Handbook of Satisfiability
Armin Biere, Marijn Heule, Hans van Maaren and Toby Walsch
IOS Press, 2008

Chapter 8

Random Satisfiability

Dimitris Achlioptas

8.1. Introduction

Satisfiability has received a great deal of study as the canonical NP-complete problem. In the last twenty years a significant amount of this effort has been devoted to the study of randomly generated satisfiability instances and the performance of different algorithms on them. Historically, the motivation for studying random instances has been the desire to understand the hardness of "typical" instances. In fact, some early results suggested that deciding satisfiability is "easy on average". Unfortunately, while "easy" is easy to interpret, "on average" is not.

One of the earliest and most often quoted results for satisfiability being easy on average is due to Goldberg [Gol79]. In [FP83], though, Franco and Paull pointed out that the distribution of instances used in the analysis of [Gol79] is so greatly dominated by "very satisfiable" formulas that if one tries truth assignments completely at random, the expected number of trials until finding a satisfying one is $O(1)$. Alternatively, Franco and Paull pioneered the analysis of random instances of k-SAT, i.e., asking the satisfiability question for random k-CNF formulas (defined precisely below). Among other things, they showed [FP83] that for all $k \geq 3$ the DPLL algorithm needs an *exponential* number of steps to report all cylinders of solutions of such a formula, or that no solutions exist.

Random k-CNF formulas. *Let $F_k(n, m)$ denote a Boolean formula in Conjunctive Normal Form with m clauses over n variables, where the clauses are chosen uniformly, independently and without replacement among all $2^k\binom{n}{k}$ non-trivial clauses of length k, i.e., clauses with k distinct, non-complementary literals.*

Typically, k is a (small) fixed integer, while n is allowed to grow. In particular, we will say that a sequence of random events $\mathcal{E}_n$ occurs with high probability (w.h.p.) if $\lim_{n\to\infty} \Pr[\mathcal{E}_n] = 1$ and with uniformly positive probability (w.u.p.p.) if $\liminf_{n\to\infty} \Pr[\mathcal{E}_n] > 0$. One of the first facts to be established [FP83] for random k-CNF formulas is that $F_k(n, rn)$ is w.h.p. unsatisfiable for $r \geq 2^k \ln 2$. A few years later, Chao and Franco [CF86, CF90] showed that, in contrast, if $r < 2^k/k$, a very simple algorithm will find a satisfying truth assignment w.u.p.p. Combined, these two facts established $m = \Theta(n)$ as the most interesting regime for considering the satisfiability of random k-CNF formulas.

The study of random k-CNF formulas took off in the late 1980s. In a celebrated result, Chvátal and Szemerédi [CS88] proved that random k-CNF formulas w.h.p. have exponential resolution complexity, implying that if F is a random k-CNF formula with $r \geq 2^k \ln 2$, then w.h.p. *every* DPLL-type algorithm needs exponential time to prove its unsatisfiability. A few years later, Chvátal and Reed [CR92] proved that random k-CNF formulas are satisfiable w.h.p. for $r = O(2^k/k)$, strengthening the w.u.p.p. result of [CF90]. Arguably, the biggest boost came from the experimental work of Mitchell, Selman and Levesque [MSL92] and the analysis of these experiments by Kirkpatrick and Selman in [KS94].

In particular, [SML96] gave extensive experimental evidence suggesting that for $k \geq 3$, there is a range of the clauses-to-variables ratio, r, within which it seems hard even to *decide* if a randomly chosen k-SAT instance is satisfiable or not (as opposed to reporting all cylinders of solutions or that no solutions exist). The analysis of these experiments using finite-size scaling methods of statistical physics in [KS94] showed that this peak in experimental decision complexity coincided with a precipitous drop in the probability that a random formula is satisfiable. For example, for $k = 3$ the analysis drew the followng remarkable picture: for $r < 4$, a satisfying truth assignment can be easily found for almost all formulas; for $r > 4.5$, almost all formulas are unsatisfiable; for $r \approx 4.2$, a satisfying truth assignment can be found for roughly half the formulas and around this point the computational effort for finding a satisfying truth assignment, whenever one exists, is maximized.

This potential connection between phase transitions and computational complexity generated a lot of excitement. A particularly crisp question is whether the probability of satisfiability indeed exhibits a sharp threshold around some critical density (ratio of clauses to variables).

Satisfiability Threshold Conjecture. *For every $k \geq 3$, there exists a constant $r_k > 0$ such that,*

$$\lim_{n\to\infty} \Pr[F_k(n, rn) \text{ is satisfiable}] = \begin{cases} 1, & \text{if } r < r_k \\ 0, & \text{if } r > r_k. \end{cases}$$

By now, the Satisfiability Threshold Conjecture has attracted attention in computer science, mathematics, and statistical physics. It has stimulated numerous interesting results and is at the center of an area of intense research activity, some of which we will survey in the following sections. As we will see, recent results have both strengthened the connection between computational complexity and phase transitions and shown that it is far more nuanced than originally thought. In particular, to the extent that finding satisfying assignments in random formulas is hard, this hardness comes from phase transitions in the solution-space geometry of the formulas, not in their probability of satisfiability. But this is getting ahead of our story.

To conclude this introduction we note that while a number of other generative models have been proposed for random SAT instances over the years (see [Fra01] for an excellent survey) random k-SAT is by far the dominant model. One reason is that random k-CNF formulas enjoy a number of intriguing mathematical prop-

erties. Another is that random k-SAT instances remain computationally hard for a certain range of densities, making them a popular benchmark for testing and tuning satisfiability algorithms. In fact, some of the better practical ideas in use today come from insights gained by studying the performance of algorithms on random k-SAT instances [GSCK00].

The rest of this chapter is divided into two major pieces. The first piece is an overview of the current state of the art regarding mathematical properties of random k-CNF formulas where the number of clauses $m = \Theta(n)$. The second piece concerns the performance of different algorithms on sparse random k-CNF formulas. Both pieces are concerned with (and limited to) *rigorous mathematical results*. For experimental algorithmic results we refer the reader to the survey by Cook and Mitchell [CM97]. For results using the methods of statistical physics we refer the reader to Part 1, Chapter 9.

8.2. The State of the Art

8.2.1. The Satisfiability Threshold Conjecture

The satisfiability threshold conjecture remains open. A big step was made by Friedgut [Fri99] who showed the existence of a sharp threshold for satisfiability around a critical *sequence* of densities (rather than a single density r_k).

Theorem 1 ([Fri99])**.** *For every $k \geq 3$, there exists a sequence $r_k(n)$ such that for any $\epsilon > 0$,*

$$\lim_{n\to\infty} \Pr[F_k(n,m) \text{ is satisfiable}] = \begin{cases} 1, & \text{if } m = (r_k(n) - \epsilon)n \\ 0, & \text{if } m = (r_k(n) + \epsilon)n. \end{cases}$$

While it is widely believed that $r_k(n) \to r_k$, a proof remains elusive. A useful corollary of Friedgut's theorem is the following, as it allows one to give lower bounds for r_k by only establishing satisfiability w.u.p.p. rather than w.h.p.

Corollary 2. *If $F_k(n, r^*n)$ is satisfiable w.u.p.p., then for all $r < r^*$, $F_k(n, rn)$ is satisfiable w.h.p.*

Although the existence of r_k has not been established, we will allow ourselves the notational convenience of writing $r_k \leq r^*$ to denote that for $r > r^*$, $F_k(n, rn)$ is w.h.p. unsatisfiable (and analogously for $r_k \geq r^*$). In this notation, the current best bounds for the location of the satisfiability threshold are, roughly,

$$2^k \ln 2 - \Theta(k) \leq r_k \leq 2^k \ln 2 - \Theta(1) \ .$$

The precise form of these bounds is given by Theorem 3, which we illustrate for some small values of k in Table 8.1.

Theorem 3. *There exists sequences $\delta_k, \epsilon_k \to 0$ such that for all $k \geq 3$,*

$$2^k \ln 2 - (k+1)\frac{\ln 2}{2} - 1 - \delta_k \leq r_k \leq 2^k \ln 2 - \frac{1 + \ln 2}{2} + \epsilon_k \ .$$

It is very easy to show that $r_k \leq 2^k \ln 2$. This is because the probability that $F_k(n,m)$ is satisfied by at least one assignment is trivially bounded by 2^n times the probability that it is satisfied by any particular assignment, which, in turn, is bounded by $(1-2^{-k})^m$ (indeed, this last expression is exact if we assume that clauses are chosen with replacement). Since $2(1-2^{-k})^r < 1$ for $r \geq 2^k \ln 2$, this implies that for all such r the probability of satisfiability is exponentially small. The sharpening of this upper bound for general k is due, independently, to Dubois and Boufkhad [DB97] and Kirousis et al. [KKKS98]. It corresponds to bounding the probability of satisfiability by 2^n times the probability that a particular assignment is a *locally maximum* satisfying assignment, i.e., one in which no 0 can be turned into 1 without violating satisfiability (it is clear that every satisfiable formula has at least one such satisfying assignment, e.g., the lexicographically greatest one).

The lower bound in Theorem 3 is due to Achlioptas and Peres [AP04] and was proven via the, so-called, second moment method. It corresponds to the largest density for which $F_k(n,m)$ w.h.p. has *balanced* satisfying assignments, i.e., satisfying assignments in which the number of satisfied literals is $km/2 + O(n)$. In other words, balanced satisfying assignments have the property that in spite of being satisfying their number of satisfied literals is like that of a uniformly random $\sigma \in \{0,1\}^n$. Focusing on balanced assignments is done of technical necessity for the second moment method to work (we discuss this point in detail in Section 8.6.3) and there does not appear to be an inherent reason for doing so. Indeed, as k grows, the upper bound of Theorem 3 coincides with the presults[1] for the location of the threshold (up to a $o(1)$ term in k), giving even more evidence that the $O(k)$ term in the lower bound is an artifact of the analysis.

The second moment method, used to prove the lower bound, ignores individual solutions and captures, instead, statistical properties of the entire solution–space. As such, it offers no guidance whatsoever on how to efficiently *find* satisfying assignments for those densities for which it establishes their existence. Indeed, as we will see shortly, no efficient algorithm is known that can find solutions beyond density $O(2^k/k)$, even w.u.p.p. In the third row of Table 8.1 we indicate this phenomenon by giving the largest densities for which any efficient algorithm is known to succeed [KKL06, FS96]).

Finally, we note that while both general bounds above extend to $k = 3$, better bounds exist for that case. In particular, the lower bound for r_3 in Table 8.1 is actually algorithmic and due, independently, to Hajiaghayi and Sorkin [HS03] and Kaporis, Kirousis and Lalas [KKL06]. We discuss its derivation in Section 8.7.2. The upper bound is due to Dubois, Boufkhad and Mandler [DBM03] and uses the idea of locally maximum assignments mentioned above in combination with conditioning on the literal degree sequence, thus curtailing certain large deviations contributions that inflate the unconditional expectation.

[1] We use the term *presults* as a way of referring to "p[hysics] results" in this area, the subject of Part 1, Chapter 9. In general, presults rest on combining mathematically rigorous arguments with highly non-trivial unproven assumptions, the latter often informed by general considerations of statistical physics. The term is also motivated by the fact that many presults eventually became rigorous mathematical results via proofs that were deeply informed by the physical arguments.

Table 8.1. Best known rigorous bounds for the location of the satisfiability threshold for some small values of k. The last row gives the largest density for which a polynomial-time algorithm has been proven to find satisfying assignments.

k	3	4	5	7	10	20
Best upper bound	4.51	10.23	21.33	87.88	708.94	726,817
Best lower bound	3.52	7.91	18.79	84.82	704.94	726,809
Algorithmic lower bound	3.52	5.54	9.63	33.23	172.65	95,263

8.3. Random MAX k-SAT

The methods used to derive bounds for the location of the satisfiability threshold, also give bounds for the fraction of clauses that can be satisfied above it. Specifically, let us say that a k-CNF formula with m clauses is *p-satisfiable*, for some $p \in [0,1]$, if there exists an assignment satisfying at least $\left(1 - \frac{1-p}{2^k}\right) m$ clauses. Observe that every k-CNF formula is 0-satisfiable, since the average number of satisfied clauses over all assignments is $(1-2^{-k})m$, while satisfiability is simply 1-satisfiability. Thus, given $p \in (0,1]$, the relevant quantity is the largest density, $r_k(p)$, for which a random k-CNF formula is w.h.p. p-satisfiable. Analogously to the case $p = 1$, the naive union bound over $\{0,1\}^n$ for the existence of a p-satisfying truth assignment implies $r_k(p) \leq T_k(p)$, where

$$T_k(p) = \frac{2^k \ln 2}{p + (1-p)\ln(1-p)} \ .$$

Note that since $\lim_{p \to 1} T_k(p) = 2^k \ln 2$, this recovers the naive satisfiability upper bound. Using the second moment method, in [ANP07] it was shown that asymptotically, this upper bound is tight up to second order terms.

Theorem 4. *There exists a sequence $\delta_k = O(k 2^{-k/2})$, such that for all $k \geq 2$ and $p \in (0,1]$,*

$$(1 - \delta_k)\, T_k(p) \ \leq \ r_k(p) \ \leq \ T_k(p) \ .$$

Algorithms fall far short from this bound. Analogously to satisfiability, the best known algorithm [CGHS04] finds p-satisfying assignments only for $r = O(T_k(p)/k)$.

8.3.1. The case $k \in [2,3)$

For $k = 2$, satisfiability can be decided in polynomial time using a very simple method: tentatively set any unset variable v to 0; repeatedly satisfy any 1-clauses that result; if a 0-clause is generated set v permanently to 1, otherwise set it permanently to 0. A 2-CNF formula is satisfiable iff when this process terminates no 0-clauses are present. For random 2-CNF formulas, the trees of implications generated by this process mimick the connected component structure of random digraphs. Using this connection, Chvátal and Reed [CR92], Goerdt [Goe96] and Fernandez de la Vega [FdlV92] independently proved $r_2 = 1$. Later, in [BBC$^+$01], Bollobás et al. [BBC$^+$01], also using this connection, determined the scaling window for random 2-SAT, showing that the transition from satisfiability to unsatisfiability occurs for $m = n + \lambda n^{2/3}$ as λ goes from $-\infty$ to $+\infty$.

In [MZK$^+$, MZK$^+$99a, MZ98, MZK$^+$99b], Monasson et al., using mathematically sophisticated but non-rigorous techniques of statistical physics, initiated the analytical study of random CNF formulas that are mixtures of 2- and 3-clauses, which they dubbed $(2+p)$-CNF. Such formulas arise for a number of reasons. For example, a frequent observation when converting problems from other domains into satisfiability problems is that they result into mixed CNF formulas with a substantial number of clauses of length 2, along with the clauses of length 3. Another reason is that DPLL algorithms run by recursively solving satisfiability on *residual formulas*, restricted versions of their input CNF formula, which are mixtures of clauses of length at least 2. When given random 3-CNF formulas as input, many DPLL algorithms produce residual formulas that are mixtures of random 2- and 3-clauses, making properties of random $(2 + p)$-CNF crucial for analyzing their running time.

A random $(2 + p)$-CNF on n variables with m clauses is formed by choosing pm clauses of length 3 and $(1-p)m$ clauses of length 2, amongst all clauses of each length, uniformly and independently. Thus, $p = 0$ corresponds to random 2-SAT, while $p = 1$ corresponds to random 3-SAT. Below we will find it convenient to sidestep this original formulation and state results directly in terms of the number of 2- and 3-clauses, not of the total number of clauses m and p.

The fact $r_2 = 1$ implies that for any $\epsilon > 0$ a random 2-CNF formula on n variables with $(1-\epsilon/2)n$ clauses is w.h.p. satisfiable, but adding ϵn 2-clauses to it w.h.p. results in an unsatisfiable formula. The presults in [MZK$^+$99b] suggested, rather remarkably, that if instead of adding ϵn random 2-clauses one adds up to $0.703...n$ random 3-clauses, the formula remains satisfiable. In other words, that there is no finite "exchange rate" between 2- and 3-clauses.

Inspired by these claims, Achlioptas et al. [AKKK01] proved that

Theorem 5. *A random CNF formula with n variables, $(1-\epsilon)n$ 2-clauses and Δn 3-clauses is w.h.p. satisfiable for all $\epsilon > 0$ and all $\Delta \leq 2/3$, but w.h.p. unsatisfiable for $\epsilon = 0.001$ and $\Delta = 2.28$.*

In other words, the physical prediction of an infinite exchange ratio is valid as one can add at least $0.66n$ 3-clauses (and no more than $2.28n$). In [Ach99] it was conjectured that the inequality $\Delta \leq 2/3$ in Theorem 5 is tight. That is,

Conjecture 6. *For all $\Delta > 2/3$, there exists $\epsilon = \epsilon(\Delta) > 0$, such that a random CNF formula with n variables, $(1 - \epsilon)n$ 2-clauses and Δn 3-clauses is w.h.p. unsatisfiable.*

Conjecture 6 is supported by a presult of Biroli, Monasson and Weigt [BMW00], subsequent to [MZK$^+$, MZK$^+$99a], asserting that $2/3$ is indeed tight. As we will see, if true, it implies that the running time of a large class of DPLL algorithms exhibits a sharp threshold behavior: for each algorithm $\mathcal{A}$, there exists a critical density $r_{\mathcal{A}} \leq r_k$ such that $\mathcal{A}$ takes linear time for $r < r_{\mathcal{A}}$, but exponential time for $r > r_{\mathcal{A}}$.

8.3.2. Proof Complexity and its Implications for Algorithms

We saw that sparse random k-CNF formulas are hard to prove unsatisfiable using resolution [CS88]. Ben-Sasson and Impagliazzo [BSI99] and Alekhnovich and

Razborov [AR01] proved that the same is true for the polynomial calculus, while Alekhnovich [Ale05] proved its hardness for k-DNF resolution. Random k-CNF formulas are believed to be hard for other proof systems also, such as cutting planes, and this potential hardness has been linked to hardness of approximation [Fei02]. Moreover, the hardness of proving their unsatisfiability has been explored for dense formulas, where for sufficiently high densities it provably disappears [BKPS02, FGK05, GL03, COGLS04, COGL07]. For a more general discussion of the connections between k-CNF formulas and proof-complexity see Chapter **??**. Here we will focus on the resolution complexity of random k-CNF formulas as it has immediate and strong implications for the most commonly used satisfiability algorithm, namely the DPLL procedure.

8.3.2.1. Resolution Complexity of k-CNF formulas

The resolution rule allows one to derive a clause $(A \vee B)$ from two clauses $(A \vee x)$ and $(B \vee \bar{x})$. A resolution derivation of a clause C from a CNF formula F is a sequence of clauses $C_1, \ldots, C_\ell = C$ such that each C_i is either a clause of F or follows from two clauses C_j, C_k for $j, k < i$ using the resolution rule. A resolution refutation of an unsatisfiable formula F is a resolution derivation of the empty clause. The size of a resolution refutation is its number of clauses.

In contrast, the Davis-Putnam/DLL (DPLL) algorithm on a CNF formula F performs a backtracking search for a satisfying assignment of F by extending partial assignments until they either reach a satisfying assignment or violate a clause of F. It is well known that for an unsatisfiable formula F, the tree of nodes explored by any DPLL algorithm can be converted to a resolution refutation of F where the pattern of inferences forms the same tree.

For random k-CNF formulas in the unsatisfiable regime, the behavior of DPLL algorithms, and the more general class of resolution-based algorithms, is well-understood. Specifically, since every unsatisfiable 2-CNF formula has a linear-size resolution refutation, if $r > 1$ then even the simplest DPLL algorithms w.h.p. run in polynomial time on a random 2-CNF formula. On the other hand, for $k \geq 3$ the aforementioned result of Chvátal and Szemerédi [CS88] asserts that w.h.p. a random k-CNF formula in the unsatisfiable regime requires an exponentially long resolution proof of unsatisfiability. More precisely, let $\mathrm{res}(F)$ be the size of the minimal resolution refutation a formula F (assume $\mathrm{res}(F) = \infty$ if F is satisfiable). In [CS88] it was proved that

Theorem 7. *For all $k \geq 3$ and any constant $r > 0$, w.h.p. $\mathrm{res}(F_k(n, rn)) = 2^{\Omega(n)}$.*

Corollary 8. *Every DPLL algorithm w.h.p. takes exponential time on $F_k(n, rn)$, for any constant $r \geq 2^k \ln 2$.*

8.3.2.2. $(2+p)$-CNF formulas and their Algorithmic Implications

Since all unsatisfiable 2-CNF formulas have linear-size resolution refutations and $r_2 = 1$, it follows that adding $(1+\epsilon)n$ random 2-clauses to a random 3-CNF formula w.h.p. causes its resolution complexity to collapse from exponential to linear. In [ABM04b], Achlioptas, Beame and Molloy proved that, in contrast,

adding $(1-\epsilon)n$ random 2-clauses w.h.p. has essentially no effect on its proof complexity.

Theorem 9. *For any constants $r, \epsilon > 0$, let F be a random formula with n variables, $(1-\epsilon)n$ 2-clauses and rn 3-clauses. W.h.p.* $\mathrm{res}(F) = 2^{\Omega(n)}$.

Theorem 9 allows one to readily prove exponential lower bounds for the running times of DPLL algorithms for *satisfiable* random k-CNF formulas. This is because many natural DPLL algorithms when applied to random k-CNF formulas generate at least one unsatisfiable subproblem consisting of a random mixture of 2- and higher-length clauses, where the 2-clauses alone are satisfiable. (We will discuss how, when and for which algorithms this happens in greater detail in Section 8.9.) By Theorem 9, such a mixture has exponential resolution complexity (converting k-clauses with $k > 3$ to 3-clauses arbitrarily can only reduce resolution complexity) and, as a result, to resolve any such subproblem (and backtrack) any DPLL algorithm needs exponential time.

8.4. Physical Predictions for Solution-space Geometry

Random k-SAT, along with other random Constraint Satisfaction Problems, such as random graph coloring and random XOR-SAT have also been studied systematically in physics in the past two decades. For a general introduction and exposition to the physical methods see Part 1, Chapter 9. In particular, motivated by ideas developed for the study of materials known as spin glasses, physicists have put forward a wonderfully complex picture about how the geometry of the set of satisfying assignments of a random k-CNF formula evolves as clauses are added. Perhaps the most detailed and sophisticated version of this picture comes from [KMRT$^+$07].

Roughly speaking, statistical physicists have predicted (using non-rigorous but mathematically sophisticated methods) that while for low densities the set of satisfying assignments forms a single giant cluster, at some critical density this cluster shatters into exponentially many clusters, each of which is relatively tiny and far apart from all other clusters. These clusters are further predicted to be separated from one another by huge "energy barriers", i.e., every path connecting satisfying assignments in different clusters must pass through assignments that violate $\Omega(n)$ constraints. Moreover, inside each cluster the majority of variables are predicted to be frozen, i.e., take the same value in all solutions in the cluster; thus getting even a single frozen variable wrong requires traveling $\Omega(n)$ away and over a huge energy barrier to correct it.

The regime of exponentially many tiny clusters, each one having constant probability of vanishing every time a clause is added (due to its frozen variables), is predicted in [KMRT$^+$07] to persist until very close to the threshold, namely for densities up to

$$2^k \ln 2 - \frac{3 \ln 2}{2} + o(1) \ , \tag{8.1}$$

where the $o(1)$ term is asymptotic in k. In comparison, the satisfiability threshold

is predicted [MMZ06] to occur at

$$2^k \ln 2 - \frac{1+\ln 2}{2} + o(1) \ , \tag{8.2}$$

i.e., fewer than $0.2n$ k-clauses later. For densities between (8.1) and (8.2) , it is predicted that nearly all satisfying assignments lie in a small (finite) number of (atypically large) clusters, while exponentially many clusters still exist.

8.4.1. The Algorithmic Barrier

Perhaps the most remarkable aspect of the picture put forward by physicists is that the predicted density for the shattering of the set of solutions into exponentially many clusters scales, asymptotically in k, as

$$\ln k \cdot \frac{2^k}{k} \ , \tag{8.3}$$

fitting perfectly with the fact that all known algorithms fail at some density

$$c_{\mathcal{A}} \cdot \frac{2^k}{k} \ , \tag{8.4}$$

where the constant $c_{\mathcal{A}}$ depends on the algorithm. We will refer to this phenomenon as the "algorithmic barrier". We note that so far there does not appear to be some natural general upper bound for $c_{\mathcal{A}}$ for the types of algorithms that have been analyzed, i.e., more sophisticated algorithms do have a greater constant, but with rapidly diminishing returns in terms of their complexity.

8.4.2. Rigorous Results for Solution-space Geometry

By now a substantial part of the picture put forward by physicists has been made rigorous, at least for $k \geq 8$. Success for smaller k stumbles upon the fact that the rigorous results rely on the second moment method which does not seem to perform as well for small k.

For the definitions in the rest of this section we assume we are dealing with an arbitrary CNF formula F defined over variables $X = x_1, \ldots, x_n$, and we let $\mathcal{S}(F) \subseteq \{0,1\}^n$ denote its set of satisfying assignments. We say that two assignments are adjacent if their Hamming distance is 1 and we let $H_F : \{0,1\}^n \to \mathbb{N}$ be the function counting the number of clauses of F violated by each $\sigma \in \{0,1\}^n$.

Definition 10. *The **clusters** of a formula F are the connected components of $\mathcal{S}(F)$. A **region** is a non-empty union of clusters. The **height** of any path $\sigma_0, \sigma_1, \ldots, \sigma_t \in \{0,1\}^n$ is $\max_i H(\sigma_i)$.*

One can get an easy result regarding the solution-space geometry of *very* sparse random formulas by observing that if a formula F is satisfiable by the *pure literal* rule alone, then the set $\mathcal{S}(F)$ is connected (recall that the pure literal rule amounts to permanently satisfying any literal ℓ whose complement does not appear in the formula and permanently removing all clauses containing ℓ; the

proof of connectivity is left as an exercise). The pure literal rule alone w.h.p. finds satisfying assignments in random k-CNF formulas with up to $\rho_k \cdot n$ clauses, for some $\rho_k \to 0$, so this is very far away from the densities up to which the best known algorithms succeed, namely $O(2^k/k)$.

The following theorem of Achlioptas and Coja-Oghlan [ACO08], on the other hand, asserts that for all $k \geq 8$, precisely at the asymptotic density predicted by physics in (8.3), the set of satisfying assignments shatters into an exponential number of well-separated regions. Given two functions $f(n), g(n)$, let us write $f \sim g$ if $= \lim_{n\to\infty} f(n)/g(n) = 1$. Recall that the lower bound for the location of the satisfiability threshold provided by the second moment method is $s_k \equiv 2^k \ln 2 - (k+1)\frac{\ln 2}{2} - 1 - o(1) \sim 2^k \ln 2$.

Theorem 11. *For all $k \geq 8$, there exists $c_k < s_k$ such that for all $r \in (c_k, s_k)$, the set $\mathcal{S}(F_k(n, rn))$ w.h.p. consists of exponentially many regions where:*

1. *Each region only contains an exponentially small fraction of $\mathcal{S}$.*
2. *The Hamming distance between any two regions is $\Omega(n)$.*
3. *Every path between assignments in distinct regions has height $\Omega(n)$.*

In particular, $c_k \sim \ln k \cdot 2^k/k$.

The picture of Theorem 11 comes in even sharper focus for large k. In particular, for sufficiently large k, sufficiently close to the threshold, the regions become arbitrarily small and maximally far apart, while still exponentially many.

Theorem 12. *For any $0 < \delta < 1/3$, fix $r = (1-\delta)2^k \ln 2$, and let*

$$\alpha_k = \frac{1}{k} \ , \qquad \beta_k = \frac{1}{2} - \frac{5}{6}\sqrt{\delta} \ , \qquad \epsilon_k = \frac{\delta}{2} - 3k^{-2} \ .$$

For all $k \geq k_0(\delta)$, w.h.p. the set $\mathcal{S} = \mathcal{S}(F_k(n, rn))$ consists of $2^{\epsilon_k n}$ regions where:

1. *The diameter of each region is at most $\alpha_k n$.*
2. *The distance between every pair of regions is at least $\beta_k n$.*

Thus, the solution-space geometry becomes like that of an correcting code with a little bit of "fuzz" around each codeword. This "shattering" phase transition is known as a *dynamical* transition, while the transition in which nearly all assignments concentrate on an finite number of clusters is known as a *condensation* transition. This dynamical phase transition has strong negative repercussions for the performance of random-walk type algorithms on random k-CNF formulas and exploring them precisely is an active area of research.

Remark 13. *The presults of [KMRT$^+$07] state that the picture of Theorem 11 should hold for all $k \geq 4$, but* **not** *for $k = 3$. In particular for $k = 3$, the solution-space geometry is predicted to pass from a single cluster to a condensed phase directly.*

It turns out that after the set of satisfying assignments has shattered into exponentially many clusters, we can "look inside" these clusters and make some statements about their shape. For that, we need the following definition.

Definition 14. *The **projection** of a variable x_i over a set of assignments C, denoted as $\pi_i(C)$, is the union of the values taken by x_i over the assignments in C. If $\pi_i(C) \neq \{0,1\}$ we say that x_i is **frozen** in C.*

Theorem 15 below [ACO08] asserts that the dynamical transition is followed, essentially immediately, by the massive appearance of frozen variables.

Theorem 15. *For all $k \geq 8$, there exists $d_k < s_k$ such that for all $r \in (d_k, s_k)$, a random $\sigma \in \mathcal{S}(F_k(n, rn))$ w.h.p. has at least $\gamma_k \cdot n$ frozen variables, where $\gamma_k \to 1$ for all $r \in (d_k, s_k)$. In particular, $d_k \sim \ln k \cdot 2^k/k$.*

The first rigorous analysis of frozen variables by Achlioptas and Ricci-Tersenghi [ART06] also establishes that

Corollary 16. *For every $k \geq 9$, there exists $r < s_k$ such that w.h.p.* **every** *cluster of $F_k(n, rn)$ has frozen variables.*

It remains open whether frozen variables exist for $k < 8$.

8.5. The Role of the Second Moment Method

Recall that the best upper bound for the satisfiability threshold scales as $\Theta(2^k)$, whereas all efficient algorithms known work only for densities up to $O(2^k/k)$. To resolve whether this gap was due to a genuine lack of solutions, as opposed to the difficulty of finding them, Achlioptas and Moore [AM06] introduced an approach which allows one to avoid the pitfall of computational complexity: namely, using the second-moment method one can prove that solutions exist in random instances, without the need to identify any particular solution for each instance (as algorithms do). Indeed, if random formulas are genuinely hard at some densities below the satisfiability threshold, then focusing on the *existence* of solutions rather than their efficient discovery is essential: one cannot expect algorithms to provide accurate results on the threshold's location; they simply cannot get there!

Before we delve into the ideas underlying some of the results mentioned above it is helpful to establish the probabilistic equivalence between a few different models for generating random k-CNF formulas.

8.6. Generative models

Given a set V of n Boolean variables, let $C_k = C_k(V)$ denote the set of all proper k-*clauses* on V, i.e., the set of all $2^k\binom{n}{k}$ disjunctions of k literals involving distinct variables. As we saw, a random k-CNF formula $F_k(n, m)$ is formed by selecting a uniformly random m-subset of C_k. While $F_k(n, m)$ is perhaps the most natural model for generating random k-CNF formulas, there are a number of slight variations of the model, largely motivated by their amenability to calculations.

For example, it is fairly common to consider the clauses as ordered k-tuples (rather than as k-sets) and/or to allow replacement in sampling the set C_k. Clearly, for properties such as satisfiability the issue of ordering is irrelevant. Moreover, as long as $m = O(n)$, essentially the same is true for the issue of replacement. To see that, observe that w.h.p. the number, q of repeated clauses is

$o(n)$, while the set of $m-q$ distinct clauses is a uniformly random $(m-q)$-subset of C_k. Thus, if a monotone decreasing property (such as satisfiability) holds with probability p for a given $m = r^*n$ when replacement is allowed, it holds with probability $p - o(1)$ for all $r < r^*$ when replacement is not allowed.

The issue of selecting the literals of each clause with replacement (which might result in some "improper" clauses) is completely analogous. That is, the probability that a variable appears more than once in a given clause is at most $k^2/n = O(1/n)$ and hence w.h.p. there are $o(n)$ improper clauses. Finally, we note that by standard techniques, e.g., see [FS96], results also transfer between $F_k(n,m)$ and the model where every clause in C_k is included in the formula independently of all others with probability p, when $pC_k \sim m$.

8.6.1. The Vanilla Second Moment Method

The second moment method approach [ANP05] rests on the following basic fact: every non-negative random variable X satisfies $\Pr[X > 0] \geq \mathbf{E}[X]^2/\mathbf{E}[X^2]$. Given any k-SAT instance F on n variables, let $X = X(F)$ be its number of satisfying assignments. By computing $\mathbf{E}[X^2]$ and $\mathbf{E}[X]^2$ for random formulas with a given density r one can hope to get a lower bound on the probability that $X > 0$, i.e., that $F_k(n, rn)$ is satisfiable. Unfortunately, this direct application fails dramatically as $\mathbf{E}[X^2]$ is exponentially (in n) greater than $\mathbf{E}[X]^2$ for every density $r > 0$. Nevertheless, it is worth going through this computation below as it points to the source of the problem and also helps in establishing the existence of clusters. We perform all computations below in the model where clauses are chosen with replacement from C_k.

For a k-CNF formula with m clauses chosen independently with replacement it is straightforward to show that the number of satisfying assignments X satisfies

$$\mathbf{E}[X^2] = \sum_{z=0}^{n} 2^n \binom{n}{z} f_S(z/n)^m \ , \tag{8.5}$$

where $f_S(\alpha) = 1 - 2^{1-k} + 2^{-k}\alpha^k$ is the probability that two fixed truth assignments that agree on $z = \alpha n$ variables, *both* satisfy a randomly drawn clause. Thus, (8.5) decomposes the second moment of X into the expected number of pairs of satisfying assignments at each possible distance.

Observe that f is an increasing function of α and that $f_S(1/2) = (1 - 2^{-k})^2$, i.e., truth assignments at distance $n/2$ are uncorrelated. Using the approximation $\binom{n}{\alpha n} = (\alpha^\alpha (1-\alpha)^{1-\alpha})^{-n} \times \mathrm{poly}(n)$ and letting

$$\Lambda_S(\alpha) = \frac{2\, f_S(\alpha)^r}{\alpha^\alpha (1-\alpha)^{1-\alpha}}$$

we see that

$$\begin{aligned} \mathbf{E}[X^2] &= \left(\max_{0 \leq \alpha \leq 1} \Lambda_S(\alpha) \right)^n \times \mathrm{poly}(n) \ , \\ \mathbf{E}[X]^2 &= \Lambda_S(1/2)^n \ . \end{aligned}$$

Therefore, if there exists some $\alpha \neq 1/2$ such that $\Lambda_S(\alpha) > \Lambda_S(1/2)$, then the second moment is exponentially greater than the square of the expectation and we only get an exponentially small lower bound for $\Pr[X > 0]$. Put differently, unless the dominant contribution to $\mathbf{E}[X^2]$ comes from uncorrelated pairs of satisfying assignments, i.e., pairs with overlap $n/2$, the second moment method fails.

Unfortunately, this is precisely what happens for all $r > 0$ since the entropic factor $\mathcal{E}(\alpha) = 1/(\alpha^\alpha(1-\alpha)^{1-\alpha})$ in Λ_S is symmetric around $\alpha = 1/2$, while f_S is increasing in $(0,1)$, implying $\Lambda'_S(1/2) > 0$. That is, Λ_S is maximized at some $\alpha > 1/2$ where the correlation benefit balances the penalty of decreased entropy.

While the above calculation does not give us what we want, it is still useful. For example, for any real number $\alpha \in [0,1]$, it would be nice to know the number of pairs of satisfying truth assignments that agree on $z = \alpha n$ variables in a random formula. Each term in the sum in (8.5) gives us the *expected* number of such pairs. While this expectation may overemphasize formulas with more satisfying assignments (as they contribute more heavily to the expectation), it still gives valuable information on the distribution of distances among truth assignments in a random formula. For example, if for some values of z (and k, r) this expectation tends to 0 with n, we can readily infer that w.h.p. there are no pairs of truth assignments that agree on z variables in $F_k(n, rn)$. This is because for any integer-valued random variable Y, $\Pr[Y > 0]] \leq \mathbf{E}[Y]$. Indeed, this simple argument is enough to provide the existence of clustering for all $k \geq 8$, at densities in the $\Theta(2^k)$ range (getting down to $(2^k/k) \cdot \ln k$ requires a lot more work).

8.6.2. Proving the Existence of Exponentially Many Clusters

To prove the existence of exponentially many regions one divides a lower bound for the total number of satisfying assignments with an upper bound for the number of truth assignments in each region. The lower bound comes from the expected number of *balanced* satisfying assignments since the success of the second moment method for such assignments, which we will see immediately next, implies that w.h.p. the actual number of balanced assignments is not much lower than its expectation. For the upper bound, one bounds the total number of *pairs* of truth assignments in each region as $\text{poly}(n) \times g(k,r)^n$, where

$$g(k,r) = \max_{\alpha \in [0,\Delta]} \Lambda_S(\alpha, k, r) \ ,$$

where Δ is the smallest number such that $\Lambda_S(\Delta, k, r) < 1$, i.e., Δn is a bound on the diameter of any region. Dividing the lower bound for the total number of satisfying assignments with the square root of this quantity yields the existence of exponentially many clusters.

8.6.3. Weighted second moment: the importance of being balanced

An attractive feature of the second moment method is that we are free to apply it to any random variable $X = X(F)$ such that $X > 0$ implies that F is satisfiable. With this in mind, let us consider random variables of the form

$$X = \sum_{\sigma} \prod_{c} w(\sigma, c)$$

where w is some arbitrary function. (Eventually, we will require that $w(\sigma, c) = 0$ if σ falsifies c.) Similarly to (8.5), it is rather straightforward to prove that

$$\mathbf{E}[X^2] = 2^n \sum_{z=0}^{n} \binom{n}{z} f_w(z/n)^m \ ,$$

where $f_w(z/n) = \mathbf{E}[w(\sigma, c)\, w(\tau, c)]$ is the correlation, with respect to a single random clause c, between two truth assignments σ and τ that agree on z variables. It is also not hard to see that $f_w(1/2) = \mathbf{E}[w(\sigma, c)]^2$, i.e., truth assignments at distance $n/2$ are uncorrelated for *any* function w. Thus, arguing as in the previous section, we see that $\mathbf{E}[X^2]$ is exponentially greater than $\mathbf{E}[X]^2$ unless $f'_w(1/2) = 0$.

At this point we observe that since we are interested in random formulas where literals are drawn uniformly, it suffices to consider functions w such that: for every truth assignment σ and every clause $c = \ell_1 \vee \cdots \vee \ell_k$, $w(\sigma, c) = w(\mathbf{v})$, where $v_i = +1$ if ℓ_i is satisfied under σ and -1 if ℓ_i is falsified under σ. (So, we will require that $w(-1, \ldots, -1) = 0$.) Letting $A = \{-1, +1\}^k$ and differentiating f_w, yields the geometric condition

$$f'_w(1/2) = 0 \iff \sum_{\mathbf{v} \in A} w(\mathbf{v})\mathbf{v} = 0 \ . \tag{8.6}$$

The condition in the r.h.s. of (8.6) asserts that the vectors in A, when scaled by $w(\mathbf{v})$, must cancel out. This gives us another perspective on the failure of the vanilla second moment method: when $w = w_S$ is the indicator variable for satisfiability, the condition in the right hand side of (8.6) does not hold since the vector $(-1, -1, \ldots, -1)$ has weight 0, while all other $\mathbf{v} \in A$ have weight 1.

Note that the r.h.s. of (8.6) implies that in a successful w each coordinate must have mean 0, i.e., that each literal must be equally likely to be $+1$ or -1 when we pick truth assignments with probability proportional to their weight under w. We will call truth assignments with $km/2 \pm O(\sqrt{m})$ satisfied literal occurrences *balanced*. As was shown in [AP04], if X is the number of balanced satisfying assignments then $\mathbf{E}[X^2] < C \cdot \mathbf{E}[X]^2$, for all $r \leq s_k$, where $C = C(k) > 0$ is independent of n. Thus, $\Pr[X > 0] \geq 1/C$ and by Corollary 2 we get $r_k \geq s_k$.

To gain some additional intuition on the success of balanced assignments it helps to think of $F_k(n, m)$ as generated in two steps: first choose the km literal occurrences randomly, and then partition them randomly into k-clauses. At the end of the first step, truth assignments that satisfy many literal occurrences cleary have significantly greater conditional probability of eventually being satisfying assignments. But such assignments are highly correlated with each other since in order to satisfy many literal occurrences they tend to agree with the majority truth assignment on more than half the variables. Focusing on balanced assignments only, curbs the tendency of satisfying assignments to lean towards the majority vote assignment.

Finally, we note that it is not hard to prove that for $r \geq s_k + O(1)$, w.h.p. $F_k(n, rn)$ has no satisfying truth assignments that only satisfy $km/2 + o(n)$ literal occurrences. Thus, any asymptotic improvement over the lower bound for r_k provided by balanced assignments would mean that tendencies toward the majority assignment become essential as we approach the threshold. By the same token,

we note that as k increases the influence exerted by the majority vote assignment becomes less and less significant as most literals occur very close to their expected $kr/2$ times. As a result, as k increases, typical satisfying assignments get closer and closer to being balanced, meaning that the structure of the space of solutions for small values of k (e.g., $k = 3, 4$) might be significantly different from the structure for large values of k, something also predicted by the physical methods.

8.7. Algorithms

For the purposes of this chapter we will categorize satisfiability algorithms into two broad classes: DPLL algorithms and random walk algorithms. Within the former, we will distinguish between backtracking and non-backtracking algorithms. More concretely, the DPLL procedure for satisfiability is as follows:

DPLL(F)

1. Repeatedly satisfy any pure literals and 1-clauses.
   If the resulting formula F' is empty, exit reporting "satisfiable".
   If a contradiction (0-clause) is generated, exit.
2. Select a variable $x \in F'$ and a value v for x
3. DPLL($F'_{v=x}$)
4. DPLL($F'_{v=1-x}$)

Clearly, different rules for performing Step 2 give rise to different algorithms and, in practice, the complexity of these rules can vary from minimal to huge. Perhaps the simplest possible rule is to consider the variables in a fixed order, e.g., $x_1, x_2, \ldots, x_n$, always selecting the first variable in the order that is present in the formula, and always setting x to the same value, e.g., $x = 0$. If one forgoes the pure literal rule, something which simplifies the mathematical analysis on random formulas, the resulting algorithms in known as ORDERED DLL. If one also forgoes backtracking, the resulting algorithm is known as UC, for Unit Clause propagation, and was one the first algorithms to be analyzed on random formulas.

8.7.1. Random walk algorithms

In contrast to DPLL algorithms, random walk algorithms always maintain an entire assignment which they evolve until either it becomes satisfying or the algorithm is exhausted. While, in principle, such an algorithm could switch the value of arbitrarily many variables at a time, typically only a constant number of variables are switched and, in fact, the most common case is to switch only one variable (hence "walk"). An idea that appears to make a significant difference in this context is that of *focusing* [SAO05], i.e., restricting the choice of variable(s) to switch among those contained in clauses violated by the present assignment.

Unfortunately, while there are numerous experimental results regarding performance of random walk algorithms on random formulas, the only mathematically rigorous result known is due to Alekhnovich and Ben-Sasson [ABS07] asserting that the algorithm "select a violated clause at random and among its

violated literals select one at random and flip it", due to Papadimitriou [Pap91], succeeds in linear time on random 3-CNF for densities as high as 1.63, i.e., up to the largest density for which the pure literal rule succeeds (recall that the success of the pure literal implies that the set of satisfying assignments is connected).

It is worth pointing out that, at this point, there exist both very interesting presults regarding the performance of random walk algorithms on random formulas [CMMS03, SM03] and also very interesting mathematical results [FMV06, COMV07] regarding their performance on the planted model, i.e., on formulas generated by selecting uniformly at random among all $2^{k-1}\binom{n}{k}$ k-clauses satisfied by a fixed (planted) assignment. Perhaps, our recent understanding of the evolution of the solution-space geometry of random k-CNF formulas will help in transferring some of these results.

8.7.2. Non-backtracking Algorithms

On random CNF formulas UC is equivalent to simply repeating the following until either no clauses remain (success), or a 0-clause is generated (failure): if there is a clause of length 1 satisfy it; otherwise, select a random unassigned variable and assign it a random value. What makes the analysis of UC possible is the fact that if the input is a random k-CNF formula, then throughout the execution of UC the following ia true. Let $V(t)$ be the set of variables that have not yet been assigned a value after t steps of the algorithm, and let $\mathcal{C}_i(t)$ be the set of clauses of length i at that time. Then the set $\mathcal{C}_i(t)$ is distributed exactly as a random i-CNF formula on the variables $V(t)$, with exactly $|\mathcal{C}_i(t)|$ clauses.

To see the above claim, imagine representing a CNF formula by a column of k cards for each k-clause, each card bearing the name of one literal. Assume, further, that originally all the cards are "face-down", i.e., the literal on each card is concealed (and we never had an opportunity to see which literal is on each card). At the same time, assume that an intermediary with photographic memory knows precisely which literal is on each card. To interact with the intermediary we are allowed to either

- Point to a particular card, or,
- Name a variable that has not yet been assigned a value.

In response, if the card we point to carries literal ℓ, the intermediary reveals (flips) all the cards carrying $\ell, \bar{\ell}$. Similarly, if we name variable v, the intermediary reveals all the cards carrying $v, \bar{v}$. In either case, faced with all the occurrences of the chosen variable we proceed to decide which value to assign to it. Having done so, we remove all the cards corresponding to literals dissatisfied by our setting and all the cards (some of them still concealed) corresponding to satisfied clauses. As a result, at the end of each step only "face-down" cards remain, containing only literals corresponding to unset variables. This card-game representation immediately suggests our claimed "uniform randomness" property for UC and this can be made easily rigorous by appealing to the method of "deferred decisions". In fact, *all* algorithms that can be carried out via this game enjoy this property.

Lemma 17 (Uniform randomness)**.** *If $V(t) = X$ and $|\mathcal{C}_i(t)| = q_i$, the set of i-clauses remaining at time t form $F_i(|X|, q_i)$ on the variables in X.*

Armed with such a simple representation of Markovian state, it is not hard to show that as long as an algorithm takes care of unit clauses whenever they exist, its success or failure rests on whether the set of 2-clauses ever acquires density greater than 1. The main tool used to determine whether that occurs it is to pass to a so-called "liquid" model and approximate the evolution of the number of clauses of each length via a system of differential equations. Indeed, both of these ideas (uniform randomness plus differential equations) were present in the work of Chao and Franco in the mid-80s [CF86], albeit not fully mathematically justified.

The card-game described above does not allow one to carry out the pure literal heuristic, as it offers no information regarding the number of occurrences of each literal in the formula. To allow for this, we switch to a slightly different model for generating random k-CNF formulas. First, we generate km literals independently, each literal drawn uniformly at random among all $2n$ literals and, then, we generate a formula by partitioning these km literals into k-clauses uniformly at random. (This might result in a few "improper" clauses; we addressed this essentially trivial point in Section 8.6.)

One can visualize the second part of this process as a uniformly random matching between km literals on the left and their km occurrences in k-clauses on the right. As the matching is uniformly random, similarly to the card game, it can be "exposed" on the fly. (Indeed, the card game above is just the result of concealing the left hand of the matching.) With this representation we can now execute algorithms such as: if there is a pure literal or a clause of length 1 satisfy it; otherwise, select a random literal of maximum degree and satisfy it; or, alternatively, select a "most polarized" variable and assign it its majority value. To analyze either one of these algorithms, note that it is enough to maintain as Markovian state the number of clauses of each length and the number of variables having i positive and j negative literal occurrences for each $i, j \geq 0$.

With this long introduction in place we can now describe the state of the art:

- Historically, the first fully rigorous lower bound for r_3 was given by Broder, Frieze and Upfal [BFU93] who considered the pure literal heuristic. They showed that for $r \leq 1.63$, w.h.p. this eventually sets all the variables (and that for $r > 1.7$ w.h.p. it does not). The exact threshold for its success was given later in [LMS98, Mol05].
- Following that, only algorithms expressible in the card-game model were analyzed for a while. In [AS00] Achlioptas and Sorkin showed that the optimal algorithm expressible in this model works up to 3.26... For a survey of the state of the art up to that point, along with a unified framework for analyzing satisfiability algorithms via differential equations, see [Ach01].
- The configuration model for analyzing satisfiability algorithms has been taken up for $k = 3$. Specifically, the simple algorithm mentioned above, "in the absence of unit clauses satisfy a random literal of maximum degree", was proposed and analyzed by Kaporis, Kirousis and Lalas [KKL02] who showed that it succeeds w.h.p. up to density 3.42... More complicated algorithms that select which literal to satisfy by considering the degree of each literal *and* that of its complement, achieve the best known lower bound, 3.52..., for the 3-SAT threshold, and were analyzed, independently,

by Kaporis, Kirousis and Lalas [KKL06] and Hajiaghayi and Sorkin [HS03]. It seems likely that this bound can be further improved, at least slightly, by making even more refined considerations in the choice of variable and value, but it also seems clear that this is well within the regime of diminishing returns.

- For $k > 3$, the best results come from consider the following very natural "shortest clause"algorithm SC: pick a random clause c of minimum length; among the literals in c pick one at random and satisfy it. A weakening of this algorithm which in the absence of 1-, 2-, and 3-clauses selects a random literal to satisfy was analyzed by Frieze and Suen in [FS96] and gives the best known algorithmic lower bound for the k-SAT threshold for $k > 3$, namely $r_k \geq \gamma_k 2^k/k$, where $\gamma_k \to 1.817$. In [FS96], the authors give numerical evidence that even the full SC algorithm only succeeds up to some $\zeta_k 2^k/k$, where $\zeta_k = O(1)$.

In a nutshell, the only algorithms that have been proven to find satisfying assignments efficiently in random k-CNF formulas are extremely limited and only succeed for densities up to $O(2^k/k)$. In particular, both their variable ordering and their value assignment heuristic can be implemented given very little, and completely local, information about the variabe-clause interactions. Of course, this limitation is also what enables their analysis.

Question 18. *Is there a polynomial-time algorithm which w.u.p.p. finds satisfying assignments of random k-CNF formulas of density $2^k/k^{1-\epsilon}$ for some $\epsilon > 0$?*

As we saw, the geometry of the space of satisfying assignments undergoes a phase transition at $\ln k \cdot 2^k/k$. As a result, an affirmative answer to Question 18 might require a significant departure from the type of algorithms that have been analyzed so far.

8.8. Belief/Survey Propagation and the Algorithmic Barrier

In [MPR02], Mézard, Parisi, and Zecchina proposed a new satisfiability algorithm called Survey Propagation (SP) which performs extremely well experimentally on instances of random 3-SAT. This was a big breakthrough and allowed for optimism that, perhaps, random k-SAT instances might not be so hard, even close to the threshold. Unfortunately, conducting experiments with random k-CNF formulas becomes practically harder at a rapid pace as k increases: the interesting densities scale as $\Theta(2^k)$ so, for example, already $k = 10$ requires extremely large n in order for the formulas to be plausibly considered sparse.

As mentioned earlier, in [KMRT$^+$07] it is predicted that just below the satisfiability threshold there is a small range of densities, scaling as $2^k \ln 2 - \Theta(1)$, for which although exponentially many clusters exist, almost all satisfying assignments lie in a *finite* number of (atypically large) clusters. This "condensation" of nearly all satisfying assignments to a small number of clusters induces long-range correlations among the variables, making it difficult to estimate their marginal distributions by examining only a bounded neighborhood around each variable. SP is an ingenuous heuristic idea for addressing this problem by considering not

the uniform measure over satisfying assignments but, rather, (an approximation of) the uniform measure over clusters, where each cluster is represented by the fixed point of a certain iterative procedure applied to any assignment in the cluster.

That said, for all densities below the condensation transition, SP is not strictly necessary: if SP can compute variable marginals, then so can a much simpler algorithm called Belief Propagation, i.e., dynamic programming on trees. This is because when the measure is carried by exponentially many well-scattered clusters, marginals are expected to decorrelate. Indeed Gershenfeld and Montanari [GM07] gave very strong rigorous evidence that BP succeeds in computing marginals in the uncondensed regime for the coloring problem. So, although SP might be useful when working very close to the threshold, it is not readily helpful in designing an algorithm that can *provably* find solutions even at *much* lower densities, e.g., say at $r = 2^{k-2}$, roughly in the middle of the satisfiable regime.

One big obstacle is that, currently, to use either BP or SP to find satisfying assignments one sets variables iteratively. When a constant fraction of the variables are frozen in each cluster, as is the case after the dynamical transition, setting a single variable typically eliminates a constant fraction of all clusters. As a result, very quickly, one can be left with so few remaining clusters that decorrelation stops to hold. Concretely, in [MRTS07], Montanari, Ricci-Tersenghi and Semerjian showed that (even with the relatively generous assumptions of statistical physics computations) the following algorithm **fails** for densities greater than $\ln k \cdot 2^k/k$. That is, step 2 below fails to converge after only a small fraction of all variables have been assigned a value:

1. Select a variable v at random.
2. Compute the marginal distribution of v using Belief Propagation.
3. Set v to $\{0, 1\}$ according to the computed marginal distribution; simplify the formula; go to step 1.

8.9. Backtracking Algorithms

Backtracking satisfiability algorithms operate by building an assignment step by step. In particular, the choices they make when no unit clauses and pure literals are present are called *free* and when those choices lead to contradictions (empty clauses), backtracking occurs. In contrast, their non-backtracking variants encountered in Section 8.7.2 simply give up when that happens. Moreover, the only non-backtracking algorithms that have been analyzed maintain their residual formula uniformly random conditional on some small notion of "state", reflecting the number of clauses of each length and perhaps, additionally, the number of occurrences of each literal. Let us call such algorithms "myopic".

It is not hard to prove that the largest density, $r_{\mathcal{A}}$, for which a myopic non-backtracking algorithm $\mathcal{A}$ will find satisfying assignments of a random k-CNF formula is precisely the largest density for which its residual 2-CNF subformula remains below density 1 throughout $\mathcal{A}$'s execution (see e.g. [Ach01]). For densities beyond $r_{\mathcal{A}}$ one can endow $\mathcal{A}$ with a backtracking scheme and attempt to analyze its performance. Unfortunately, any non-trivial amount of backtracking makes it

hard to have a compact probabilistic model for the residual formula. As a result, a probabilistic analysis akin to that possible for $r < r_{\mathcal{A}}$ appears beyond the reach of current mathematical techniques (but see [CM04, CM05, Mon05] for an analysis using techniques of statistical physics). Nevertheless, for *all* backtracking extensions of myopic algorithms it is possible to prove that they take exponential time when the initial k-CNF formula is above a certain critical density. This is because of the following immediate implication of Theorem 9.

Corollary 19. *If a DPLL algorithm ever generates a residual formula that is an unsatisfiable mixture of uniformly random clauses in which the 2-clause density is below 1, then w.h.p. it will spend exponential time before backtracking from it.*

That is, by Corollary 19, once a node in the backtracking search is reached that corresponds to an unsatisfiable random mixture (but where the 2-clauses alone are satisfiable), the search cannot leave the sub-tree for an exponentially long time. Standard results (see e.g. [Ach01]) imply that w.u.p.p. this is precisely what happens for UC started with $3.81n$ 3-clauses and for SC started with $3.98n$ 3-clauses. This is because for such initial densities, at some point, the corresponding algorithm w.u.p.p. generates a residual $(2+p)$-CNF formula which is unsatisfiable w.h.p. per the results of Theorem 5.

Theorem 20. *For any constant $r \geq 3.81$, any backtracking extension of* UC *w.u.p.p. takes time $2^{\Omega(n)}$ on $F_3(n, rn)$. Similarly for* SC *and $r \geq 3.98$.*

We note that the only reason for Theorem 20 is not a high probability result is that w.u.p.p. each algorithm might generate a contradiction and backtrack, thus destroying the uniform randomness of the residual formula, before creating a formula like the one mandated by Corollary 19. It is worth pointing out, though, that whenever this occurs w.h.p. it is for trivial local reasons. In particular, Frieze and Suen in [FS96], introduced the following form of backtracking: when a contradiction is reached, record the portion of the assignment between the last free choice and the contradiction; these literals become *hot*. After flipping the variable involved in the last free choice, instead of making the choice that the original heuristic would suggest, give priority to the complements of the hot literals in the order that they appeared; once the hot literals are exhausted continue as with the original heuristic. This backtracking rule is quite natural in that it is the last part of the partial assignment that got is into trouble in the first place. Moreover, it appears to be a genuinely good idea. Experiments on random formulas comparing this backtracking extension of UC with just reversing the last free choice show that the histogram of run-times is *significantly better* for a large range of densities [ABM04b]. Another property of this backtracking rule is that as long as the value of each variable in a partial assignment has been flipped at most once, as happens when dealing with trivial, local contradictions, the residual formula is uniformly random. In particular, for this particular backtracking extension Theorem 20 holds w.h.p.

Theorem 20 sheds light on a widely-cited observation of Mitchell, Selman and Levesque [MSL92], based on experiments with ORDERED-DLL on small problems, stating that random 3-SAT is easy in the satisfiable region up to the 4.2 threshold, becomes sharply much harder at the threshold and quickly becomes easy again at

larger densities in the unsatisfiable region. The upper end of this "easy-hard-easy" characterization is somewhat misleading since, as we saw, the result of [CS88] in fact asserts that w.h.p. random 3-CNF formulas only have exponential-size proofs of unsatisfiability above the threshold. By now, the rate of decline in running time as the density is increased has been analyzed as well by Beame et al. [BKPS02]. Theorem 20 shows that the lower end of this characterization is also misleading in that the onset of exponential behavior occurs significantly below the (conjectured) satisfiability threshold at density 4.2. This concurs with experimental evidence that even the best of current DPLL implementations seem to have bad behavior below the threshold [CDA$^+$03]. Moreover, as we will see shortly, the gap between the onset of exponential behavior and the satisfiability threshold increases rapidly with k.

Corollary 19 probably points to a much larger truth than what is specifically derived for the algorithms and backtracking schemes mentioned above. This is because the proof of Theorem 9 is quite robust with respect to the probability distribution of the clauses in the mixture. The essential ingredient seems to be that the variable-clause incidence graph is an expander, suggesting that, in fact, random k-CNF formulas are not the only formulas for which one could hope to prove a result similar to Theorem 20. Moreover, precisely the combinatorial richness of expanders suggests that restarting a DPLL algorithm on a random k-CNF formula is unlikely to yield dramatically different results from run to run, unless, of course, one is willing to restart an exponential number of times.

The bounds on the 3-clause density needed to cause exponential behavior in satisfiability algorithms will be readily improved with any improvement on the $2.28n$ upper bound for unsatisfiability in random $(2+p)$-SAT. In particular, if Conjecture 6 is true, then Theorem 9 implies [ABM04b] that the running time of every myopic algorithm goes from linear to exponential around a critical, algorithm-specific density.

8.10. Exponential Running-Time for $k > 3$

The most obvious drawback of Theorem 20 is that it implies exponential behavior for densities that are only *conjectured* (but not proven) to be in the "satisfiable regime". For all $k \geq 4$, the analogue to Theorem 20 holds for densities that are *provably* in the satisfiable regime [ABM04a]. Moreover, the ratio between the density at which the algorithms begins to require exponential time, and the greatest density for which formulas are known to be satisfiable is of order $1/k$. Concretely,

Theorem 21. *For $k = 4$ and $r \geq 7.5$ and for $k \geq 5$ and $r \geq (11/k)\,2^{k-2}$,* ORDERED-DLL *w.u.p.p. requires time $2^{\Omega(n)}$ on $F_k(n, rn)$.*

Analogues of Theorem 21 hold for many other backtracking extensions of "myopic" [Ach01] algorithms and, similarly to the results for $k = 3$, one can get hight probability results by considering the Frieze-Suen style backtracking.

References

[ABM04a] Dimitris Achlioptas, Paul W. Beame, and Michael Molloy. Exponential bounds for dpll below the satisfiability threshold. In *SODA*, pages 139–140, 2004.

[ABM04b] Dimitris Achlioptas, Paul W. Beame, and Michael S. O. Molloy. A sharp threshold in proof complexity yields lower bounds for satisfiability search. *J. Comput. Syst. Sci.*, 68(2):238–268, 2004.

[ABS07] Mikhail Alekhnovich and Eli Ben-Sasson. Linear upper bounds for random walk on small density random 3-CNFs. *SIAM J. Comput.*, 36(5):1248–1263, 2007.

[Ach99] Dimitris Achlioptas. *Threshold phenomena in random graph colouring and satisfiability.* PhD thesis, Toronto, Ont., Canada, Canada, 1999. Adviser-Allan Borodin and Adviser-Michael Molloy.

[Ach01] Dimitris Achlioptas. Lower bounds for random 3-SAT via differential equations. *Theor. Comput. Sci.*, 265(1-2):159–185, 2001.

[ACO08] Dimitris Achlioptas and Amin Coja-Oghlan. Algorithmic barriers from phase transitions, 2008. preprint.

[AKKK01] Dimitris Achlioptas, Lefteris M. Kirousis, Evangelos Kranakis, and Danny Krizanc. Rigorous results for random $(2 + p)$-SAT. *Theoret. Comput. Sci.*, 265(1-2):109–129, 2001.

[Ale05] Mikhail Alekhnovich. Lower bounds for k-dnf resolution on random 3-cnfs. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 251–256, New York, NY, USA, 2005. ACM.

[AM06] Dimitris Achlioptas and Cristopher Moore. Random k-SAT: Two moments suffice to cross a sharp threshold. *SIAM J. Comput.*, 36(3):740–762, 2006.

[ANP05] Dimitris Achlioptas, Assaf Naor, and Yuval Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435:759–764, 2005.

[ANP07] Dimitris Achlioptas, Assaf Naor, and Yuval Peres. On the maximum satisfiability of random formulas. *J. ACM*, 54(2), 2007.

[AP04] Dimitris Achlioptas and Yuval Peres. The threshold for random k-SAT is $2^k \log 2 - O(k)$. *J. Amer. Math. Soc.*, 17(4):947–973, 2004.

[AR01] Mikhail Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. In *FOCS*, pages 190–199, 2001.

[ART06] Dimitris Achlioptas and Federico Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. In *STOC*, pages 130–139. ACM, 2006.

[AS00] Dimitris Achlioptas and Gregory B. Sorkin. Optimal myopic algorithms for random 3-sat. In *FOCS*, pages 590–600, 2000.

[BBC+01] Béla Bollobás, Christian Borgs, Jennifer T. Chayes, Jeong Han Kim, and David B. Wilson. The scaling window of the 2-SAT transition. *Random Structures Algorithms*, 18(3):201–256, 2001.

[BFU93] Andrei Z. Broder, Alan M. Frieze, and Eli Upfal. On the satisfiability

and maximum satisfiability of random 3-CNF formulas. In *SODA*, pages 322–330, 1993.

[BKPS02] Paul W. Beame, Richard Karp, Toniann Pitassi, and Michael Saks. The efficiency of resolution and Davis-Putnam procedures. *SIAM J. Comput.*, 31(4):1048–1075 (electronic), 2002.

[BMW00] Giulio Biroli, Remi Monasson, and Martin Weigt. A variational description of the ground state structure in random satisfiability problems. *Eur. Phys. J. B*, 14:551568, 2000.

[BSI99] Eli Ben-Sasson and Russell Impagliazzo. Random CNF's are hard for the polynomial calculus. In *FOCS*, pages 415–421, 1999.

[CDA+03] Cristian Coarfa, Demetrios D. Demopoulos, Alfonso San Miguel Aguirre, Devika Subramanian, and Moshe Y. Vardi. Random 3-SAT: The plot thickens. *Constraints*, 8(3):243–261, 2003.

[CF86] Ming-Te Chao and John Franco. Probabilistic analysis of two heuristics for the 3-satisfiability problem. *SIAM J. Comput.*, 15(4):1106–1118, 1986.

[CF90] Ming-Te Chao and John Franco. Probabilistic analysis of a generalization of the unit-clause literal selection heuristics for the k-satisfiability problem. *Inform. Sci.*, 51(3):289–314, 1990.

[CGHS04] Don Coppersmith, David Gamarnik, Mohammad Taghi Hajiaghayi, and Gregory B. Sorkin. Random max sat, random max cut, and their phase transitions. *Random Struct. Algorithms*, 24(4):502–545, 2004.

[CM97] Stephen A. Cook and David G. Mitchell. Finding hard instances of the satisfiability problem: a survey. In *Satisfiability problem: theory and applications (Piscataway, NJ, 1996)*, volume 35 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 1–17. 1997.

[CM04] Simona Cocco and Rémi Monasson. Heuristic average-case analysis of the backtrack resolution of random 3-satisfiability instances. *Theoret. Comput. Sci.*, 320(2-3):345–372, 2004.

[CM05] Simona Cocco and Rémi Monasson. Restarts and exponential acceleration of the Davis-Putnam-Loveland-Logemann algorithm: a large deviation analysis of the generalized unit clause heuristic for random 3-SAT. *Ann. Math. Artif. Intell.*, 43(1-4):153–172, 2005.

[CMMS03] Simona Cocco, Rémi Monasson, Andrea Montanari, and Guilhem Semerjian. Approximate analysis of search algorithms with "physical" methods. *CoRR*, cs.CC/0302003, 2003.

[COGL07] Amin Coja-Oghlan, Andreas Goerdt, and André Lanka. Strong refutation heuristics for random k-SAT. *Combin. Probab. Comput.*, 16(1):5–28, 2007.

[COGLS04] Amin Coja-Oghlan, Andreas Goerdt, André Lanka, and Frank Schädlich. Techniques from combinatorial approximation algorithms yield efficient algorithms for random $2k$-SAT. *Theoret. Comput. Sci.*, 329(1-3):1–45, 2004.

[COMV07] Amin Coja-Oghlan, Elchanan Mossel, and Dan Vilenchik. A spectral approach to analyzing belief propagation for 3-coloring. *CoRR*, abs/0712.0171, 2007.

[CR92] Vasek Chvátal and B. Reed. Mick gets some (the odds are on his side). In *FOCS*, pages 620–627, 1992.

[CS88] Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *J. Assoc. Comput. Mach.*, 35(4):759–768, 1988.

[DB97] Olivier Dubois and Yacine Boufkhad. A general upper bound for the satisfiability threshold of random r-SAT formulae. *J. Algorithms*, 24(2):395–420, 1997.

[DBM03] Olivier Dubois, Yacine Boufkhad, and Jacques Mandler. Typical random 3-sat formulae and the satisfiability threshold. *Electronic Colloquium on Computational Complexity (ECCC)*, 10(007), 2003.

[FdlV92] Wenceslas Fernandez de la Vega. On random 2-SAT. 1992. Unpublished Manuscript.

[Fei02] Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC*, pages 534–543, 2002.

[FGK05] Joel Friedman, Andreas Goerdt, and Michael Krivelevich. Recognizing more unsatisfiable random k-SAT instances efficiently. *SIAM J. Comput.*, 35(2):408–430 (electronic), 2005.

[FMV06] Uriel Feige, Elchanan Mossel, and Dan Vilenchik. Complete convergence of message passing algorithms for some satisfiability problems. In *APPROX-RANDOM*, pages 339–350, 2006.

[FP83] John Franco and Marvin Paull. Probabilistic analysis of the Davis–Putnam procedure for solving the satisfiability problem. *Discrete Appl. Math.*, 5(1):77–87, 1983.

[Fra01] John Franco. Results related to threshold phenomena research in satisfiability: lower bounds. *Theoret. Comput. Sci.*, 265(1-2):147–157, 2001.

[Fri99] Ehud Friedgut. Sharp thresholds of graph properties, and the k-SAT problem. *J. Amer. Math. Soc.*, 12:1017–1054, 1999.

[FS96] Alan Frieze and Stephen Suen. Analysis of two simple heuristics on a random instance of k-SAT. *J. Algorithms*, 20(2):312–355, 1996.

[GL03] Andreas Goerdt and André Lanka. Recognizing more random unsatisfiable 3-SAT instances efficiently. In *Typical case complexity and phase transitions*, volume 16 of *Electron. Notes Discrete Math.*, page 26 pp. (electronic). Elsevier, Amsterdam, 2003.

[GM07] Antoine Gerschenfeld and Andrea Montanari. Reconstruction for models on random graphs. In *FOCS*, pages 194–204, 2007.

[Goe96] Andreas Goerdt. A threshold for unsatisfiability. *J. Comput. System Sci.*, 53(3):469–486, 1996.

[Gol79] Allen Goldberg. On the complexity of the satisfiability problem. In *4th Workshop on Automated Deduction (Austin, TX, 1979)*, pages 1–6, 1979.

[GSCK00] Carla P. Gomes, Bart Selman, Nuno Crato, and Henry Kautz. Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. *J. Automat. Reason.*, 24(1-2):67–100, 2000.

[HS03] Mohammad Taghi Hajiaghayi and Gregory B. Sorkin. The satisfiability threshold of random 3-SAT is at least 3.52. volume RC22942 of *IBM Research Report*. 2003.

[KKKS98] Lefteris M. Kirousis, Evangelos Kranakis, Danny Krizanc, and Yiannis Stamatiou. Approximating the unsatisfiability threshold of random formulas. *Random Structures Algorithms*, 12(3):253–269, 1998.

[KKL02] Alexis C. Kaporis, Lefteris M. Kirousis, and Efthimios G. Lalas. The probabilistic analysis of a greedy satisfiability algorithm. In *Algorithms—ESA 2002*, volume 2461 of *Lecture Notes in Comput. Sci.*, pages 574–585. Springer, Berlin, 2002.

[KKL06] Alexis C. Kaporis, Lefteris M. Kirousis, and Efthimios G. Lalas. The probabilistic analysis of a greedy satisfiability algorithm. *Random Structures Algorithms*, 28(4):444–480, 2006.

[KMRT+07] Florent Krząkała, Andrea Montanari, Federico Ricci-Tersenghi, Guilhem Semerjian, and Lenka Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proc. Natl. Acad. Sci. USA*, 104(25):10318–10323 (electronic), 2007.

[KS94] Scott Kirkpatrick and Bart Selman. Critical behavior in the satisfiability of random Boolean expressions. *Science*, 264(5163):1297–1301, 1994.

[LMS98] Michael G. Luby, Michael Mitzenmacher, and M. Amin Shokrollahi. Analysis of random processes via And-Or tree evaluation. In *SODA*, pages 364–373, 1998.

[MMZ06] Stephan Mertens, Marc Mézard, and Riccardo Zecchina. Threshold values of random K-SAT from the cavity method. *Random Structures Algorithms*, 28(3):340–373, 2006.

[Mol05] Michael S. O. Molloy. Cores in random hypergraphs and Boolean formulas. *Random Structures Algorithms*, 27(1):124–135, 2005.

[Mon05] Rémi Monasson. A generating function method for the average-case analysis of DPLL. In *Approximation, randomization and combinatorial optimization*, volume 3624 of *Lecture Notes in Comput. Sci.*, pages 402–413. Springer, Berlin, 2005.

[MPR02] Marc Mézard, Giorgio Parisi, and Zecchina Ricardo. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297:812 – 815, 2002.

[MRTS07] Andrea Montanari, Federico Ricci-Tersenghi, and Guilhem Semerjian. Solving constraint satisfaction problems through belief propagation-guided decimation. *CoRR*, abs/0709.1667, 2007.

[MSL92] David G. Mitchell, Bart Selman, and Hector J. Levesque. Hard and easy distributions of sat problems. In *AAAI*, pages 459–465, 1992.

[MZ98] Rémi Monasson and Riccardo Zecchina. Tricritical points in random combinatorics: the $(2 + p)$-SAT case. *J. Phys. A: Math. and Gen.*, 31(46):9209–9217, 1998.

[MZK+] Rémi Monasson, Riccardo Zecchina, Scott Kirkpatrick, Bart Selman, and Lidror Troyansky. Phase transition and search cost in the $(2+p)$-SAT problem. In *4th Workshop on Physics and Computation, (Boston, MA, 1996)*.

[MZK+99a] Rémi Monasson, Riccardo Zecchina, Scott Kirkpatrick, Bart Selman, and Lidror Troyansky. $2 + p$-SAT: relation of typical-case complexity to the nature of the phase transition. *Random Struc-*

tures Algorithms, 15(3-4):414–435, 1999. Statistical physics methods in discrete probability, combinatorics, and theoretical computer science (Princeton, NJ, 1997).

[MZK+99b] Rémi Monasson, Riccardo Zecchina, Scott Kirkpatrick, Bart Selman, and Lidror Troyansky. Determining computational complexity from characteristic "phase transitions". *Nature*, 400(6740):133–137, 1999.

[Pap91] Christos H. Papadimitriou. On selecting a satisfying truth assignment (extended abstract). In *FOCS*, pages 163–169. IEEE, 1991.

[SAO05] Sakari Seitz, Mikko Alava, and Pekka Orponen. Focused local search for random 3-satisfiability. *Journal of Statistical Mechanics: Theory and Experiment*, 6:6–+, June 2005.

[SM03] Guilhem Semerjian and Rémi Monasson. A study of pure random walk on random satisfiability problems with "physical" methods. In *SAT*, pages 120–134, 2003.

[SML96] Bart Selman, David G. Mitchell, and Hector J. Levesque. Generating hard satisfiability problems. *Artificial Intelligence*, 81(1-2):17–29, 1996.