ON THE GREEDY ALGORITHM FOR SATISFIABILITY

Elias Koutsoupias and Christos H. Papadimitriou

Department of Computer Science and Engineering, UCSD

ABSTRACT: *We show that for the vast majority of satisfiable 3CNF formulae, the local search heuristic that starts at a random truth assignment, and repeatedly flips the variable that improves the number of satisfied clauses the most, almost always succeeds in discovering a satisfying truth assignment.*

Keywords: Analysis of Algorithms, Satisfiability

Consider the following simple heuristic for SATISFIABILITY:

> start with a random truth assignment, call it T;
> **while** there is a truth assignment T' which differs from T in
> one variable, and T' satisfies more clauses than T **do**
> choose the T' that satisfies the most clauses, and set $T := T'$;
> return T.

Naturally, when the formula is unsatisfiable, this heuristic will not return a satisfying truth assignment, and so it will be in some sense "correct." We therefore ask the question: *If the formula is satisfiable,* how often will this heuristic return a satisfying truth assignment? We show that the answer is *almost always!*

Naturally, there is nothing surprising or original about an NP-complete problem with a good average–case algorithm under the most natural probabilistic distribution. What is perhaps a little surprising is that the problem is so fundamental, the algorithm so simple, and the proof so easy.

Let F be a Boolean formula of n variables in conjunctive normal form with three literals per clause (the familiar NP-complete 3SAT problem, Garey and Johnson 1979). Suppose that F is satisfiable, say by a truth assignment $\hat{T}$. We shall show that, for almost all such F, the greedy algorithm will in fact discover $\hat{T}$ with very high probability. The reason is, of course, that most satisfiable formulae have just one satisfying truth assignment. Intuitively then, the greedy algorithm always discovers the satisfying assignment, since the clauses always "point the way." The probabilistic calculation that follows makes this intuition precise.

If the greedy heuristic starts at a truth assignment that agrees with $\hat{T}$ in very few variables, then it is likely that a local optimum that is not a global optimum will be found. However, it is straightforward to show that such starting points are very rare. Let us call a truth assignment *bad* if it agrees with $\hat{T}$ in fewer than $(\frac{1}{2} - \epsilon)n$ variables, for some $\epsilon > 0$; all other assignments are *good.*

Lemma 1. The probability that the initial truth assignment, chosen at random, is bad is at most $e^{-2\epsilon^2 n}$.

Let $B(p,n)$ denote a binomial random variable, i.e. $B(p,n)$ is the number of successes in n independent trials when the probability of success in each trial is p. Lemma 1 follows from the following Chernoff bound:

$$Pr[B(p,n) \le (1-\theta)pn] \le e^{-\theta^2 np/2}$$

Here $p = 1/2$ and $\theta = 2\epsilon$. Let us mention here another Chernoff bound we will need later:

$$Pr[B(p,n) \ge (1+\theta)pn] \le e^{-\theta^2 np/3}$$

By the above lemma, the greedy algorithm usually starts from a good truth assignment. Consider now a good assignment T, that agrees with $\hat{T}$ in exactly k variables, $k \ge (1/2-\epsilon)n$, and one variable, call it a, in which T and $\hat{T}$ disagree. Suppose that we flip a; how many clauses do we gain, and how many do we lose? We denote by A_+ the set of *all clauses* on n variables that are satisfied by $\hat{T}$ (whether they are in F or not) that change from false to true if a is flipped, and by A_- the set of those clauses that change from true to false. We need to calculate the size of A_+ and A_-. Without loss of generality assume that $\hat{T}$ is the constant *true* assingment (flip literals in F if necessary). Then A_+ is the set of clauses that contain the variable a positively and two other variables that occur positively if and only if they are false in T. So, $N_1 = |A_+| = \binom{n-1}{2}$. Similarly A_- is the set of clauses that contain the variable a negatively and two other variables that occur positively if and only if they are false in T. But, we have to exclude the clauses that contain all three variables negatively, because they are not satisfied by $\hat{T}$. So, $N_0 = |A_-| = \binom{n-1}{2} - \binom{k}{2}$.

Notice that by flipping a we do not gain anything if and only if F has at least as many clauses in A_- as in A_+. We are going to find an upper bound of the probability of this to happen. Assuming that each clause that satisfies $\hat{T}$ is in F with probability p, the probability that we do not gain anything is $Pr[B(p,N_0) \ge B'(p,N_1)]$, where B and B' are independent binomial random variables. It is easy to see that this probability is at most $Pr[B(p,N_0) \ge m] + Pr[B(p,N_1) \le m]$, for any m. Using the mentioned Chernoff bounds we get that this probability is at most:

$$e^{-\frac{(m-pN_0)^2}{3pN_0}} + e^{-\frac{(m-pN_1)^2}{2pN_1}} \le e^{-\frac{(m-pN_0)^2}{3pN_0}} + e^{-\frac{(m-pN_1)^2}{3pN_1}}$$

Let $m = p\sqrt{N_0 N_1}$. Then the probability that by flipping a we do not gain anything is at most $2e^{-cpn^2}$, where c depends only on ϵ, i.e. it is constant for fixed ϵ. So, the probability that the greedy algorithm starting from a good assignment will ever be misled by flipping a variable is at most $n2^n e^{-cpn^2}$, since there are at most $n2^{n-1}$ such possible flippings—the number of edges of the n-hypercube. So, we have:

Lemma 2. If all clauses satisfied by $\hat{T}$ are chosen independently at random with probability p, then the probability that the greedy algorithm starting from a good truth assignment does not discover $\hat{T}$ is at most $n2^n e^{-cpn^2}$. •

¿From this, for $p = 1/2$, we have:

Theorem. Let $0 \le \epsilon < \frac{1}{2}$. Then there exists c, depending only on ϵ such that for all but a fraction of at most $n2^n e^{cn^2/2}$ of satisfiable 3CNF Boolean formulae with n variables, the probability that the greedy algorithm succeeds in discovering a truth assignment in each independent trial from a random start is at least $1 - e^{-2\epsilon^2 n}$. $\bullet$

Notice that by Lemma 2, if $p \geq d/n$, for some constant d, the probability that the greedy algorithm will fail is exponentially small. For such p, the formula F has expected number of clauses $\Omega(n^2)$. So, the greedy algorithm performs very well for dense formulae, i.e. for formulae with $\Omega(n^2)$ clauses. But the careful reader will notice that the above result holds for *any local search algorithm*. We have chosen to state our result only for the greedy algorithm, because it is reasonable to expect that the greedy algorithm performs well for formulae with fewer clauses, even for formulae with $O(n)$ clauses. Recent experimental results by Kautz and Selman (which actually motivated this work) seem to support this hope.

REFERENCES

[1] M.R. Garey and D.S. Johnson, *Computers and Intractability: A guide to the Theory of NP-completeness*, (Freeman, San Fransisco, 1979)

[2] T. Hagerup and C. Rüb, *A guided tour of the Chernoff bounds*, Inf. Proc. Letters 33 (1989/90) 305-308.