Phase transition for Local Search on planted SAT

Andrei A. Bulatov[1] Evgeny S. Skvortsov[2]

[1] Simon Fraser University
[2] Google Canada
abulatov@cs.sfu.ca, evgenys@google.com

Abstract. The Local Search algorithm (or Hill Climbing, or Iterative Improvement) is one of the simplest heuristics to solve the Satisfiability and Max-Satisfiability problems. Although it is not the best known Satisfiability algorithm even for the class of problems we study, the Local search is a part of many satisfiability and max-satisfiability solvers, where it is used to find a good starting point for a more sophisticated heuristics, and to improve a candidate solution. In this paper we give an analysis of Local Search on random planted 3-CNF formulas. We show that a sharp transition of efficiency of Local Search occurs at density $\varrho = \frac{7}{6}\ln n$. Specifically we show that if there is $\kappa < \frac{7}{6}$ such that the clause-to-variable ratio is less than $\kappa \ln n$ (n is the number of variables in a CNF) then Local Search whp does not find a satisfying assignment, and if there is $\kappa > \frac{7}{6}$ such that the clause-to-variable ratio is greater than $\kappa \ln n$ then the local search whp finds a satisfying assignment. As a byproduct we also show that for any constant ϱ there is γ such that Local Search applied to a random (not necessarily planted) 3-CNF with clause-to-variable ratio ϱ produces an assignment that satisfies at least γn clauses less than the maximal number of satisfiable clauses.

The k-SAT and MAX-k-SAT problems are known to be hard both to solve exactly and to approximate. In particular, Håstad [17] proved that the MAX-k-SAT problem is NP-hard to approximate within ratio better than $1 - 2^{-k}$. These worst case hardness results motivate the study of the typical case complexity of those problems, and a quest for probabilistic or heuristic algorithms with satisfactory performance, in the typical case. In this paper we analyze the performance of one of the simplest algorithms for (MAX-)k-SAT, the Local Search algorithm, on random planted instances.

The distribution. One of the most natural and well studied probability distributions of 3-CNFs is the uniform distribution $\Phi(n, \varrho n)$ on the set of 3-CNFs with a given clauses-to-variables ratio. For a given number of variables n and *density* $\varrho = \varrho(n)$ the formula is constructed and sampled as follows. The formula contains $m(n) = \varrho n$ 3-clauses over variables $x_1, \ldots, x_n$. These clauses are chosen uniformly at random from the set of all possible clauses. So the probability of every 3-CNF from $\Phi(n, \varrho n)$ is the same.

The typical case complexity for this distribution is not very complex except for a limited range of densities. The reason is that the random 3-SAT under this distribution demonstrates a sharp satisfiability threshold in the density [2, 12]. A random 3-CNF with density below the threshold (estimated to be around 4.2) is satisfiable whp (with high probability, meaning that the probability tends to 1 as n goes to infinity), and a 3-CNF with density above the threshold is unsatisfiable whp. Therefore the trivial algorithm outputting yes or no by just counting the density of a 3-CNF gives a right answer to 3-SAT whp. For more results on the threshold see [11, 1, 10]. It is also known that, as the density grows, the number of clauses satisfied by a random assignment differs less and less from the maximal number of satisfiable clauses. If the density is *infinite* (i.e. it is an unbounded function of n), then whp this difference becomes negligible, i.e. $o(n)$.

An interesting and useful distribution $\Phi^{\mathsf{sat}}(n, \varrho n)$ is obtained from $\Phi(n, \varrho n)$ by conditioning on satisfiability: such distribution is uniform and its elements are the satisfiable 3-CNFs. Then the problem is to find or approximate a satisfying assignment knowing it exists. If a satisfiable formula has sufficiently high density, the set of satisfying assignments of such formula whp satisfies strong restrictions on its structure, which makes it possible to find a solution in polynomial time [9], see also [24]. In some aspects, however, the satisfiable distribution is not very convenient, for instance, it is not clear how to sample the satisfiable distribution (see, e.g., [6]). An approximation for such a distribution is the planted distribution $\Phi^{\mathsf{plant}}(n, \varrho n)$, which is obtained from $\Phi(n, \varrho n)$ by conditioning on satisfiability by a specific "planted" assignment. To construct an element of a planted distribution we select an assignment of a set of n variables and then uniformly at random include ϱn clauses satisfied by the assignment selected. Coja-Oghlan et al. [9] argued that when density is sufficiently high satisfiable formulas whp have satisfying assignments tightly clustered, and consequently the planted distribution is statistically close to the satisfiable one. They also managed to transfer certain results for $\Phi^{\mathsf{plant}}(n, \varrho n)$ to $\Phi^{\mathsf{sat}}(n, \varrho n)$.

Another interesting feature of the planted distribution (and the satisfiable one, see [9]) is that there is a hope to design an algorithm that solves all planted instances whp [6, 14, 19]. Algorithms from [14] and [19] use different approaches to solve planted 3-SAT of high but constant density. Note that if the density of formulas is low, properties of the planted and satisfiable distributions may be significantly different. Due to the phase transition phenomenon, almost all 3-CNFs of density below the threshold are satisfiable, while finding a solution of such a formula seems to be hard. On the other hand, experiments show that the algorithm from [5] achieves the goal, but a rigorous analysis of this algorithm is not yet made. For a survey on SAT algorithms the reader is referred to [7].

The algorithm. The Local Search algorithm (LS) is one of the oldest heuristics for SAT that has been around since the eighties. Numerous variations of LS have been proposed since then, see, e.g., [22]. We study one of the most basic versions of LS, which, given a CNF, starts with a random assignment to its variables, and then on each step chooses at random a variable such that flipping this variable increases the number of satisfied clauses, or stops if no such variable exists. Thus LS finds a random local optimum accessible from the initial assignment.

LS has been studied before. The worst-case performance of pure LS appears to be not very good: the only lower bound for local optima of a k-CNF is $\frac{k}{k+1}m$ of clauses satisfied, where m is the number of all clauses [16]. Local Search was proven to solve 3-SAT in the typical case by Koutsoupias and Papadimitriou [18] for formulas of linear density, that is, $m = \Omega(n^2)$. Finally, in [8], we gave an estimation of the dependence of the number of clauses LS typically satisfies and the density of the formula.

Often visualization of the number of clauses satisfied by an assignment is useful: Assignments can be thought of as points of a landscape, and the elevation of a point corresponds to the number of clauses unsatisfied, the higher the point is, the less clauses it satisfies. It is suspected that 'topographic' properties of such a landscape are responsible for many complexity properties of satisfiability instances [7, 20]. As we shall see the performance of LS is closely related to geometric properties of the assignments.

The behavior of other SAT/MAXSAT algorithms has been studied as well. In some cases the algorithms in question are basic heuristics used as a part or prototypes of practical Satisfiability solvers. For example, the random walk has been analyzed in [21] and then in [3], and the performance of GSAT, an improved version of LS, has been studied in [23]. A number of algorithms have been carefully examined to show that they solve whp a certain class of SAT problems. These include, e.g., [6, 14, 19, 9, 24, 5] for the planted and satisfiable distributions. Message passing type algorithm such as Survey Propagation are known be very efficient for solving 3-SAT of density close to the threshold [7]. Although this algorithm still eludes a rigorous analysis, a similar message passing algorithm, Warning Propagation, is studied in [13].

Our contribution. We classify the performance of LS for all densities higher than an arbitrary constant. In particular, we demonstrate that LS has a sharp threshold (see [15]) in its performance. The main result is the following theorem.

Theorem 1. *(1) Let $\varrho \geq \kappa \cdot \ln n$, and $\kappa > \frac{7}{6}$. Then LS whp finds a solution of an instance from $\Phi^{\mathsf{plant}}(n, \varrho n)$.*

(2) Let $\sigma \leq \varrho \leq \kappa \cdot \ln n$, where σ is an arbitrary positive constant, and $0 < \kappa < \frac{7}{6}$. Then LS whp does not find a solution of an instance from $\Phi^{\mathsf{plant}}(n, \varrho n)$.

To prove part (1) of Theorem 1 we show that under those conditions all the local optima of a 3-CNF whp are either satisfying assignments, that is, global optima, or are far from the planted solution, and so are located on the opposite side of the set of assignments. In the former case LS finds a satisfying assignment, while whp it does not reach the local optima of the second type.

For part (2) we prove that if the density is insufficient, the formula whp contains sufficiently many local minima so that LS inevitably falls into one of them, rather than proceeding into the planted global minimum. We also show that for any constant density ϱ there is γ such that the assignment produced by

LS on an instance from $\Phi^{\mathsf{plant}}(n, \varrho n)$ or $\Phi(n, \varrho n)$ satisfies at least γn clauses less than the maximal number of satisfiable clauses.

Another region where LS can find a solution of a random planted 3-CNF is the case of very low density. Methods similar to Lemma 3 and Theorem 2 show that this low density transition happens around $\varrho \approx n^{-1/4}$. However, we do not go into details here.

1 Preliminaries

SAT. A 3-CNF is a conjunction of *3-clauses*. As we consider only 3-CNFs, we will always call them just clauses. Depending on the number of negated literals, we distinguish 4 types of clauses: $(-,-,-), (+,-,-)$, $(+,+,-)$, and $(+,+,+)$. If φ is a 3-CNF over variables $x_1, \dots, x_n$, an *assignment* of these variables is a Boolean n-tuple $\boldsymbol{u} = (u_1, \dots, u_n)$, so the value of x_i is u_i. The *density* of a 3-CNF φ is the number $\frac{m}{n}$ where m is the number of clauses, and n is the number of variables in φ.

The *uniform* distribution of 3-CNFs of density ϱ (density may be a function of n), $\Phi(n, \varrho n)$, is the set of all 3-CNFs containing n variables and ϱn clauses equipped with the uniform probability distribution on this set. To sample a 3-CNF accordingly to $\Phi(n, \varrho n)$ one chooses uniformly and independently ϱn clauses out of the $2^3 \begin{pmatrix} n \\ 3 \end{pmatrix}$ possible clauses. Thus, we allow repetitions of clauses, but not repetitions of variables within a clause. *Random 3-SAT* is the problem of deciding the satisfiability of a 3-CNF randomly sampled accordingly to $\Phi(n, \varrho n)$. For short, we will call such a random formula a 3-CNF from $\Phi(n, \varrho n)$.

The *uniform planted* distribution of 3-CNFs of density ϱ is constructed as follows. First, choose at random a Boolean n-tuple $\boldsymbol{u}$, a *planted* satisfying assignment. Then let $\Phi^{\mathsf{plant}}(n, \varrho n, \boldsymbol{u})$ be the uniform probability distribution over the set of all 3-CNFs over variables $x_1, \dots, x_n$ with density ϱ and such that $\boldsymbol{u}$ is a satisfying assignment. For our goals we can always assume that $\boldsymbol{u}$ is the all-ones tuple, that is a 3-CNF belongs to $\Phi^{\mathsf{plant}}(n, \varrho n, \boldsymbol{u})$ if and only if it contains no clauses of the type $(-,-,-)$. We also simplify the notation $\Phi^{\mathsf{plant}}(n, \varrho n, \boldsymbol{u})$ by $\Phi^{\mathsf{plant}}(n, \varrho n)$. To sample a 3-CNF accordingly to $\Phi^{\mathsf{plant}}(n, \varrho n)$ one chooses uniformly and independently ϱn clauses out of $7 \begin{pmatrix} n \\ 3 \end{pmatrix}$ possible clauses of types $(+,-,-), (+,+,-)$, and $(+,+,+)$. *Random Planted 3-SAT* is the problem of deciding the satisfiability of a 3-CNF from $\Phi^{\mathsf{plant}}(n, \varrho n)$.

The problems *Random MAX-3-SAT* and *Random Planted MAX-3-SAT* are the optimization versions of Random 3-SAT and Random Planted 3-SAT. The goal in these problems is to find an assignment that satisfies as many clauses as possible. Although the two problems usually are treated as maximization problems, it will be convenient for us to consider them as problems of minimizing the number of unsatisfied clauses. Since we always evaluate the absolute error of our algorithms, not the relative one, such transformation does not affect the results.

Local search. A description of the Local Search algorithm (LS) is given below.

```
Algorithm 1 Local Search
Require: 3-SAT formula $\varphi$ over variables $x_1, \ldots, x_n$.
Ensure: Boolean $n$-tuple $\boldsymbol{v}$, which is a local minimum of $\varphi$.
1: choose uniformly at random a Boolean $n$-tuple $\boldsymbol{u}$
2: let $U$ be the set of all variables $x_i$ such that the number of clauses that can be
   made satisfied by flipping the value of $x_i$ is strictly greater than the number of
   those made unsatisfied
3: while $U$ is not empty do
4:    pick uniformly at random a variable $x_j$ from $U$
5:    change the value of $x_j$
6:    recompute $U$
7: end while
```

Observe that LS stops when reaches a local minimum of the number of unsatisfied clauses.

Given an assignment $\boldsymbol{u}$ and a clause c it will be convenient to say that c *votes* for a variable x_i to have value 1 if c contains literal x_i and its other two literals are unsatisfied. Similarly, we say that c votes for x_i to have value 0 if c contains the negation of x_i and its other two literals are not satisfied. Using this terminology we can define set U (see Algorithm 1) as the set of all variables such that the number of votes received to change the current value is greater than the number of those to keep it.

Random graphs. Probabilistic tools we use are fairly standard and can be found in the book [4].

Let φ be a 3-CNF with variables $x_1, \ldots, x_n$. The *primal graph* $G(\varphi)$ of φ is the graph with vertex set $\{x_1, \ldots, x_n\}$ and edge set $\{x_i x_j \mid \text{literals containing } x_i, x_j$ appear in the same clause$\}$. Note that $G(\varphi)$ is not a random graph.

We will need the following properties that a graph $G(\varphi)$ has, provided its density is sufficiently low.

Lemma 1. *Let $\varrho < \kappa \ln n$ for a certain constant κ, and let $\varphi \in \Phi^{\mathsf{plant}}(n, \varrho n)$.*

(1) For any $\alpha < 1$, whp all the subgraphs of $G(\varphi)$ induced by at most $O(n^\alpha)$ vertices have the average degree less than 4.

(2) The probability that $G(\varphi)$ has a vertex of degree greater than $\ln^2 n$ is $o(n^{-3})$.

2 Success of Local Search

In this section we prove Theorem 1(1). This will be done as follows. First, we show that if a 3-CNF has high density, that is, greater than $\kappa \log n$ for some $\kappa > \frac{7}{6}$ then whp all the local minima that do not satisfy the CNF — we call

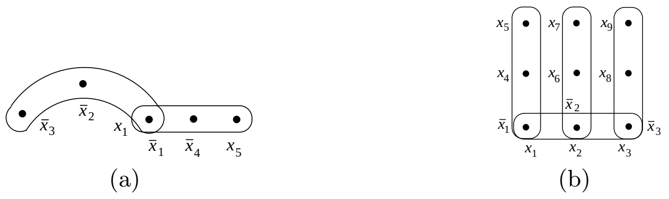

Fig. 1. Caps and crowns

such minima *proper* — concentrate very far from the planted assignment. This is the statement of Proposition 1 below. Then we use Lemma 2 to prove that starting from a random assignment LS whp does not go to that remote region. Therefore the algorithm does not get stuck to a local minimum that is not a solution. Recall that the planted solution is the all-ones one.

Lemma 2. *Let $\varrho \geq \kappa \ln n$ for some constant $\kappa > 0$, and let constants q_0, q_1 be such that $0 \leq q_0 < q_1 \leq 1$. Whp any assignment with less than $q_0 n$ zeros satisfies more clauses than any assignment with more than $q_1 n$ zeros.*

Proposition 1. *Let $\varrho \geq \kappa \cdot \ln n$, and $\kappa > \frac{7}{6}$. Then whp proper local minima of a 3-CNF from $\Phi^{\mathsf{plant}}(n, \varrho n)$ have at most $\frac{n}{10}$ ones.*

Proof (of Theorem 1(1)). By Lemma 2 for a $\varphi \in \Phi^{\mathsf{plant}}(n, \varrho n)$ whp any assignment with dn variables equal to 1, where $\frac{1}{3} \leq d \leq \frac{2}{3}$, satisfies more clauses than any assignment with less than $\frac{n}{10}$ variables equal to 1. Then, whp a random initial assignment for LS assigns between $\frac{1}{3}$ and $\frac{2}{3}$ of all variables to 1. Therefore, whp LS never arrives to a proper local minimum with less than $\frac{n}{10}$ variables equal to 1, and, by Proposition 1, to any proper local minimum.

3 Failure of Local Search

We now prove statement (2) of Theorem 1. The overall strategy is the following. First, we show, Proposition 2, that in contrast to the previous case there are many proper local minima in the close proximity of the planted assignment. Then we show, Proposition 3, that those local minima are located so that they intercept almost every run of LS, and thus almost every run is unsuccessful.

A variable of a CNF is called *k-isolated* if it appears positively in at most k clauses of the type $(+, -, -)$. The *distance* between variables of a CNF φ is the length of the shortest path in $G(\varphi)$ connecting them. A pair of clauses $c_1 = (x_1, \overline{x}_2, \overline{x}_3)$, $c_2 = (\overline{x}_1, \overline{x}_4, x_5)$ is called a *cap* over variables $x_1, \ldots, x_5$ if x_1, x_5 are 1-isolated, that is they do not appear positively in any clause of the type $(+, -, -)$ except for c_1 and c_2, respectively, and x_2, x_3 are not 0-isolated (see Figure 1(a)). If pair c_1, c_2 is a cap, then clause c_2 is called *cap support.*

The proof of the following lemma is fairly standard, see, e.g. the proof of Theorem 4.4.4 in [4].

Lemma 3. *Let $n^{-\frac{1}{4}} < \varrho \le \kappa \cdot \ln n$, and $\kappa < \frac{7}{6}$. There is α, $0 < \alpha < 1$, such that whp a random planted CNF $\varphi \in \Phi^{\mathsf{plant}}(n, \varrho n)$ contains at least n^α caps.*

Proposition 2. *Let $\sigma > 0, 0 < \kappa < \frac{7}{6}$, and density ϱ be such that $\sigma < \varrho \le \kappa \cdot \ln n$. Then there is α, $0 < \alpha \le 1$, such that a 3-CNF from $\Phi^{\mathsf{plant}}(n, \varrho n)$ whp has at least n^α proper local minima.*

Indeed, let $c_1 = (x_1, \overline{x}_2, \overline{x}_3)$, $c_2 = (\overline{x}_1, \overline{x}_4, x_5)$ be a cap and $\boldsymbol{u}$ an assignment such that $u_1 = u_5 = 0$, and $u_i = 1$ otherwise. It is straightforward that $\boldsymbol{u}$ is a proper local minimum because c_1 is the only unsatisfied clause but Local Search does not flip any variables. By Lemma 3, there is α such that whp the number of such minima is at least n^α.

Before proving Theorem 1(2), we note that a construction similar to caps helps evaluate the approximation rate of the local search applied to random and random planted CNFs. A subformula $c = (x_1, x_2, x_3), c_1 = (\overline{x}_1, x_4, x_5), c_2 = (\overline{x}_2, x_6, x_7), c_3 = (\overline{x}_3, x_8, x_9)$ is called a *crown* if the variables $x_1, \dots, x_9$ do not appear in any clauses other than c, c_1, c_2, c_3 (see Fig. 1(b)). The crown is satisfiable, while the all-zero assignment is a proper local minimum. For a CNF φ and an assignment $\boldsymbol{u}$ to its variables, by $\mathsf{OPT}(\varphi)$ and $\mathsf{sat}(\boldsymbol{u})$ we denote the maximal number of simultaneously satisfiable clauses and the number of clauses satisfied by $\boldsymbol{u}$, respectively.

Theorem 2. *If the density ϱ is such that $\sigma \le \varrho \le \kappa \ln n$ for some $\sigma > 0, 0 < \kappa < 1/27$, then there is $\gamma_\varrho(n) > 0$ such that whp Local Search on a 3-CNF $\varphi \in \Phi(n, \varrho n)$ ($\varphi \in \Phi^{\mathsf{plant}}(n, \varrho n)$) returns an assignment $\boldsymbol{u}$ such that $\mathsf{OPT}(\varphi) - \mathsf{sat}(\boldsymbol{u}) \ge \gamma_\varrho \cdot n$. If ϱ is constant then γ_ϱ is also constant.*

Proof of Theorem 2 is similar to that of Lemma 3. It can be shown that for ϱ that satisfies conditions of this theorem there is $\gamma'(n) > 0$ such that whp a random [random planted] formula has at least $\gamma' n$ crowns. If ϱ is a constant, γ' is also a constant. For a random assignment $\boldsymbol{u}$, whp the variables of at least $\frac{\gamma'}{1024} n$ crowns are assigned zeroes. Such an all-zero assignment of a crown cannot be changed by the local search.

Proposition 3. *Let $\sigma > 0, 0 < \kappa < \frac{7}{6}$, and density ϱ be such that $\sigma < \varrho \le \kappa \cdot \ln n$. Then Local Search on a 3-CNF from $\Phi^{\mathsf{plant}}(n, \varrho n)$ whp ends up in a proper local minimum.*

In what follows we prove Proposition 3.

If $\varrho = o(\ln n)$ then Proposition 3 follows from Theorem 2. So we assume that $\varrho > \kappa' \cdot \ln n$. The main tool of proving Proposition 3 is coupling of local search (LS) with the algorithm Straight Descent (SD) that on each step chooses at random a variable assigned to 0 and changes its value to 1. Obviously SD is not a practical algorithm, since to apply it we need to know the solution. For the purposes of our analysis we modify SD as follows. At each step SD chooses a variable at random, and if it is assigned 0 changes its value (see Alg. 2). The algorithm LS is modified in a similar way. It chooses a variable at random and

Algorithm 2 Straight Descent

```
Require: $\varphi \in \Phi^{\mathsf{plant}}(n, \varrho n)$ with the all-ones solution, Boolean tuple $\boldsymbol{u}$,
Ensure: The all-ones Boolean tuple.
1: while there is a variable assigned 0 do
2:     pick uniformly at random variable $x_j$ from the set of all variables
3:     if $u_j = 0$ then
4:         $u_j = 1$
5:     end if
6: end while
```

flips it if it increases the number of satisfied clauses. Modified Local Search is identical to Local Search in the sense that it flips the same sequence of variables.

Since actions of SD do not depend on the formula its position at each step is random, in a sense that under the condition that SD at step t arrived to a vector with m ones, each vector with m ones has the same probability of being the location of SD.

Given 3-CNF φ and an assignment $\boldsymbol{u}$ we say that a variable x_i is *k-righteous* if the number of clauses voting for it to be one is greater by at least k than the number of clauses voting for it to be zero. Let $\varphi \in \Phi^{\mathsf{plant}}(n, \varrho n)$ and $\boldsymbol{u}$ be a Boolean tuple. The *ball* of radius m with the center at $\boldsymbol{u}$ is the set of all tuples of the same length as $\boldsymbol{u}$ at Hamming distance at most m from $\boldsymbol{u}$. Let $f(n)$ and $g(n)$ be arbitrary functions and d be an integer constant. We say that a set S of n-tuples is *$(g(n), d)$-safe*, if for any $\boldsymbol{u} \in S$ the number of variables that are not d-righteous does not exceed $g(n)$. A run of SD is said to be *$(f(n), g(n), d)$-safe* if at each step of this run the ball of radius $f(n)$ with the center at the current assignment is $(g(n), d)$-safe.

Lemma 4. *Let $\varrho > \kappa \cdot \ln n$ for some $\kappa, \kappa > 0$. For any positive constants γ and d there is a constant $\alpha_1 < 1$ such that, for any α with $\alpha_1 < \alpha < 1$, whp a run of SD on $\varphi \in \Phi^{\mathsf{plant}}(n, \varrho n)$ and a random initial assignment $\boldsymbol{u}$ is $(\gamma n^\alpha, n^\alpha, d)$-safe.*

For CNFs ψ_1, ψ_2 we denote by $\psi_1 \wedge \psi_2$ their conjunction.

We will need formulas that are obtained from a random formula by adding some clauses in an ‘adversarial’ manner. Following [19] we call distributions for such formulas *semi-random*. However, the type of semi-random distributions we need is different from that in [19]. Let $\eta < 1$ be some constant. A formula φ is sampled according to semi-random distribution $\Phi_\eta^{\mathsf{plant}}(n, \varrho n)$ if $\varphi = \varphi' \wedge \psi$, where φ' is sampled according to $\Phi^{\mathsf{plant}}(n, \varrho n)$ and ψ contains at most n^η clauses and is given by an adversary. That is, by definition an event E whp takes place for $\varphi = \varphi' \wedge \psi \in \Phi_\eta^{\mathsf{plant}}(n, \varrho n)$ if whp φ' is such that $E(\varphi)$ is true for any ψ. It is not difficult to see that if $\eta' \le \eta$ then any event that occurs whp for $\varphi \in \Phi_\eta^{\mathsf{plant}}(n, \varrho n)$, also occurs whp for $\varphi \in \Phi_{\eta'}^{\mathsf{plant}}(n, \varrho n)$.

We say that a variable *plays d-righteously in a run of LS* if every time it is considered for flipping it is d-righteous. For semi-random distributions we then have the following

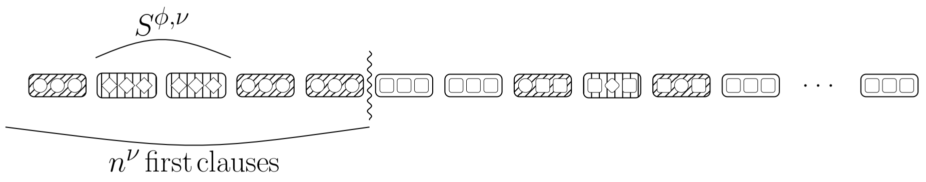

Fig. 2. A scheme of a 3-CNF. Every clause is shown as a rectangle with its literals represented by squares, circles, or diamonds inside the rectangle. Literals corresponding to variables from $L^{\phi\nu}$ and from $\mathrm{var}\left([n^{\nu}] \setminus S^{\varphi,\nu}\right)$ are shown as diamonds and circles, respectively. Shaded rectangles with vertical and diagonal lines represent clauses from $T^{\varphi\nu}$ and $U^{\varphi\nu}$, respectively.

Lemma 5. *Let $\varrho > \kappa \cdot \ln n$ for some $\kappa, \kappa > 0$. For any d there is $\alpha_2 < 1$ such that, for a run of LS on $\varphi \in \Phi_\eta^{\mathtt{plant}}(n, \varrho n)$ whp the number of variables that do not play d-righteously is bounded above by n^{α_2}.*

For a formula ψ we denote the set of variables that occur in it by $\mathrm{var}(\psi)$. For a set of clauses K we denote by $\bigwedge K$ a CNF formula constructed by conjunction of the clauses. For the sake of simplicity we will write $\mathrm{var}(K)$ instead of $\mathrm{var}\left(\bigwedge K\right)$. In what follows it will be convenient to view a CNF as a sequence of clauses. Note that such representation of a CNF is quite natural when we sample a random CNF by generating random clauses. This way every clause occupies certain position in the formula. For a set of positions P we denote the formula obtained from φ by removing all clauses except for occupying positions P by $\varphi\downarrow_P$. The set of variables occurring in the clauses in positions from P will be denoted by $\mathrm{var}(P)$. For a set of variables V the set of positions of clauses which contain a variable from V is denoted by $\mathrm{clp}(V)$. By $[k]$ we denote the set of the first k positions of clauses in φ. Recall that a clause $(\overline{x}, \overline{y}, z)$ is called a *cap support* if there are w_1, w_2 such that $(x, \overline{w}_1, \overline{w}_2), (\overline{x}, \overline{y}, z)$ is a cap in φ.

For a real constant $\nu, 0 < \nu < 1$ we will recognize the following subsets of clauses and variables of φ. Let $S^{\varphi,\nu}$ be the set of positions from $[n^{\nu}]$ occupied by clauses that are cap supports in φ; let $L^{\varphi,\nu} = \mathrm{var}(S^{\varphi,\nu})$; let $T^{\varphi,\nu} = \mathrm{clp}\left(L^{\varphi,\nu}\right)$, $U^{\varphi,\nu} = \mathrm{clp}\left(\mathrm{var}\left([n^{\nu}] \setminus S^{\varphi,\nu}\right)\right)$; finally, let $M^{\varphi,\nu} = \mathrm{var}(T^{\varphi,\nu})$ and $N^{\varphi,\nu} = \mathrm{var}(U^{\varphi,\nu})$. Fig. 2 pictures the notation just introduced.

We denote equality $f(n) = g(n)(1 + o(1))$ by $f(n) \sim g(n)$.

Lemma 6. *Let $\sigma > 0, 0 < \kappa < \frac{7}{6}$, and density ϱ be such that $\sigma < \varrho \le \kappa \cdot \ln n$. Then there is $\mu_0 > 0$ such that for any $\mu < \mu_0$ there is $\nu < 1$ such that whp:*

(1) $|S^{\varphi,\nu}| \sim n^{\mu}$;
(2) $M^{\varphi,\nu} \cap N^{\varphi,\nu} = \varnothing$, i.e. variables from clauses in $U^{\varphi,\nu}$ do not appear in the same clauses with variables from $S^{\varphi,\nu}$;
(3) $|M^{\varphi,\nu}| = 3|T^{\varphi,\nu}|$, that is no variable occurs twice in the clauses from $T^{\varphi,\nu}$.

Let n be the number of variables, let ϱ be the density, let ν be a real constant such that $0 < \nu < 1$, let T_0 and U_0 be subsets of $[\varrho n]$ such that $T_0 \cap U_0 = \varnothing$,

$[n^\nu] \subseteq T_0 \cup U_0$, and let $S_0 = T_0 \cap [n^\nu]$. We denote by $H_{T_0 U_0 \nu}$ a hypothesis stating that φ is such that $S^{\varphi,\nu} = S_0$, $T^{\varphi,\nu} = T_0$, $U^{\varphi,\nu} = U_0$ and also $M^{\varphi,\nu} \cap N^{\varphi,\nu} = \varnothing$, $|M^{\varphi,\nu}| = 3\,|T^{\varphi,\nu}|$. The following lemma allows us to focus on small sets U_0, T_0.

Lemma 7. *Let $\phi \in \Phi^{\mathsf{plant}}(n, \varrho n)$, and $\mu_0, \mu < \mu_0$, and ν be as in Lemma 6; also let $\varepsilon > 0$ be such that $\nu + \varepsilon < 1$. If for an event E there is a sequence $\delta(n) \underset{n \longrightarrow \infty}{\longrightarrow} 0$ such that for all pairs (T_0, U_0), $|T_0 \cup U_0| < n^{\nu+\varepsilon}$ we have $\mathbf{P}(E|H_{T_0 U_0 \nu}) \le \delta(n)$ then $\mathbf{P}(E) \underset{n \longrightarrow \infty}{\longrightarrow} 0$.*

Observation 1 *If φ is selected according to $\Phi^{\mathsf{plant}}(n, \varrho n)$ conditioned to $H_{T_0 U_0 \nu}$ then formula $\varphi \downarrow_{[\varrho n] \setminus (T_0 \cup U_0)}$ follows random planted 3-CNF distribution with $\varrho n - |T_0| - |U_0|$ clauses over variables $V \setminus \mathrm{var}([n^\nu])$.*

Proof (of Proposition 3). For this proof we shall need a small enough μ. Here is how to pick it. We start by applying Lemma 6 to $\varrho \in \Phi^{\mathtt{plant}}(n, n\varrho)$ and get $\tilde\mu$ and $\tilde\nu$ that satisfy conditions (1)-(3) of Lemma 6. Let η be such that $\tilde\nu < \eta < 1$ and α_2 be picked from Lemma 5 for $\Phi_\eta^{\mathtt{plant}}(n, n\varrho)$, so the number of variables that are not playing d-righteously is bounded by n^{α_2}. Now we set $\mu = \min(\tilde\mu, \frac{1-\alpha_2}{4})$ and by Lemma 6 there is $\nu \le \tilde\nu$ which makes conditions (1)-(3) true for μ.

We fix an arbitrary pair (T_0, U_0) of subsets of $[\varrho n]$ with $T_0 \cap U_0 = \varnothing$, $[n^\nu] \subseteq T_0 \cup U_0, |T_0 \cup U_0| < n^\eta$. We will bound the probability of success of LS under a hypothesis of the form $H_{T_0 U_0 \nu}$ and apply Lemma 7 to get the result.

Let $M = M^{\varphi,\nu}$ and $L = L^{\varphi,\nu}$. We split formula φ into $\varphi_1 = \varphi \downarrow_{T_0}$ and $\varphi_2 = \varphi \downarrow_{[\varrho n] \setminus T_0}$ and first consider a run of LS applied to φ_2 only. Formula φ_2 can in turn be considered as the conjunction of $\varphi_{21} = \varphi \downarrow_{U_0}$ and $\varphi_{22} = \varphi \downarrow_{[\varrho n] \setminus (T_0 \cup U_0)}$. In Fig. 2 formula φ_1 consists of clauses shaded with vertical lines, formula φ_{21} of clauses shaded with diagonal lines and formula φ_{22} of clauses that are not shaded. By Observation 1 formula φ_{22} is sampled according to $\Phi^{\mathtt{plant}}(n - \delta_1(n), n\varrho - \delta_2(n))$ modulo the names of variables where $\delta_1(n)$ and $\delta_2(n)$ are $o(n)$. So formula φ_2 is sampled according to $\Phi_\eta^{\mathtt{plant}}(n - \delta_1(n), n\varrho - \delta_2(n))$ and by the choice of α_2 the number of variables that do not play 2-righteously during a run of LS on φ_2 is bounded from above by n^{α_2}.

We consider coupling $(LS_\varphi, LS_{\varphi_2})$ of runs of LS on φ and φ_2. The assignments obtained by the runs of the algorithm at step t we denote by $\boldsymbol{u}_\varphi(t)$ and $\boldsymbol{u}_{\varphi_2}(t)$ respectively. Let K be the set of those variables which do not belong to L (squares and circles in Fig. 2). Formula φ_2 is a 3-CNF containing only variables from K. If $\boldsymbol{u}$ is an assignment of values to all variables, then $\boldsymbol{u}|_K$ will denote the restriction of $\boldsymbol{u}$ onto variables from K. We make process LS_φ start with a random assignment $\boldsymbol{u}_\varphi(0) = \boldsymbol{u}_\varphi^0$ to all variables, and LS_{φ_2} with a random assignment $\boldsymbol{u}_{\varphi_2}(0) = \boldsymbol{u}_{\varphi_2}^0$ to variables in K, such that $\boldsymbol{u}_\varphi^0|_K = \boldsymbol{u}_{\varphi_2}^0$. Now the algorithms work as follows. At every step a random variable x_i is chosen. Process LS_φ makes its step, and process LS_{φ_2} makes its step if $x_i \in K$.

Let W denote the set of variables from $\mathrm{var}(\varphi_2)$ that do not play 2-righteously during the run of LS_{φ_2}. By choice of α_2 whp we have $|W| \le n^{\alpha_2}$. Variables in formula φ_1 are selected uniformly at random and $\alpha_2 + 2\mu < 1$ so whp set M does not intersect with W. Hence, every time LS_φ considers some variable from M

it is 2-righteous in φ_2 and belongs to at most one clause of φ_1. Therefore such a variable is at least 1-righteous in φ and is flipped to 1, or stays 1, whichever is to happen for LS_{φ_2}. Thus whp at every step of $(LS_\varphi, LS_{\varphi_2})$ we have $\boldsymbol{u}_\varphi(t)|_K = \boldsymbol{u}_{\varphi_2}(t)$. In the rest of the proof we consider only this highly probable case.

Consider some cap support $c_i = (\overline{x}_1, \overline{x}_4, x_5)$ occupying a position $i \in [n^\nu]$ and such that $x_1 = 0, x_4 = 1, x_5 = 0$ at time 0, and a set P_{c_i} of variables occurring in clauses that contain variables $\mathrm{var}(c_i)$ (obviously $\mathrm{var}(c_i) \subseteq P_{c_i}$). Let c_j be the clause that forms a cap with c_i. We say that a variable is *discovered* at step t if it is considered for the first time at step t. Let $p_1, \dots, p_k$ be an ordering of elements of P_{c_i} according to the step of their discovery. In other words if variable p_1 is the first variable from P_{c_i} that is discovered, p_k was the last. If some variables are not considered at all, we place them in the end of the list in a random order. Observe that all variables that play at least 1-righteously are discovered at some step. All orderings of variables are equiprobable, hence, the probability of variables $\mathrm{var}(c_i)$ to occupy places p_{k-2}, p_{k-1} and p_k equals $3!/k(k-1)(k-2)$. We will call this ordering *unlucky*.

Let us consider what happens if the order of discovery of P_{c_i} is unlucky. All variables in $P_{c_i} \setminus \mathrm{var}(c_i)$ play 1-righteously, therefore once they are discovered by LS_φ they are equal to 1. Thus when x_1, x_4, x_5 are finally considered all clauses they occur in are satisfied, except for c_j. So variables x_1, x_4, x_5 do not change their values and the clause c_j remains unsatisfied by the end of the work of LS_φ.

By Lemma 1(2) whp no vertex has degree greater than $\ln^2 n$, so the size of the set P_{c_i} is bounded above by $3\ln^2 n$. Thus the probability of event $Unluck(i) =$ "order of discovery of P_{c_i} is unlucky" is greater than $\frac{1}{\ln^6 n}$. Thus, the expectation of $|\{i|Unluck(i)\}|$ equals $\frac{|S_0|}{\ln^6 n} = \frac{n^\mu}{\ln^6 n}$. By definition of $H_{T_0 U_0 \nu}$ any variable whp occurs in clauses from $T^{\varphi,\nu}$ at most once, hence there is no variable that occurs in the same clause with a variable from c_{i_1} and a variable from c_{i_2} for $i_1, i_2 \in S_0$, $i_1 \neq i_2$. This implies that events of the form $Unluck(i)$ are independent. Therefore random variable $|\{i|Unluck(i)\}|$ is Bernoulli and, as its expectation tends to infinity, the probability that it equals to 0 goes to 0. As unlucky ordering of at least one cap support leads to failure of the LS this proves the result.

References

1. Achlioptas, D.: Lower bounds for random 3-SAT via differential equations. Theor. Comput. Sci. 265(1-2), 159–185 (2001)
2. Achlioptas, D., Friedgut, E.: A sharp threshold for k-colorability. Random Struct. Algorithms 14(1), 63–70 (1999)
3. Alekhnovich, M., Ben-Sasson, E.: Linear upper bounds for random walk on small density random 3-CNFs. SIAM J. Comput. 36(5), 1248–1263 (2007)
4. Alon, N., Spencer, J.: The Probabilistic Method. John Wiley (2000)
5. Amiri, E., Skvortsov, E.: Pushing random walk beyond golden ratio. In: Diekert, V., Volkov, M., Voronkov, A. (eds.) CSR. Lecture Notes in Computer Science, vol. 4649, pp. 44–55. Springer (2007)
6. Ben-Sasson, E., Bilu, Y., Gutfreund, D.: Finding a randomly planted assignment in a random 3-CNF. (2002), manuscript

7. Braunstein, A., Mézard, M., Zecchina, R.: Survey propagation: An algorithm for satisfiability. Random Struct. Algorithms 27(2), 201–226 (2005)
8. Bulatov, A., Skvortsov, E.: Efficiency of local search. In: SAT. pp. 297–310 (2006)
9. Coja-Oghlan, A., Krivelevich, M., Vilenchik, D.: Why almost all k-colorable graphs are easy. In: Thomas, W., Weil, P. (eds.) STACS. Lecture Notes in Computer Science, vol. 4393, pp. 121–132. Springer (2007)
10. Coja-Oghlan, A., Panagiotou, K.: Going after the k-SAT threshold. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) STOC. pp. 705–714. ACM (2013)
11. Crawford, J.M., Auton, L.D.: Experimental results on the crossover point in random 3-SAT. Art. Int. 81(1-2), 31–57 (1996)
12. Ding, J., Sly, A., Sun, N.: Proof of the satisfiability conjecture for large k. In: Servedio, R., Rubinfeld, R. (eds.) STOC. pp. 59–68. ACM (2015)
13. Feige, U., Mossel, E., Vilenchik, D.: Complete convergence of message passing algorithms for some satisfiability problems. In: Díaz, J., Jansen, K., Rolim, J.D.P., Zwick, U. (eds.) APPROX-RANDOM. Lecture Notes in Computer Science, vol. 4110, pp. 339–350. Springer (2006)
14. Flaxman, A.: A spectral technique for random satisfiable 3CNF formulas. In: SODA. pp. 357–363. ACM/SIAM (2003)
15. Friedgut, E.: Sharp thresholds of graph properties, and the k-SAT problem. J. Amer. Math. Soc. 12, 1017–1054 (1999)
16. Hansen, P., Jaumard, B.: Algorithms for the maximum satisfiability problem. Computing 44, 279–303 (1990)
17. Håstad, J.: Some optimal inapproximability results. J. ACM 48(4), 798–859 (2001)
18. Koutsoupias, E., Papadimitriou, C.: On the greedy algorithm for satisfiability. Inf. Process. Lett. 43(1), 53–55 (1992)
19. Krivelevich, M., Vilenchik, D.: Solving random satisfiable 3CNF formulas in expected polynomial time. In: SODA. pp. 454–463. ACM Press (2006)
20. Mézard, M., Mora, T., Zecchina, R.: Clustering of solutions in the random satisfiability problem. CoRR abs/cond-mat/0504070 (2005)
21. Papadimitriou, C.: On selecting a satisfying truth assignment (extended abstract). In: FOCS. pp. 163–169. IEEE Computer Society (1991)
22. Selman, B., Levesque, H., Mitchell, D.: A new method for solving hard satisfiability problems. In: Swartout, W. (ed.) AAAI. pp. 440–446. AAAI Press / The MIT Press (1992)
23. Skvortsov, E.: A theoretical analysis of search in GSAT. In: Kullmann, O. (ed.) SAT. Lecture Notes in Computer Science, vol. 5584, pp. 265–275. Springer (2009)
24. Vilenchik, D.: It's all about the support: A new perspective on the satisfiability problem. JSAT 3(3-4), 125–139 (2007)