Chapter 1
On the Satisfiability and Maximum Satisfiability of Random 3-CNF Formulas

Andrei Z. Broder* Alan M. Frieze† Eli Upfal‡

Abstract

We analyze the *pure literal rule* heuristic for computing a satisfying assignment to a random 3-CNF formula with n variables. We show that the pure literal rule by itself finds satisfying assignments for almost all 3-CNF formulas with up to $1.63n$ clauses, but it fails for more than $1.7n$ clauses.

As an aside we show that the value of *maximum satisfiability* for random 3-CNF formulas is tightly concentrated around its mean.

1 Introduction

Given a boolean formula ω in conjunctive normal form, the *satisfiability problem* (SAT) is to determine whether there is a truth assignment that satisfies ω. Since SAT is NP-complete, one is interested in efficient heuristics that perform well "on average," or with high probability. The choice of the probabilistic space is crucial for the significance of such a study. In particular, it is easy to decide SAT in probabilistic spaces that generate formulas with large clauses [16]. To circumvent this problem, recent studies have focused on formulas with exactly k literals per clause (the k-SAT problem). Of particular interest is the case $k = 3$, since this is the minimal k for which the problem is NP-complete.

Consider the space $\Omega^{(3)}_{m,n}$ of all m clause formulas over n variables with exactly 3 literals per clause. It is clear that if the ratio $c = m/n$ is small then a random formula is almost surely satisfiable. (As a trivial example, if $m/n = o(\sqrt{n})$, then with high probability no variable occurs twice.) Experimental evidence [18, 19] strongly suggests that there exists a threshold γ, such that formulas are almost surely satisfiable for $c < \gamma$ and almost surely unsatisfiable for $c > \gamma$, where γ is about 4.2 . So far however, only much weaker bounds were proven and it is not known whether a sharp threshold really exists. Such a threshold (namely c=1) exists for 2-CNF formulas [15, 5].

Chao and Franco [3] and Chvatal and Reed [5] analyzed heuristics that almost surely find satisfying assignments for $\omega \in \Omega^{(3)}_{m,n}$ with $m \leq n$, thus proving that $c = 1$ is a lower bound for the maximum value of c that guarantees almost sure satisfiability in $\Omega^{(3)}_{m,n}$. Very recently, Frieze and Suen [14] have increased this lower bound to ≈ 3.003.

A simple counting argument [7] shows that if c exceeds a constant greater than $\log_{8/7} 2 = 5.190\ldots$ then a formula in $\Omega^{(3)}_{m,n}$ is almost surely unsatisfiable. This bound is not optimal; a minuscule improvement (to about $\log_{8/7} 2 - 10^{-7}$) will be presented in the final paper.

Most practical algorithms for the satisfiability problem (such as the well-known Davis-Putnam algorithm [6]) work iteratively. At each iteration, the algorithm selects a literal and assigns to it the value 1. All clauses containing this literal are erased from the formula, and the complement of the chosen literal is erased from the remaining clauses. Algorithms differ in the way they select the literal for each iteration. The following three rules are the most common ones:

1. *The unit clause rule:* If a clause contains only one literal, that literal must have the value 1;
2. *The pure literal rule:* If a formula contains a literal but does not contain its complement, this literal is assigned the value 1;
3. *The smallest clause rule:* If there is no unit clause or a pure literal, give value 1 to a (random) variable in a smallest clause.

Previous analyses of algorithms for random SAT instances avoided the pure literal rule, and considered only the unit clause rule, or a combination of the unit clause rule and the smallest clause rule. The reason is that if one starts with a random formula and applies the unit clause rule and/or the smallest clause rule, the distribution of literals in the remaining formula is random and uniform, conditional only on the number

*DEC Systems Research Center, 130 Lytton Ave, Palo Alto, CA 94301. E-mail: broder@src.dec.com

†Department of Mathematics, Carnegie-Mellon University. A portion of this work was done while the author was visiting DEC SRC. Supported in part by NSF grant CCR9024935. E-mail: af1p@euler.math.cmu.edu

‡IBM Almaden Research Center, San Jose, CA 95120, and Department of Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel. E-mail: ely@almaden.ibm.com

and size of the remaining clauses. This property, which greatly simplifies the analysis, does not hold when the pure literal rule is applied since in the remaining formula there is a dependency between the occurrence of a literal and the occurrence of its complement.

In this paper we present the analysis of an algorithm based on the pure literal rule. We show that in the $\Omega^{(3)}_{m,n}$ probabilistic space, the pure literal rule alone is sufficient to find, with high probability, a satisfying assignment for a random formula $\omega \in \Omega^{(3)}_{m,n}$, for $m/n \leq 1.63$. On the other hand, if $m/n > 1.7$, then the pure literal rule by itself does not suffice. The gap between 1.63 and 1.7 is not a "real gap". It seems that by increased computation we can make the gap as small as we like, although we do not at present have a rigorous proof that there is a precise threshold.

Maximum satisfiability (MAX-SAT) is the optimization version of the satisfiability problem. Given a CNF formula ω the goal is to determine the maximum number of clauses in ω that can be simultaneously satisfied. This problem arises often in database and expert-systems applications. The decision version of MAX-SAT is NP-complete; however, MAX-SAT can be approximated within a constant ratio. In particular if all clauses contain at least 3 literals, MAX-SAT can be approximated within a 7/8 factor, since one can always find a truth assignment that satisfies at least 7/8 of the clauses in a formula with at least 3 literals per clause [20].

We prove a concentration phenomenon for MAX-SAT: we show that there is a function $T(m,k)$ such that if $\omega \in \Omega^{(k)}_{m,n}$, with high probability the difference between MAX-SAT(ω) and $T(m,k)$ is $o(T(m,k))$. For large m (e.g. $n = o(m)$) clearly $T(m,k) = (1-2^{-k})m$. For smaller values of m we prove tight concentration using a martingale technique, but could not determine the actual value. This result shows that the approximation problem for maximum satisfiability is in a certain sense trivial in this probabilistic setting since the maximum satisfiability value is almost always very close to a fixed value that depends only on m and k.

2 Definitions and Notations

Throughout this paper formulas are represented in conjunctive normal form. Let $V = \{x_1, ..., x_n\}$ be a set of n variables. A *literal* is a variable x_i or its negation $\bar{x}_i$. The set of all literals is denoted L. A *clause* is a disjunction of literals, a formula is a conjunction of clauses.

Let ω be a formula over the set V of variables. A truth assignment for ω is a function $t : L \to \{0,1\}$, such that $t(\bar{x}) = 1 - t(x)$. A truth assignment satisfies ω if at least one literal in each clause of ω is assigned the value 1.

Our analysis is done in the probabilistic space $\Omega^{(3)}_{m,n}$, the space of all formulas over n variables with m clauses and exactly 3 literals per clause. To avoid irrelevant intricacies we view the formula as an ordered list of clauses, and each clause as an ordered list of literals. A random formula $\omega \in \Omega^{(3)}_{m,n}$ is generated by choosing each of the $3m$ literals in ω uniformly at random from the $2n$ possible literals.

Call a literal z (resp. $\bar{z}$) *pure* in a formula ω, if it appears in ω but $\bar{z}$ (resp. z) does not. We also refer to the associated variable as being pure.

Let $\Theta_{m,n,p}$ denote the set of 3-CNF formulas in which there are m clauses that contain n variables out of which p are pure. (Note that $\Theta_{m,n,p}$ is not simply $\Omega^{(3)}_{m,n}$ conditional on p: a formula in $\Theta_{m,n,p}$ must actually contain n variables, while a formula in $\Omega^{(3)}_{m,n}$ might contain fewer.)

We say that a property holds *with high probability* (w.h.p.) if it holds with probability $1 - o(1)$ as $n \to \infty$ and *quite surely* (q.s.) if the $o(1)$ term is $O(n^{-a})$ for any constant a. (The latter terminology is borrowed from [10])

3 Algorithm

The algorithm that we analyze consists of the repeated simultaneous elimination of all clauses containing pure literals. More formally the algorithm can be described as follows:

ALGORITHM 3.1.

while ω contains pure literals **do**
  Let $\pi = \{$All pure literals in $\omega\}$.
  Assign 1 to all literals in π.
  Remove from ω all clauses containing a literal from π.
od
if $\omega = \emptyset$ **then Success else Failure**.

4 Analysis of the algorithm

We first observe that if the algorithm fails on an instance ω then it will also fail on an instance $\hat{\omega}$ obtained by adding an extra clause to ω. Hence, for a fixed n, the probability that the algorithm succeeds decreases as m increases.

4.1 Maintenance of uniformity.

LEMMA 4.1. *Suppose ω is chosen uniformly from $\Theta_{m,n,p}$ and all clauses containing pure variables are deleted. Let $\omega' \in \Theta_{m',n',p'}$ be the formula that remains. Then conditional on the values of m', n', and p', the formula ω' is equally likely to be any formula in $\Theta_{m',n',p'}$.*

Proof. Fix $\omega' \in \Theta_{m',n',p'}$. We only need to show that the number of formulas $\omega \in \Theta_{m,n,p}$ which map onto ω' by the deletion process depends only on n, m, p, n', m', and p' and not on the particular ω'.

Assume that the variables in $\Theta_{m,n,p}$ are x_j for $j \in N$ and those in $\Theta_{m',n',p'}$ are x_j for $j \in N' \subseteq N$. We can construct all the ω that map to ω' as follows:

1. Choose a set $\pi \subseteq N \setminus N'$ of size p. Let $G = N \setminus (N' \cup \pi)$, thus $|G| = n - n' - p$.

2. Assign a label x_j or $\bar{x}_j$ for each $j \in \pi$. Call these the π-literals.

3. Make up $m - m'$ clauses using the variables x_j for $j \in N$ such that
   - Each clause contains at least one π-literal;
   - Each π-literal occurs at least once;
   - For each $j \in G$, both x_j and $\bar{x}_j$ appear at least once;
   - If x_j is pure in ω' (there are exactly p' such literals) then its complementary literal must appear at least once.

4. Insert the new clauses somewhere among the old.

Finally observe that the number of sets of clauses satisfying 1–4 above depends only on n, m, p, n', m', and p'. □

LEMMA 4.2. *Suppose that ω is chosen uniformly from $\Omega^{(3)}_{m,n}$. Let n' be the number of variables that actually appear in ω and let p' be the number of pure variables in ω. Then conditional on the values of n' and p', the formula ω' is equally likely to be any formula in $\Theta_{m,n',p'}$.*

Proof. Obvious. □

We conclude that during the entire execution of the algorithm, conditional on the current values of m, n, and p, the formula w is uniformly distributed over $\Theta_{m,n,p}$.

4.2 The results of one iteration.

We will first state a local central limit theorem which will be used a number of times in the paper. It is a special case of Theorem 4.5.2 of Durrett [9].

THEOREM 4.1. *Let $Z_1, Z_2, \ldots, Z_n$ be non-negative i.i.d. integer valued random variables with $\mathbf{E}(Z_1) = \mu$, $\mathbf{Var}(Z_1) = \sigma^2 \in (0, \infty)$, and $\mathbf{Pr}(Z_1 = k) > 0$ for all non-negative integers k. Let $S_n = Z_1 + Z_2 + \cdots + Z_n$.*

Let $a = a(n)$ be a positive integer, and define x by $x = (a - n\mu)/(\sigma\sqrt{n})$. Further let $p_n(x) = \mathbf{Pr}(S_n = a)$. Then

$$|n^{1/2} p_n(x) - \phi(x)| \to 0, \qquad \text{as } n \to \infty,$$

where $\phi(x) = (2\pi)^{-1/2} e^{-x^2/2}$ is the density of the standard normal distribution.

Let $\lambda = 3m/(2n-p)$. Then λ is the average number of occurrences of a literal in $\omega \in \Theta_{m,n,p}$. Define $\hat{\lambda}$ by $\lambda = \hat{\lambda}/(1 - e^{-\hat{\lambda}})$. (This is well defined since the RHS increases from -1 to ∞ with $\hat{\lambda} \geq 0$.)

The distribution Z given by

$$\mathbf{Pr}(Z = k) = \frac{\hat{\lambda}^k e^{-\hat{\lambda}}}{(1 - e^{-\hat{\lambda}})k!} = \frac{\hat{\lambda}^k}{(e^{\hat{\lambda}} - 1)k!}, \qquad k \geq 1,$$

is called a *truncated* Poisson distribution. Note that $\mathbf{E}(Z) = \lambda$.

Let now $\vec{X} = (X_1, X_2, \ldots, X_N)$ denote the number of occurrences of the $N = 2n - p$ literals of the formula ω chosen uniformly at random from $\Theta_{m,n,p}$. Let $\vec{Y} = (Y_1, Y_2, \ldots, Y_N)$ denote N independent random variables with distribution Z. As before, let $M = 3m$.

LEMMA 4.3.

(a) The variables $X_1, X_2, \ldots, X_N$ are jointly distributed as $Y_1, Y_2, \ldots, Y_N$ conditional on $\sum_{1 \leq j \leq N} Y_j = M$.

(b) $\mathbf{Pr}\left(\sum_{1 \leq j \leq N} Y_j = M\right) = \Omega(1/\sqrt{N})$.

Proof. Let

$$A = \left\{ \vec{x} \in [N]^M \,\middle|\, \sum_{1 \leq j \leq N} x_j = M \text{ and } \forall j, x_j \geq 1 \right\}.$$

Fix $\vec{\xi} \in A$. Then

$$\mathbf{Pr}(\vec{X} = \vec{\xi}) = \left(\frac{M!}{\xi_1! \xi_2! \ldots \xi_N!}\right) \Big/ \left(\sum_{\vec{x} \in A} \frac{M!}{x_1! x_2! \ldots x_N!}\right),$$

and

$$\begin{aligned}\mathbf{Pr}\big(\vec{Y}=\vec{\xi}\mid \textstyle\sum_{1\le j\le N} Y_j = M\big) &= \left(\prod_{1\le j\le N}\frac{\hat\lambda^{\xi_j}}{(e^{\hat\lambda}-1)\xi_j!}\right)\Big/\left(\sum_{\vec{x}\in A}\prod_{1\le j\le N}\frac{\hat\lambda^{x_j}}{(e^{\hat\lambda}-1)x_j!}\right)\\ &= \left(\frac{(e^{-\hat\lambda}-1)^{-N}\hat\lambda^{M}}{\xi_1!\xi_2!\ldots\xi_N!}\right)\Big/\left(\sum_{\vec{x}\in A}\frac{(e^{-\hat\lambda}-1)^{-N}\hat\lambda^{M}}{x_1!x_2!\ldots x_N!}\right)\end{aligned}$$

and (a) follows. To prove (b), apply Theorem 4.1 with $Z_j = Y_j$ for $j = 1,\ldots,N$ and $x = 0$. □

For the remainder of the paper, we fix an arbitrary constant δ, such that $1/2 < \delta < 1$.

THEOREM 4.2. *Suppose that ω is chosen uniformly from $\Theta_{m,n,p}$ and all clauses containing pure variables are deleted. Assume that $m, p \ge n^\delta$. Let $\omega' \in \Theta_{m',n',p'}$ be the formula that remains. Then quite surely*

$$\begin{aligned}\left|m' - m(1-\alpha)^3\right| &= O(n^\delta)\\ \left|n' - (n-p)(1-\beta^2)\right| &= O(n^\delta)\\ \left|p' - 2(n-p)\beta(1-\beta)\right| &= O(n^\delta)\end{aligned}$$

where

$$\alpha = \frac{p}{2n-p},$$

(α is the probability that a random literal in ω is pure), and

$$\beta = \frac{1}{e^{-\hat\lambda}-1}\big(\exp((2\alpha-\alpha^2)\hat\lambda)-1\big),$$

(β is approximately the probability that a fixed literal appears only in clauses that are deleted).

The value $\hat\lambda$ above is defined as before by

$$\lambda = \frac{\hat\lambda}{1-e^{-\hat\lambda}},$$

where $\lambda = 3m/(2n-p)$ is the average number of occurrences of a literal in ω.

Proof. Let X_i denote the number of occurrences of the i'th literal contained in ω. We start by analyzing the number of pure literals in ω. Assume that the first p literals correspond to pure variables. Let $D_p = X_1 + X_2 + \cdots + X_p$. Thus $\mathbf{E}(D_p) = \lambda p$.

CLAIM 4.1.

$$|D_p - \lambda p| < n^\delta \qquad q.s.$$

Proof. Define the random variable $\hat D_p = Y_1 + Y_2 + \cdots + Y_p$, where $Y_1, Y_2, \ldots, Y_p$ are as in Lemma 4.3. Then part (b) of this lemma implies that

$$\begin{aligned}&\mathbf{Pr}\big(|D_p - \lambda p| \ge n^\delta\big)\\ &\qquad = O(n^{1/2})\,\mathbf{Pr}\big(|\hat D_p - \lambda p| \ge n^\delta\big).\end{aligned} \tag{4.1}$$

(Fix v, condition on $D_p = v$, and apply the Lemma to $X_1, \ldots, X_p$.)

Now let $\tilde Y_i = \min\{Y_i, \ln n\}$ and let $\tilde D_p = \tilde Y_1 + \tilde Y_2 + \cdots + \tilde Y_p$. Then

$$\begin{aligned}&\mathbf{Pr}(\exists i : \tilde Y_i \ne Y_i)\\ &\qquad \le \exp\big(-(1-o(1))\ln n \ln\ln n\big).\end{aligned} \tag{4.2}$$

Note that this implies $\mathbf{E}(\tilde D_p) = \lambda p + O(n^{-10})$, say.

Since $\tilde Y_1, \tilde Y_2, \ldots, \tilde Y_p$ are independent, *bounded* random variables, we can apply Hoeffding's Theorem [17] to show

$$\begin{aligned}&\mathbf{Pr}\big(|\tilde D_p - \mathbf{E}(\tilde D_p)| \ge n^\delta/2\big)\\ &\qquad \le 2\exp\left(-\frac{2n^{2\delta}}{4p(\ln n)^2}\right).\end{aligned} \tag{4.3}$$

Combining (4.1), (4.2), and (4.3) completes the proof of Claim 4.1. □

Returning to the proof of the theorem we now consider the number of clauses left in ω'. Fix D_p. The probability that a certain clause does not contain any pure variables is precisely

$$\begin{aligned}\frac{3m-D_p}{3m}\cdot\frac{3m-D_p-1}{3m-1}\cdot\frac{3m-D_p-2}{3m-2} &= \left(1-\frac{D_p}{3m}\right)^3\big(1+O(m^{-1})\big).\end{aligned}$$

Hence using Claim 4.1 we see that

$$\begin{aligned}\mathbf{E}(m') &= \mathbf{E}\left(m\left(1-\frac{D_p}{3m}\right)^3\big(1+O(m^{-1})\big)\right)\\ &= m(1-\alpha)^3 + O(n^\delta),\end{aligned}$$

where

$$\alpha = \frac{\lambda p}{3m} = \frac{p}{2n-p}.$$

We also need to show that m' is concentrated around its mean. This can be easily derived via the use of martingale tail inequalities. To do so, fix $X_1, X_2, \ldots, X_{2n-p}$, the number of occurrences of each literal. Now consider some random permutation

$\phi_1, \phi_2, \ldots, \phi_{3m}$ of the $3m$ literals. Now interchanging a pair ϕ_i, ϕ_j can change m' by at most one. Hence (see e.g. Alon and Spencer [1]),

$$\mathbf{Pr}(|m' - \mathbf{E}(m')| \geq t \mid X_1, X_2, \ldots, X_{2n-p}) \leq 2\exp\left(-\frac{2t^2}{3m}\right).$$

Putting $t = n^\delta$ and removing the conditioning shows that

$$|m' - \mathbf{E}(m')| < n^\delta \qquad \text{q.s.}$$

Let us now consider n' and p'. The same use of the martingale argument above shows that both are sufficiently concentrated around their means. Thus we need only to estimate $\mathbf{E}(n')$ and $\mathbf{E}(p')$.

Again fix the number of occurrences of each literal. Consider a fixed non-pure variable, x_j say. Suppose that the literal x_j occurs $k \geq 1$ times, and that $\bar{x}_j$ occurs $l \geq 1$ times. Now throw the literals randomly into $3m$ slots corresponding to the literals of ω as follows: (a) throw the $k+l$ literals x_j and $\bar{x}_j$; (b) fill the clauses containing them with other literals; (c) fill the other clauses.

With probability $1 - O((k+l)^2/m)$ no two of the $k+l$ literals end up in the same clause. Assuming this, the probability r that the variable x_j does not make it to the next round satisfies

$$(2\alpha_{\min} - \alpha_{\min}^2)^{k+l} \leq r \leq (2\alpha_{\max} - \alpha_{\max}^2)^{k+l}$$

where

$$\alpha_{\min} = \frac{D_p - 2(k+l)}{3m - (k+l)}, \qquad \alpha_{\max} = \frac{D_p}{3m - 3(k+l)}.$$

(During part (b) of the construction there are at most $3m-(k+l)$ and at least $3m-3(k+l)$ literals not yet used, out of which at most D_p and at least $D_p - 2(k+l)$ are pure.) Thus assuming $k, l \leq \ln n$ (see Equation (4.2)) we conclude that

$$r = \left(\frac{2D_p}{3m} - \left(\frac{D_p}{3m}\right)^2\right)\left(1 + O\left(\frac{k+l}{D_p}\right)\right).$$

Let $\nu_{k,l}$ denote the number of j's such that $X_{x_j} = k$, and $X_{\bar{x}_j} = l$. Then

$$\begin{aligned}\mathbf{E}(n-p-n') &= \mathbf{E}\Bigg(\sum_{1\leq k,l\leq \ln n} \nu_{k,l}\left(\frac{2D_p}{3m} - \left(\frac{D_p}{3m}\right)^2\right)^{k+l} \\ &\qquad + O\left(\frac{n(\ln n)^2}{D_p}\right)\Bigg) \\ &= \mathbf{E}\Bigg(\sum_{1\leq k,l\leq \ln n} \nu_{k,l}(2\alpha - \alpha^2)^{k+l}\Bigg) + O(n^\delta).\end{aligned}$$

By martingale arguments again, we can show that for all $k, l \leq \ln n$, almost surely

$$\nu_{k,l} = (n-p)\frac{\hat{\lambda}^{k+l}}{(e^{\hat{\lambda}} - 1)^2 k!l!} + O(n^\delta).$$

Putting

$$\beta = \frac{1}{e^{-\hat{\lambda}} - 1}\big(\exp((2\alpha - \alpha^2)\hat{\lambda}) - 1\big),$$

we see that

$$\mathbf{E}(n - p - n') = (n-p)\beta^2 + O(n^\delta).$$

Note that β is approximately the probability that a fixed literal appears only in clauses that are deleted. A similar argument to the above yields the (intuitively reasonable) fact that

$$\mathbf{E}(p') = 2(n-p)\beta(1-\beta) + O(n^\delta).$$

□

4.3 The first iteration. The first iteration of the algorithm is different since we start with a random $\omega \in \Omega^{(3)}_{m,n}$.

THEOREM 4.3. *Suppose that ω is chosen uniformly from $\Omega^{(3)}_{m,n}$. Let n' be the number of variables that actually appear in ω and let p' be the number of pure variables in ω. Then q.s.*

$$n' = n\big(1 - \exp(-3m/n)\big) + O(n^\delta)$$
$$p' = 2n\exp(-3m/(2n))\big(1 - \exp(-3m/(2n))\big) + O(n^\delta)$$

Proof. Use the martingale argument. □

4.4 A sufficient condition for success.

LEMMA 4.4. *Let ω be a random formula in $\Omega^{(3)}_{m,n}$, and let $c = m/n$. With high probability every subset of $n/(600c^2)$ clauses in ω has at least one pure literal with respect to itself.*

Proof. If a certain subset of k clauses does not have a pure literal with respect to itself, then its $3k$ literals are all chosen from among a set of less than $3k/2$ variables. The probability that there exists a subset of k clauses in ω such that all its $3k$ literals belong to a set of $\ell < 3k/2$ variables is less than

$$P = \sum_{\ell \leq 3k/2} \binom{m}{k}\binom{n}{\ell}\left(\frac{\ell}{n}\right)^{3k}.$$

Since $\ell = 3k/2$ gives the largest term in the sum,

$$P \le \frac{3k}{2}\left(\frac{ecn}{k}\right)^k \left(\frac{2en}{3k}\right)^{\frac{3k}{2}} \left(\frac{3k}{2n}\right)^{3k}$$
$$= \frac{3k}{2}\left(ce^{5/2}\left(\frac{3}{2}\right)^{3/2}\left(\frac{k}{n}\right)^{1/2}\right)^k = o(1),$$

for $k \le n/(600c^2)$. □

Hence if the algorithm starts with cn clauses, and at some point during its execution the number of clauses remaining becomes less than $n/(600c^2)$ then the algorithm will succeed (w.h.p.), since from that point on the Lemma above promises that the algorithm will not run out of pure literals.

4.5 Putting everything together. In this subsection we show that if the algorithm starts with ω drawn from $\Omega^{(3)}_{cn,n}$, it almost surely finds a satisfying assignment, if $c \le 1.63$. The idea of the proof is to use Theorem 4.3 once, and then Theorem 4.2 repeatedly, to show that after a *fixed, finite* number of iterations, with high probability the number of clauses left in ω is less than $n/(600c^2)$, after which by Lemma 4.4, the algorithm almost surely does not fail.

Lemmas 4.1 and 4.2 ensures that the uniformity conditions required by Theorem 4.2 are satisfied. However there are two potential stumbling blocks:

- In principle, at the start of each application of Theorem 4.2 the values m, n, and p are known only within a $1+o(1)$ factor. Nevertheless it can be shown that if we use such approximate values, the values predicted for m', n', and p' still are almost surely within a $1+o(1)$ factor of the actual values. Since the number of iterations is finite, this suffices to prove that the final values are accurate within a a $1+o(1)$ factor.

- In practice, what we have at the start of each application of Theorem 4.2 are the *numeric* estimates for m, n, and p. (More precisely numeric estimates of the ratios m/n_0, n/n_0, and p/n_0, where n_0 is the initial n.) Since we use finite precision, we need to worry about the cumulative round-off error. Again since the number of applications is small (say < 100) if we use enough precision (say 40 digits) then we can guarantee that the final results are correct to, say, 10 digits, and Lemma 4.4 can be applied.

The full details of the proof, which include the complete error analysis, are left for the final paper.

The battle plan above when applied to $m = 1.63n$ results in the values presented in Figure 1, that is, we apply Theorem 4.3 once and Theorem 4.2, iteratively 77 times to conclude that after 78 iterations, almost surely the number of clauses left is $(.0000148 \pm 10^{-7})n(1 + o(1))$. Since this is less than $n/(600 \cdot 1.63^2) \approx .000627n$, by Lemma 4.4 the algorithm will almost surely succeed in this case. (The actual computations were done by MAPLE [4]) with 30 digits of accuracy, which ensure more than 7 digits in the final result.

We can probably prove a slightly better bound than 1.63 at the expense of even more iterations, but for $m > 1.7n$ the algorithm is almost certain to fail – the proof of this is given in the next section.

Iter.	$m/n_0 \approx$	$n/n_0 \approx$	$p/n_0 \approx$
0	1.6300000	.9924785	.1584094
1	1.2416257	.8321861	.0754947
2	1.0729162	.7559570	.0456785
3	.9757320	.7099215	.0311313
4	.9115719	.6785915	.0228269
⋮	⋮	⋮	⋮
16	.6839027	.5604441	.0039727
17	.6766309	.5564652	.0036935
18	.6698942	.5527661	.0034551
⋮	⋮	⋮	⋮
30	.6114513	.5201327	.0022309
31	.6075173	.5178994	.0022046
32	.6036380	.5156923	.0021859
⋮	⋮	⋮	⋮
44	.5567591	.4886338	.0025312
45	.5524330	.4860991	.0026174
46	.5479711	.4834778	.0027162
⋮	⋮	⋮	⋮
58	.4731273	.4383696	.0058926
59	.4635879	.4324521	.0065074
60	.4531244	.4259134	.0072370
⋮	⋮	⋮	⋮
73	.0918368	.1403240	.0492052
74	.0448331	.0806850	.0407996
75	.0129840	.0284797	.0198463
76	.0013075	.0034452	.0030089
77	.0000148	.0000434	.0000422

Figure 1: Repeated applications of Theorem 1 for $m = 1.63n$.

Simulation experiments show excellent concordance with these values even for moderate values of n. Details will be given in the final paper.

4.6 An upper bound on the performance of the algorithm. We show in this section that our analysis of the algorithm is close to optimal. For formulas with more than $1.7n$ random clauses the algorithm almost always fails.

THEOREM 4.4. *Let $\omega \in \Omega^{(3)}_{m,n}$, with $m \geq 1.7n$. Then with high probability the algorithm fails to find a satisfying assignment for ω.*

Proof. (Outline) Without loss of generality we can assume that $m/n = 1.7$. Let n_i, m_i, p_i denote the number of variables, clauses and pure literals at the end of iteration i. Let λ_i, $\hat{\lambda}_i$, α_i, and β_i denote the associated values of λ, $\hat{\lambda}$, α, and β (See Theorem 4.2.)

Suppose that α_k gets small. Then simple estimations give

$$\alpha_{k+1} = \beta_k \leq \frac{2\hat{\lambda}_k}{e^{\hat{\lambda}_k} - 1}\alpha_k + \frac{4\hat{\lambda}_k^2}{e^{\hat{\lambda}_k} - 1}\alpha_k^2. \tag{4.4}$$

If simultaneously $\hat{\lambda}_k$ is reasonably large, so that $2\hat{\lambda}_k < e^{\hat{\lambda}_k} - 1$ then we can expect α_i to tend to zero. More precisely, fix $\epsilon > 0$ and let $\xi = \xi_\epsilon$, and $\eta = \eta_\epsilon$ satisfy

$$\frac{2\xi}{e^{\xi} - 1} = 1 - 2\epsilon \quad \text{and} \quad \frac{\xi}{1 - e^{-\xi}} = \eta.$$

Then

$$\lambda_k \geq \eta \quad \text{implies} \quad \frac{2\hat{\lambda}_k}{e^{\hat{\lambda}_k} - 1} \leq 1 - 2\epsilon.$$

Suppose that after a (bounded) number of iterations we reach a stage r where quite surely

$$\alpha_r \leq \epsilon^2/2 \quad \text{and} \quad \lambda_r \geq \frac{\eta}{1 - \epsilon}.$$

(When $m = 1.7n$ and $\epsilon = 10^{-4}/2$ we find that Theorems 4.2 and 4.3 imply that this happens at $r = 20$.) Calculations, using Theorem 4.2 and (4.4) then show that as long as $m_i, n_i, p_i \geq n^\delta$ and $i > r$ then quite surely

$$\begin{aligned} \lambda_i &\geq \eta \\ \alpha_{i+1} &\leq (1 - \epsilon)\alpha_i \\ n_i &\geq n_r(1 - \epsilon) - O(n^\delta) \\ m_i &\geq m_r(1 - \epsilon)^3 - O(n^\delta) \end{aligned} \tag{4.5}$$

Thus (q.s.) there exists a $s > r$, with $s = O(\ln n)$ and a constant $\gamma > 0$ such that

$$m_s, n_s \geq \gamma n \quad \text{and} \quad p_s \leq n^\delta.$$

We complete the proof by showing that for $i \geq s$

$$\mathbf{E}(p_{i+1}) \leq (1 - \epsilon/2)\mathbf{E}(p_i) \tag{4.6}$$

$$m_{i+1} \geq m_i - O(p_i \ln n) \tag{4.7}$$

$$n_{i+1} \geq n_i - O(p_i \ln n) \tag{4.8}$$

Inequality (4.6) shows that (w.h.p.) there exists a $t \geq s$, $t = O(\ln n)$ such that $p_t = 0$ and then (4.7) shows that $m_t \geq m_r - O(n^\delta(\ln n)^2) > 0$ and thus the algorithm has failed.

The proof of (4.7) and (4.8) is immediate since quite surely no pure variable appears in the formula more than $\ln n$ times.

To prove (4.6) fix $i \geq s$ and let $\rho = p_i$, $\nu = n_i$, and $\mu = m_i$. Let $Y_1, Y_2, \ldots, Y_\nu$ be as in Lemma 4.3. Assume that the first ρ literals are pure and condition on $X_1 + \cdots + X_\rho = Y_1 + \cdots + Y_\rho = D$. Consider the probability that the complement of the $z_{\rho+1}$ (the $\rho + 1$ literal) becomes pure. The number of occurrences of $z_{\rho+1}$ is $X_{\rho+1}$ which is of course distributed as $Y_{\rho+1}$. We obtain that

$$\begin{aligned} \mathbf{E}(p_{i+1} \mid D) \leq (2\nu - \rho) &\sum_{k=1}^{\mu} \mathbf{Pr}(Y_{\rho+1} = k \mid D) \\ &\times \prod_{j=0}^{k-1} \frac{2(D - j)}{3\mu - 2j - k} \end{aligned} \tag{4.9}$$

Applying Theorem 4.1 twice we obtain that

$$\begin{aligned} &\mathbf{Pr}(Y_{\rho+1} = k \mid D) \\ &= \frac{\mathbf{Pr}(Y_{\rho+1} = k \wedge Y_{\rho+2} + \cdots + Y_\nu = 3\mu - D - k)}{\mathbf{Pr}(Y_{\rho+1} + \cdots + Y_\nu = 3\mu - D)} \\ &= \frac{\mathbf{Pr}(Y_{\rho+1} = k)\,\mathbf{Pr}(Y_{\rho+2} + \cdots + Y_\nu = 3\mu - D - k)}{\mathbf{Pr}(Y_{\rho+1} + \cdots + Y_\nu = 3\mu - D)} \\ &= \frac{\hat{\lambda}_i^k}{(e^{\hat{\lambda}_i} - 1)k!}\left(1 + O(n^{\delta-1})\right) \end{aligned}$$

But quite surely $k \leq \ln n$ and $|D - \lambda\rho| \leq \sqrt{\lambda\rho}\ln n$. Values outside of these ranges make insignificant contributions to the expectation in (4.9) and so we assume that k, D are within these ranges. Thus

$$\begin{aligned} \mathbf{E}(p_{i+1} \mid D) &\leq \frac{2\nu - \rho}{e^{\hat{\lambda}_i} - 1}\left(\exp\left(\frac{2\hat{\lambda}_i D}{3\mu - 3\ln n}\right) - 1\right) \\ &\quad \times \left(1 + O(n^{\delta-1})\right) \\ &\leq \frac{2\hat{\lambda}}{e^{\hat{\lambda}} - 1} \cdot \frac{2\nu - \rho}{3\mu} \cdot D \cdot \left(1 + O(n^{\delta-1})\right). \end{aligned}$$

Removing the conditioning, and substituting $\lambda_i = 3\mu/(2\nu-\rho)$, we get

$$\begin{aligned}\mathbf{E}(p_{i+1}) &\le \frac{2\hat{\lambda}_i}{e^{\hat{\lambda}_i}-1}\cdot\frac{\mathbf{E}(D)}{\lambda_i}\left(1+O(n^{\delta-1})\right)\\ &= \frac{2\hat{\lambda}_i}{e^{\hat{\lambda}_i}-1}\rho\left(1+O(n^{\delta-1})\right)\end{aligned}$$

and (4.6) follows since $\lambda_i \ge \eta$ (see (4.5)). This completes our outline proof. □

5 Concentration of maximum satisfiability

Given a formula ω, let $M(\omega)$ denote the maximum number of clauses in ω that can be simultaneously satisfied. Let $T(m,k) = \mathbf{E}(M(\omega))$ for $\omega \in \Omega^{(k)}_{m,n}$. We prove that $M(\omega)$ is tightly concentrated around $T(m,k)$.

Let $X_0(\sigma), X_2(\sigma), \ldots, X_m(\sigma)$ be a sequence of random variables (the "Doob martingale") defined by

$$X_i(\sigma) = \mathbf{E}(M(\omega)\ |\omega \in \Omega \text{ and the first } i \text{ clauses in } \sigma \text{ and } \omega \text{ are identical}).$$

Clearly $X_0 = \mathbf{E}(M) = T(m,k)$ and $X_m(\sigma) = M(\sigma)$. Also $\mathbf{E}(X_{i+1} \mid X_i) = X_i$, which is the martingale condition. Since $|X_{i+1}(\sigma) - X_i(\sigma)| \le 1$, we can use Azuma's inequality to prove:

THEOREM 5.1. *Let $\omega \in \Omega^{(k)}_{m,n}$ and let $T(m,k) = E(M(\omega))$. Then*

$$\mathbf{Pr}\left(|M(\omega) - T(m,k)| > \sqrt{2m\log m}\right) \le 1/m.$$

The martingale technique only shows that the value of $M(\omega)$ is almost sure close to it expectation. It does not specify the expectation. For n linear in m computing the expectation is an open problem; for $n = o(m)$ a straightforward calculation shows that $M(\omega)$ is almost always close to $(1-2^{-k})m$.

Acknowledgement

We wish to thank Moshe Vardi for introducing us to this problem, and for several discussions and references.

Appendices

A The MAPLE program

Below is a straightforward MAPLE program used to compute the table in Figures 1. (This is not meant as an example of Maple programming. The terser original was modified for ease of readability.) The program maintains three global variables, `mu`, `nu`, `pi`, that represent respectively the ratios m/n_0, n/n_0, and p/n_0.

```
    Digits := 40;

start := proc(c)
# Apply Theorem 2 to compute the initial values
    mu := c;
    nu := 1 - exp(-3*c);
    pi:= 2*exp(-3*c/2)*(1 - exp(-3*c/2));
    print(mu, nu, pi);
end;

rec := proc()
# Compute current alpha, lambda, lambdah, and beta
    alpha := pi/(2*nu-pi);
    lambda := 3*mu/(2*nu-pi);
    lambdah := fsolve(lambda=x/(1-exp(-x)),x,
                      fulldigits);
    beta := 1/(exp(lambdah)-1)
          * (exp((2*alpha - alpha^2)*lambdah) -1);
# Save the old values
    muold := mu; nuold := nu; piold := pi;
# Apply Theorem 1 to compute new values
    mu := muold*(1-alpha)^3;
    nu := (nuold - pi)*(1-beta^2);
    pi := 2*(nuold-piold)*beta*(1-beta);
    print(mu, nu, pi);
end;
```

The program used to generate Figure 1 was:

```
start(1.63); for j to 80 do rec() od;
```

B Simulation results

Below is a run of a simulation using 100000 variables and 163000 random clauses. Notice that the results are very close to the predictions made in Figure 1. If we average over several runs, then the numbers are even closer. But our proof shows, and the experiments confirm, that almost surely *for every run* the results are very close to expected values.

Iter.	m/n_0	n/n_0	p/n_0	Clauses
1	1.630000	0.992950	0.160770	123687
2	1.236870	0.830180	0.075240	107101
3	1.071010	0.754250	0.044230	97790
4	0.977900	0.709650	0.029890	91552
5	0.915520	0.679620	0.022700	86984
6	0.869840	0.656850	0.017190	83584
7	0.835840	0.639620	0.014290	80789
8	0.807890	0.625270	0.011750	78517
9	0.785170	0.613500	0.009810	76675
10	0.766750	0.603680	0.007780	75153

Iter.	m/n_0	n/n_0	p/n_0	Clauses
11	0.751530	0.595860	0.006540	73906
12	0.739060	0.589300	0.005580	72867
13	0.728670	0.583700	0.004970	71928
14	0.719280	0.578730	0.004340	71099
15	0.710990	0.574390	0.004220	70331
16	0.703310	0.570170	0.003860	69610
⋮	⋮	⋮	⋮	⋮
30	0.639180	0.535120	0.001710	63597
31	0.635970	0.533400	0.001810	63298
32	0.632980	0.531590	0.001620	63019
⋮	⋮	⋮	⋮	⋮
44	0.604350	0.515220	0.001090	60247
45	0.602470	0.514130	0.001140	60056
46	0.600560	0.512990	0.001020	59876
⋮	⋮	⋮	⋮	⋮
58	0.584220	0.503390	0.000710	58288
59	0.582880	0.502680	0.000820	58154
60	0.581540	0.501850	0.000830	58003
⋮	⋮	⋮	⋮	⋮
72	0.561520	0.490420	0.000960	55991
73	0.559910	0.489460	0.000970	55817
74	0.558170	0.488490	0.001070	55633
⋮	⋮	⋮	⋮	⋮
86	0.530090	0.471920	0.001870	52718
87	0.527180	0.470050	0.001850	52404
88	0.524040	0.468200	0.002060	52060
⋮	⋮	⋮	⋮	⋮
100	0.463400	0.431280	0.005060	45540
101	0.455400	0.426210	0.005620	44661
102	0.446610	0.420560	0.005480	43791
⋮	⋮	⋮	⋮	⋮
112	0.273310	0.303100	0.028370	23548
113	0.235480	0.273780	0.034130	19243
114	0.192430	0.237610	0.040080	14397
115	0.143970	0.193970	0.046370	9311
116	0.093110	0.140980	0.047670	4749
117	0.047490	0.083960	0.040810	1492
118	0.014920	0.031800	0.021140	195
119	0.001950	0.005010	0.004330	6
120	0.000060	0.000180	0.000180	0

References

[1] N. Alon and J. Spencer. The probabilistic method. John Wiley and Sons, 1992.

[2] M.T. Chao and J. Franco. Probabilistic analysis of two heuristics for the 3-satisfiability problem. *SIAM J. Comput.* 15 (1986) pp. 1106–1118.

[3] M.T. Chao and J. Franco. Probabilistic analysis of a generalization of the unit-clause literal selection heuristics for the k satisfiable problem. *Information Science* 51 (1990) pp. 289–314.

[4] B.W. Char *et al.* Maple V Language Reference Manual. Springer-Verlag, 1991.

[5] V. Chvátal and B. Reed. Mick gets some (the odds are on his side). *IEEE Symp. on Foundation of Computer Science* 33 (1992) pp. 620–627.

[6] M. Davis and H. Putman. A computing procedure for quantification theory. *J. ACM* 7 (1960) pp. 201–215.

[7] O. Dubois. Counting the number of solutions for instances of satisfiability. *Theoretical Computer Science* 81 (1991) pp. 49–64.

[8] O. Dubois and J. Carlier. Probabilistic approach to the satisfiability problem. *Theoretical Computer Science* 81 (1991) pp. 65–75.

[9] R. Durrett. Probability: Theory and Examples. Wadsworth & Brooks/Cole 1991.

[10] P. Flajolet, D.E. Knuth and B. Pittel. The first cycles in an evolving graph. *Discrete Mathematics* 75 (1989) pp. 167–215.

[11] J. Franco Probabilistic analysis of the pure literal heuristic for solving the satisfiability problem. *Annals of Operations Research* 1 (1984) pp. 273–289.

[12] J. Franco and Y.C. Ho. Probabilistic performance of heuristic for the satisfiability problem. *Discrete Applied Mathematics* 22 (1988/89) pp. 35–51.

[13] J. Franco and M. Paull. Probabilistic analysis of the Davis Putman procedure for solving the satisfiability problem. *Discrete Applied Mathematics* 5 (1983) pp. 77–87.

[14] A.M. Frieze and S. Suen. Analysis of simple heuristics for random instances of 3-SAT. In preparation.

[15] A. Goerdt. A threshold for unsatisfiability. To appear in *17th International Symposium on Mathematical Foundations of Computer Science*, Prague, Czechoslovakia, August 1992.

[16] A. Goldberg. Average case complexity of the satisfiability problem. In *Proc. 4th Workshop on Automated Deduction*, 1979, pp. 1–6.

[17] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association* 58 (1963), pp. 13–30.

[18] T. Larrabee and Y. Tsuji. Evidence for Satisfiability Threshold for Random 3CNF Formulas. Technical Report UCSC-CRL-92-42, University of California Santa Cruz, 1992.

[19] D. Mitchell, B. Selman, and H. Levesque. Hard and Easy Distributions of SAT problems. In *Proc. 10th National Conference on Artificial Intelligence*, 1992, pp. 459–465.

[20] M. Yannakakis. On the approximation of maximum satisfiability. *IEEE Symp. on Foundation of Computer Science* 33 (1992).