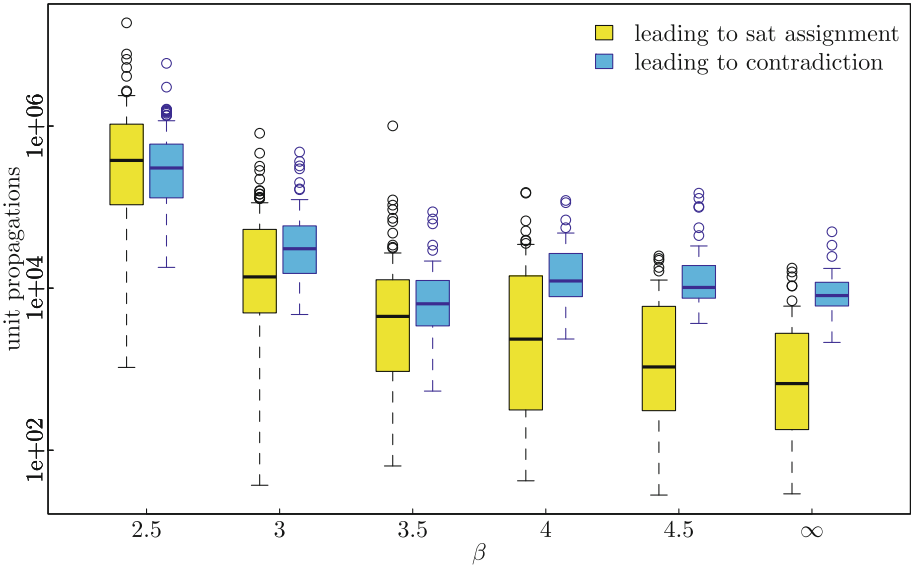


Fig. 9. Number of constraint propagations leading to a satisfying assignment or contradiction as a function of β .

4.1 Relaxed Bounded Model Checking Instances

Our aim has been to isolate heterogeneity as an impacting factor in the relative performance discrepancy between SLS solvers and complete solvers. Nevertheless, we also conjecture that the behavior of complete solvers on synthetic scale-free formulas is comparable to their behavior on certain industrial instances with power-law degree distributions. To investigate this, we compare the complexity of the remaining formula after selecting a branching literal during runs of march_hi, which was the highest performing backtracking solver for our data. Each solid line in Fig. 10 displays the average remaining formula complexity measured as clauses not yet satisfied as a percentage of original formula size. These run average are taken over all satisfiable powerlaw formulas of $n = 500$ for different values of β . Note that some early averages exceed 100%, which likely occurs because march_hi also adds binary resolvents during the solving process. Moreover, the complexity may increase during the run, as complete solvers utilize backtracking.

We compare this with the well-known BMC DIMACS benchmark set from CMU⁴. Our motivation for this choice was to utilize a widely available set of bounded model checking formulas of reasonable size. To provide a fair comparison to the filtered scale-free formulas, we “relaxed” each BMC formula by iteratively removing a single clause at random until the first time it was satisfiable. This preserves the statistical characteristics of the degree distribution while producing a satisfiable formula.

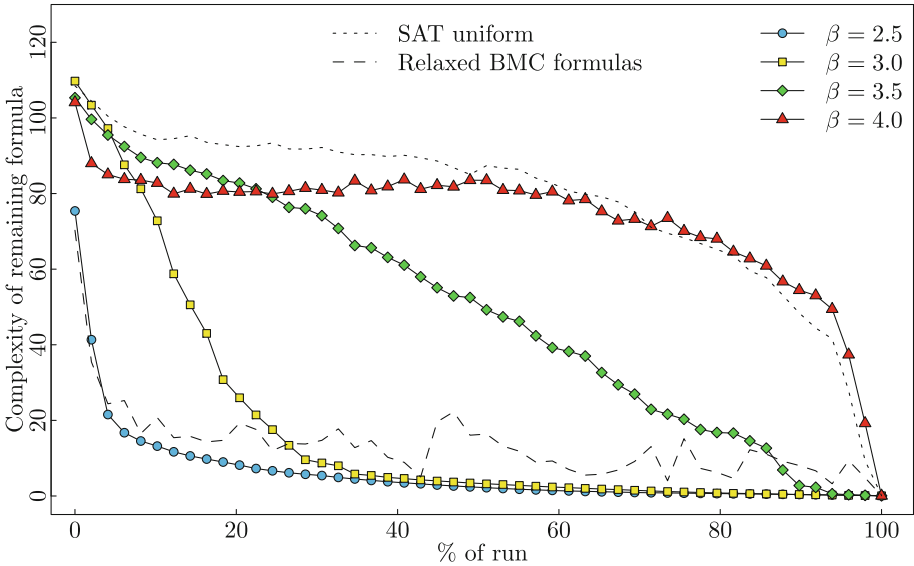


Fig. 10. Average complexity of the remaining formula depending on the percentage of already made branching decisions in a run. Reported are runs of `march_hi` on scale-free formulas of varying β , and satisfiable real-world model checking formulas: uniform (\cdots), industrial ($--$), and synthetic ($-$).

The profile of the industrial bounded model checking formulas closely matches the solver behavior on the heterogeneous scale-free formulas, whereas the solver behavior on the uniform suite corresponds to the behavior on more homogeneous formulas. We conjecture that the power-law degree distribution in the bounded model checking formulas (cf. Fig. 1) affect the search tree in similar ways.

5 Conclusions

We have investigated a parameterized distribution over propositional formulas that allows us to carefully control the heterogeneity (via the power-law exponent) to interpolate smoothly between highly heterogeneous random formulas

⁴ <http://fmv.jku.at/bmc/>.

and nearly-uniform random formulas. This allows us to observe the exact influence of this kind of heterogeneity on hardness for two different solver classes. Our empirical analysis uncovers an interesting crossover effect in the performance of SLS-based vs. complete backtracking solvers, depending on the heterogeneity.

We summarize our main findings as follows. (1) Complete solvers tuned for industrial instances perform significantly better on heterogeneous formulas than uniform formulas. This is likely due to the fact that the search space of highly heterogeneous formulas have more solutions, and solutions tend to be shallower in the decision tree. (2) Incomplete SLS-based solvers, which are typically recognized for their performance on uniform random k -SAT, do not benefit much (if at all) from heterogeneity. (3) Random instances (even heterogeneous ones) close to the satisfiability threshold are harder to solve than many industrial instances.

The first two insights are a step towards understanding the disaccoring runtime behaviors of different solvers on industrial and random instances. Moreover, these findings suggest that the behavior of SLS-based solvers are relatively heterogeneity-invariant, whereas complete solvers are far more sensitive to the homogeneous degree distributions of uniform random formulas. This may explain, at least in part, the exponential scaling of complete solvers at the uniform phase transition observed by Mu and Hoos [32].

On the other hand, the third insight shows that heterogeneity alone cannot explain why industrial instances can be solved so fast in practice. On the upside, this means that random scale-free formulas chosen close to the satisfiability threshold can serve as hard benchmark instances. It would be interesting to see whether incomplete solvers can be tuned to catch up to (or even outperform) complete solvers on heterogeneous instances. Due to the similarities between heterogeneous random formulas and industrial instances with respect to runtime behavior, we believe that tuning solvers for heterogeneous random formulas can actually lead to techniques that also help solving industrial instances faster.

The behavior of solvers on uniform random formulas is well-studied. However, there is no obvious reason to believe that solver performance near the scale-free phase transition is identical to performance near the uniform phase transition. Our work suggests that there is some kind of structure in formulas with heavy-tailed degree distributions that is being exploited by complete solvers. It is important to stress that a scale-free degree distribution alone is not enough to characterize the complex structure of real problems. Our results provide context by isolating heterogeneity (as realized by the power-law exponent) as an important feature impacting the performance of state-of-the-art CDCL- and DPLL-based SAT solvers. Other non-uniform models exist, and a future avenue of work is to investigate such models, especially the Popularity-Similarity model recently introduced by Giráldez-Cru and Levy [22], which can generate formulas with both a scale-free degree distribution and high modularity.

References

1. Ansótegui, C., Bonet, M.L., Giráldez-Cru, J., Levy, J.: The fractal dimension of SAT formulas. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) IJCAR 2014. LNCS (LNAI), vol. 8562, pp. 107–121. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08587-6_8
2. Ansótegui, C., Bonet, M.L., Levy, J.: On the structure of industrial SAT instances. In: Gent, I.P. (ed.) CP 2009. LNCS, vol. 5732, pp. 127–141. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04244-7_13
3. Ansótegui, C., Bonet, M.L., Levy, J.: Towards industrial-like random SAT instances. In: Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI), pp. 387–392 (2009)
4. Ansótegui, C., Giráldez-Cru, J., Levy, J.: The community structure of SAT formulas. In: Cimatti, A., Sebastiani, R. (eds.) SAT 2012. LNCS, vol. 7317, pp. 410–423. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31612-8_31
5. Balint, A., Schönig, U.: probSAT and pprobSAT. In: Proceedings of the 2014 SAT Competition, p. 63 (2014)
6. Biere, A.: CaDiCaL, Lingeling, Plingeling, Treengeling and YalSAT. In: Proceedings of SAT Competition 2017, pp. 14–15 (2017)
7. Biere, A.: Deep bound hardware model checking instances, quadratic propagations benchmarks and reencoded factorization problems submitted to the SAT competition 2017. In: Proceedings of SAT Competition 2017, pp. 40–41 (2017)
8. Cheeseman, P., Kanefsky, B., Taylor, W.M.: Where the really hard problems are. In: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI), pp. 331–340 (1991)
9. Chung, F., Lu, L.: The average distances in random graphs with given expected degrees. PNAS **99**(25), 15879–15882 (2002)
10. Chvátal, V., Reed, B.: Mick gets some (the odds are on his side). In: Proceedings of the Thirty-Third IEEE Annual Symposium on Foundations of Computer Science (FOCS), pp. 620–627, October 1992
11. Coja-Oghlan, A.: The asymptotic k -SAT threshold. In: Proceedings of the Forty-Sixth Annual Symposium on Theory of Computing (STOC), pp. 804–813 (2014)
12. Cook, S.A.: The complexity of theorem-proving procedures. In: Proceedings of the Third Annual Symposium on Theory of Computing (STOC), pp. 151–158 (1971)
13. Crawford, J.M., Auton, L.D.: Experimental results on the crossover point in random 3-SAT. Artif. Intell. **81**(1–2), 31–57 (1996)
14. Crawford, J.M., Baker, A.B.: Experimental results on the application of satisfiability algorithms to scheduling problems. In: Proceedings of the Twelfth AAAI Conference on Artificial Intelligence, pp. 1092–1097 (1994)
15. Dequen, G., Dubois, O.: *kcnfs*: an efficient solver for random k -SAT formulae. In: Giunchiglia, E., Tacchella, A. (eds.) SAT 2003. LNCS, vol. 2919, pp. 486–501. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24605-3_36
16. Ding, J., Sly, A., Sun, N.: Proof of the satisfiability conjecture for large k . In: Proceedings of the Forty-Seventh Annual Symposium on Theory of Computing (STOC), pp. 59–68 (2015)
17. Eén, N., Sörensson, N.: An extensible SAT-solver. In: Giunchiglia, E., Tacchella, A. (eds.) SAT 2003. LNCS, vol. 2919, pp. 502–518. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24605-3_37
18. Friedgut, E.: Sharp thresholds of graph properties, and the k -SAT problem. J. Am. Math. Soc. **12**(4), 1017–1054 (1999)

19. Friedrich, T., Krohmer, A., Rothenberger, R., Sutton, A.M.: Phase transitions for scale-free SAT formulas. In: Proceedings of the Twenty-First AAAI Conference on Artificial Intelligence, pp. 3893–3899 (2017)
20. Gent, I.P., Walsh, T.: Towards an understanding of hill-climbing procedures for SAT. In: Proceedings of the Eleventh AAAI Conference on Artificial Intelligence, pp. 28–33 (1993)
21. Giráldez-Cru, J., Levy, J.: A modularity-based random SAT instances generator. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), pp. 1952–1958 (2015)
22. Giráldez-Cru, J., Levy, J.: Locality in random SAT instances. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI), pp. 638–644 (2017)
23. Gomes, C.P., Sabharwal, A., Selman, B.: Model counting. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability. Frontiers in Artificial Intelligence and Applications, vol. 185, pp. 633–654. IOS Press, Amsterdam (2009)
24. Heule, M.J.H.: march_br. In: Proceedings of SAT Competition 2013 (2013)
25. Heule, M., van Maaren, H.: march_hi. In: Proceedings of the 2009 SAT Competition, pp. 27–28 (2009)
26. Katsirelos, G., Simon, L.: Eigenvector centrality in industrial SAT instances. In: Milano, M. (ed.) CP 2012. LNCS, pp. 348–356. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33558-7_27
27. Konolige, K.: Easy to be hard: difficult problems for greedy algorithms. In: Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR), pp. 374–378 (1994)
28. Levy, J.: Percolation and phase transition in SAT. ArXiv e-prints, July 2017. [arXiv:1708.06805](https://arxiv.org/abs/1708.06805)
29. Li, C., Huang, C., Xu, R.: Balance between intensification and diversification: a unity of opposites. In: Proceedings of SAT Competition 2014, pp. 10–11 (2014)
30. Liang, J.H., Oh, C., Ganesh, V., Czarnecki, K., Poupard, P.: MapleCOMSPS, MapleCOMSPS.lrb, MapleCOMSPS.CHB. In: Proceedings of the 2016 SAT Competition, pp. 52–53 (2016)
31. Mitchell, D.G., Selman, B., Levesque, H.J.: Hard and easy distributions of SAT problems. In: Proceedings of the Tenth AAAI Conference on Artificial Intelligence, pp. 459–465 (1992)
32. Mu, Z., Hoos, H.H.: On the empirical time complexity of random 3-SAT at the phase transition. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), pp. 367–373 (2015)
33. Rish, I., Dechter, R.: Resolution versus search: two strategies for SAT. *J. Autom. Reasoning* **24**(1–2), 225–275 (2000)
34. Selman, B., Kautz, H.A., Cohen, B.: Noise strategies for improving local search. In: Proceedings of the Twelfth AAAI Conference on Artificial Intelligence, pp. 337–343 (1994)
35. Slater, A.: Modelling more realistic SAT problems. In: McKay, B., Slaney, J. (eds.) AI 2002. LNCS (LNAI), vol. 2557, pp. 591–602. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36187-1_52
36. Yokoo, M.: Why adding more constraints makes a problem easier for hill-climbing algorithms: analyzing landscapes of CSPs. In: Smolka, G. (ed.) CP 1997. LNCS, vol. 1330, pp. 356–370. Springer, Heidelberg (1997). <https://doi.org/10.1007/BFb0017451>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

