Probabilistic Greedy Heuristics for Satisfiability Problems

Rajeev Kohli*
Ramesh Krishnamurti†

September 16, 2005

*Professor, Graduate School of Business, Columbia University; rk35@columbia.edu.
†Professor, School of Computing, Simon Fraser University; ramesh@cs.sfu.ca.

Probabilistic Greedy Heuristics for Satisfiability Problems

Abstract

We examine probabilistic greedy heuristics for maximization and minimization versions of the satisfiability problem. Like deterministic greedy algorithms, these heuristics construct a truth assignment one variable at a time. Unlike them, they set a variable true or false using a probabilistic mechanism, the probabilities of a true assignment depending on the incremental number of clauses satisfied if a variable is set true. We discuss alternative probabilistic functions, and characterize the expected performance of the simplest of these rules relative to optimal solutions. We discuss the advantages of probabilistic algorithms in general, and the probabilistic algorithms we analyze in particular.

1 Introduction

Consider a set of clauses, each clause a disjunction of Boolean variables. The maximum (minimum) satisfiability problem is to find a truth assignment that maximizes (minimizes) the number of clauses that are satisfied. For brevity, we refer to the maximization problem as maxsat, and the minimization problem as minsat. Both maxsat and minsat are NP-Hard problems (Johnson 1974, Kohli, Krishnamurti and Mirchandani 1994).

The purpose of this chapter is to examine probabilistic greedy heuristics for the maxsat and minsat problems, and to describe for the simplest of these heuristics the bounds on the average number of clauses satisfied, relative to an optimal solution. We call the algorithms probabilistic greedy heuristics because they set a truth variable true or false not with certainty, but with a probability that depends on the number of unsatisfied clauses in which the variable appears in negated or unnegated form. There are many methods by which the probabilities can be obtained. We discuss some of these, and give the known results for the simplest kinds of rules, in which the probability is (directly or inversely) proportional to the number of additional clauses satisfied if a variable were set true.

There are three main advantages that probabilistic algorithms have over deterministic ones. First, one can often obtain bounds on the average performance of such algorithms without having to specify the method by which the problem instances are generated. In such cases, one is assured of a performance guarantee across all families of statistical distributions giving rise to instances of a problem. This is a form of robustness, obtained by probabilistic algorithms because they can use the structure of a specific problem instance in such a manner as to nullify, at least in part, those peculiarities of data as give rise to worst-case instances for deterministic algorithms. One still obtains worst-case bounds for probabilistic algorithms, but these are bounds on the average performance, which are often higher than the worst-case bounds of similar deterministic algorithms. The second advantage is that analyzing the average performance of probabilistic algorithms is often much simpler than analyzing the average performance of deterministic algorithms. Note that this point is distinct from the possible derandomization of a probabilistic algorithm, an example of which in the present context is the algorithm by Mahajan and Ramesh (1999), which derandomizes the probabilistic algorithm for the maxsat problem due to Goemans and Williamson (1995). The third advantage is that by selecting the best solution value over a large number of runs of a probabilistic algorithm, one obtains a solution that almost certainly has a worst-case bound that is no smaller than the average performance bound for the probabilistic algorithm.

2 Maxsat and Minsat Problems

Let $U = \{u_1, \ldots, u_n\}$ denote a set of n Boolean variables, where each variable $u \in U$ can be true or false. We denote by $\bar{u}$ the negation of a truth variable u. We use the term "literal" to refer to a truth variable in negated or unnegated form. Thus, u and $\bar{u}$ are both literals. A truth assignment for an instance of a satisfiability problem (either maxsat or minsat) comprises a truth setting for each variable, where each variable is set either to true or to false.

Let $C = \{c_1, \ldots, c_m\}$ denote a set of m clauses. Each clause $c \in C$ is a disjunction of some, possibly all, of the literals in U. A clause is satisfied if at least one of the literals it contains is true. For example, $c = u_1 \vee \bar{u}_4$, a disjunction of two literals, is satisfied if u_1 is true and/or u_4 is false.

The satisfiability problem is to determine if there is a truth assignment that satisfies all clauses in C. The problem is NP-Complete if each clause contains three or more literals (Even, Itai and Shamir 1976). An optimization problem corresponding to satisfiability is the maximum satisfiability or maxsat problem. It is NP-Hard (Johnson 1974), and it requires the identification of a truth assignment that satisfies the maximum number of clauses in C. The complementary problem, called minimum satisfiability, or minsat, requires the identification of a truth assignment that satisfies the minimum number of clauses in C. We note that minsat is equivalent to maximizing the number of conjunctive clauses, each of which is satisfied only if all its literals are true.

Minsat is readily solved if there is an assignment that satisfies no clause, because in this case each variable, or its negation, does not appear in any clause. However, the general minsat problem is NP-Hard if each clause contains at least two literals (Kohli, Krishnamurti and Mirchandani 1994). A deterministic approximation algorithm has since been provided for minsat by using an approximation-preserving transformation from minsat to the vertex cover problem (Marathe and

Ravi, 1996). Furthermore, an approximation algorithm with a performance ratio of ρ for minsat implies the existence of an approximation algorithm with the same performance ratio for vertex cover. This suggests that it is hard to provide an approximation algorithm for minsat with a performance ratio better than 2.

Let $u_1, u_2, \ldots, u_n$ denote an arbitrary ordering of the n truth variables in U. We consider the following probabilistic greedy heuristic(s) for solving maxsat and minsat.

Initialization. (step 1): Let $C_1 = C$ denote the set of all clauses in an instance of the maxsat or minsat problem. Let $C_1(u_1)$ denote the subset of clauses in C_1 that contain variable u_1. Let $C_1(\bar{u}_1)$ denote the subset of clauses in C_1 that contain variable $\bar{u}_1$. Let x_1 and y_1 denote the number of clauses in sets $C_1(u_1)$ and $C_1(\bar{u}_1)$. Set

$$u_1 = \begin{cases} \text{True} & \text{with probability } p_1 = f(x_1, y_1) \\ \text{False} & \text{with probability } 1 - p_1, \end{cases}$$

where $f(x_1, y_1) \in [0, 1]$. Eliminate all satisfied clauses. Define C_2, the set of clauses not satisfied at the end of step 1:

$$C_2 = \begin{cases} C_1 \setminus C_1(u_1) & \text{if } u_1 \text{ is selected at step 1,} \\ C_1 \setminus C_1(\bar{u}_1) & \text{if } \bar{u}_1 \text{ is selected at step 1.} \end{cases}$$

Iteration. (step j): Let C_j denote the set of clauses that are not satisfied at the end of step $j-1$. Let $C_j(u_j)$ denote the subset of clauses in C_j that contain u_j. Let $C_j(\bar{u}_j)$ denote the subset of clauses in C_j that contain $\bar{u}_j$. Let x_j and y_j denote the number of clauses in $C_j(u_j)$ and $C_j(\bar{u}_j)$. Set

$$u_j = \begin{cases} \text{True} & \text{with probability } p_j = f(x_j, y_j) \\ \text{False} & \text{with probability } 1 - p_j, \end{cases}$$

where $f(x_j, y_j) \in [0, 1]$. Eliminate all satisfied clauses. Define C_{j+1}, the set of clauses not satisfied at the end of step 1:

$$C_{j+1} = \begin{cases} C_j \setminus C_j(u_j) & \text{if } u_j \text{ is selected at step } j, \\ C_j \setminus C_j(\bar{u}_j) & \text{if } \bar{u}_j \text{ is selected at step } j. \end{cases}$$

Termination. Stop if $C_{j+1} = \phi$ or if $j = n$.

There are many possible functions $f(\cdot)$ that one can use to determine the probabilities for each of the maxsat and minsat problems. The only conditions are that the values of the function increases with x_j (y_j) and decreases with y_j (x_j) for the maxsat (minsat) problem, and that the probabilities remain bounded between 0 and 1. For example, we can use the function

$$f(x_j, y_j) = \frac{e^{\beta x_j}}{e^{\beta x_j} + e^{\beta y_j}} = \frac{1}{1 + e^{-\beta(x_j - y_j)}}$$

for the maxsat problem. The function has value 1/2 when $x_j = y_j$, approaches 1 as x_j becomes much larger than y_j, and approaches zero as y_j becomes much larger than x_j. This is a "logit" function, which has found much use in the statistics and economics literatures (Maddala 1999). The corresponding function for the minsat problem is

$$f(x_j, y_j) = \frac{e^{\beta y_j}}{e^{\beta x_j} + e^{\beta y_j}} = \frac{1}{1 + e^{\beta(x_j - y_j)}}.$$

Perhaps the simplest class of functions we can use is

$$f(x_j, y_j) = \frac{x_j^\beta}{x_j^\beta + y_j^\beta} = \frac{1}{1 + (y_j/x_j)^\beta}, \quad \beta \geq 1,$$

for the maxsat problem and

$$f(x_j, y_j) = \frac{y_j^\beta}{x_j^\beta + y_j^\beta} = \frac{1}{1 + (x_j/y_j)^\beta}, \quad \beta \geq 1,$$

for the minsat problem. In both cases, $p_j = f(x_j, y_j) = 1/2$ if $x_j = y_j$, and p_j approaches the limiting probabilities 1 (0) in the desired manner. Different values of β give different rates at which the probabilities approach their limiting values; as β becomes arbitrarily large, the probabilities approach the limiting values 0 and 1, and the algorithm become a deterministic greedy heuristic. In this sense, a probabilistic rule of the above sort is a generalization of a deterministic greedy heuristic. The only results so far obtained are for the special case of $\beta = 1$ in the above expression; theoretical analysis of the general class of greedy heuristics remains open. In the next section, we describe the results that are known for $\beta = 1$ in the above expression.

2.1 Performance Bound for Maxsat Problem

In the following discussion, x_j denotes the number of clauses in which the literal u_j occurs, y_j denotes the number of clauses in which the literal $\bar{u}_j$ occurs. Let $n_j = x_j + y_j$. As noted above, we consider a probabilistic greedy heuristic that sets the jth variable true with probability $p_j = x_j/n_j$.

Without loss of generality, let $u_j, j = 1, \ldots, n$, comprise the optimal solution. Let z_j denote the value of the optimal solution to the subproblem comprising clauses in set C_j. Let a_j denote the value of the optimal solution to the subproblem obtained by eliminating u_j, and all the clauses satisfied when u_j is true. Similarly, let $\bar{a}_j$ denote the value of the optimal solution to the subproblem obtained by eliminating $\bar{u}_j$, and all the clauses satisfied when u_j is false. The following lemma provides a bound on both a_j and $\bar{a}_j$ in terms of z_j, x_j, and n_j.

Lemma 1 *(*Kohli and Krishnamurti 1989)
$a_j = z_j - x_j$ and $\bar{a}_j \geq \max\ \{0, z_j - n_j\}$, for all $j = 1, \ldots, k$.

Proof. Since u_j occurs in x_j clauses, the optimal solution to the maxsat subproblem comprising the set of clauses $C_j \setminus C_j(u_j)$ is, trivially, $a_j = z_j - x_j$. Also, $x_j \leq n_j$ (by definition), so that $z_j - x_j \geq z_j - n_j$. Of the $z_j - x_j$ clauses in set $C_j \setminus C_j(u_j)$, at most $n_j - x_j$ clauses contain literal $\bar{u}_j$. Hence the maxsat subproblem comprising the set of clauses $C_j \setminus C_j(\bar{u}_j)$ has an optimal solution $\bar{a}_j$ no smaller than $z_j - x_j - (n_j - x_j) = z_j - n_j$. As n_j can exceed z_j, and as the value of the optimal solution to the maxsat subproblem comprising clauses $C_j \setminus C_j(\bar{u}_j)$ is nonnegative, $\bar{a}_j \geq \max\ \{0, z_j - n_j\}$, $j = 1, \ldots, k$. $\square$

We now obtain the main result for the maxsat problem.

Theorem 1 *(*Kohli and Krishnamurti 1989)
On average, the greedy heuristic for the maxsat problem satisfies at least 2/3 of the number of clauses satisfied by an optimal truth assignment.

Proof. We prove the theorem by induction on the number of variables. We first prove the result for $n = 1$. Without loss of generality, assume that variable u_1 is true in an optimal assignment. Then the value of the optimal solution is $z_1 = x_1$. The expected value of the greedy solution is

$$p_1 x_1 + (1 - p_1)(n_1 - x_1),$$

where $p_1 = x_1/n_1$. Thus, the expected performance ratio of the heuristic is

$$E[r_1] = \frac{1}{z_1}\Big(p_1x_1+(1-p_1)(n_1-x_1)\Big) = \frac{1}{x_1}\left(\frac{x_1}{n_1}x_1 + \frac{n_1 - x_1}{n_1}(n_1 - x_1)\right).$$

Given n_1, the lower bound on $E[r_1]$ is obtained by minimizing the above expression with respect to x_1, which can be verified to occcur at $x_1 = n_1/\sqrt{2}$. Substituting this value of x_1 in $E[r_1]$ and simplifying yields

$$E[r_1] \geq 2\sqrt{2} - 2 \geq \frac{2}{3}.$$

Now suppose

$$E[r_l] \geq \frac{2}{3} \quad \text{for all } l \leq k - 1.$$

We show that

$$E[r_k] \geq \frac{2}{3} \quad \text{for all } k.$$

Suppose the probabilistic greedy heuristic sets u_k true, satisfying x_k clauses. Then the expected number of clauses satisfied by the probabilistic greedy heuristic is no smaller than

$$x_k + \frac{2}{3}a_k.$$

By a similar argument, if the greedy heuristic sets u_k false at step 1, the expected value of its solution is no less than

$$n_k - x_k + \frac{2}{3}\bar{a}_k.$$

As u_k is selected with probability $p_k = x_k/n_k$, and $\bar{u}_k$ is selected with probability $1 - p_k$, the expected value of the heuristic solution has the lower bound

$$E[f_k] \geq \frac{x_k}{n_k}\left(x_k + \frac{2}{3}a_k\right) + \frac{n_k - x_k}{n_k}\left(n_k - x_k + \frac{2}{3}\bar{a}_k\right).$$

From Lemma 1, $a_k \geq z_k - x_k$, which gives

$$E[f_k] \geq \frac{x_k}{n_k}\left(x_k + \frac{2}{3}(z_k - x_k)\right) + \frac{n_k - x_k}{n_k}\left(n_k - x_k + \frac{2}{3}\bar{a}_k\right).$$

Also, from Lemma 1,

$$\bar{a}_k \geq \max\{0, z_k - n_k\}.$$

Consider $z_k > n_k$. Then

$$\bar{a}_k \geq z_k - n_k > 0,$$

and the above inequality for $E[f_k]$ becomes

$$E[f_k] \geq \frac{x_k}{n_k}\left(\frac{2z_k}{3} + \frac{x_k}{3}\right) + \frac{(n_k - x_k)^2}{n_k} + \frac{2(n_k - x_k)}{3n_k}(z_k - n_k).$$

Simplifying,

$$E[f_k] \geq \frac{(x_k)^2}{3n_k} + \frac{2z_k x_k}{3n_k} + \frac{(n_k - x_k)^2}{n_k} + \frac{2(n_k - x_k)}{3n_k}(z_k - n_k).$$

The right hand side obtain its minimum value when $x_k = n_k/2$, which implies

$$E[f_k] \geq \frac{2z_k}{3} \quad \text{and} \quad E[r_k] = \frac{E[f_k]}{z_k} \geq \frac{2}{3}.$$

Now consider $z_k \leq n_k$. Then

$$\bar{a}_k \geq 0 \ (\geq z_k - n_k),$$

and therefore

$$E[f_k] \geq \frac{x_k}{n_k}\left(x_k + \frac{2}{3}(z_k - x_k)\right) + \frac{n_k - x_k}{n_k}(n_k - x_k).$$

Simplifying

$$E[f_k] \geq \frac{x_k^2}{3n_k} + \frac{2z_k x_k}{3n_k} + \frac{(n_k - x_k)^2}{n_k}.$$

The right hand side of the above expression can be verified to obtain its minimum value when

$$x_k = \frac{3n_k - z_k}{4},$$

at which value of x_k

$$E[f_k] \geq \frac{n_k}{4} + \frac{z_k}{2} - \frac{z_k^2}{12n_k}.$$

The right hand side of the above expression takes its smallest value when $n_k = z_k$, for which

$$E[f_k] \geq \frac{z_k}{4} + \frac{z_k}{2} - \frac{z_k}{12} = \frac{2}{3} z_k.$$

It follows that

$$E[r_k] = \frac{E[f_k]}{z_k} \geq \frac{2}{3}.$$

□

To see that the bound in Theorem 1 is tight, consider the following problem instance in which there are k variables and 2^k clauses.

Clause	u_k	$\bar{u}_k$	u_{k-1}	$\bar{u}_{k-1}$	. .	u_2	$\bar{u}_2$	u_1	$\bar{u}_1$
1	0	1	0	1	. .	0	1	0	1
2	0	1	0	1	. .	1	0	0	0
3	0	1	0	1	. .	0	0	0	0
.	.	.	.	.	. .	.	.	.	.
.	.	.	.	.	. .	.	.	.	.
2^{k-2}	0	1	0	1	. .	0	0	0	0
$2^{k-2}+1$	0	1	1	0	. .	0	0	0	0
.	.	.	.	.	. .	.	.	.	.
.	.	.	.	.	. .	.	.	.	.
.	.	.	.	.	. .	.	.	.	.
2^{k-1}	0	1	1	0	. .	0	0	0	0
$2^{k-1}+1$	1	0	0	0	. .	0	0	0	0
.	.	.	.	.	. .	.	.	.	.
.	.	.	.	.	. .	.	.	.	.
.	.	.	.	.	. .	.	.	.	.
2^k	1	0	0	0	. .	0	0	0	0

The expected performance of the probabilistic greedy heuristic is

$$\begin{aligned} E[f_k] &= 2\left(\frac{1}{2}\right)\frac{n_k}{2} + 2\left(\frac{1}{2}\right)^2\frac{n_k}{2^2} + \ldots + 2\left(\frac{1}{2}\right)^k\frac{n_k}{2^k} + \left(\frac{1}{2}\right)^k\frac{n_k}{2^k} \\ &= \frac{2n_k}{3} + \frac{n_k}{3}\left(\frac{1}{4^k}\right). \end{aligned}$$

As $z_k = n_k$, the expected performance ratio equals

$$E[r_k] = \frac{2}{3} + \epsilon, \quad \text{where } \epsilon = \frac{1}{3} \cdot \frac{1}{4^k}.$$

As ϵ can be made to approach 0 arbitrarily closely by increasing k, $E[r_k]$ can be made to approach 2/3 from above arbitrarily closely. Since the asymptotic upper bound for the probabilistic greedy heuristic is 2/3, the lower bound in Theorem 1 is tight.

2.2 Performance Bound for Minsat Problem

As in the last section, we restrict our analysis to a probabilistic greedy heuristic for the minsat problem to the case where $\beta = 1$; that is, the heuristic sets the jth variable true with probability $p_j = y_j/n_j$. The following theorem gives the lower bound on the expected performance ratio of the greedy heuristic.

Theorem 2 *(*Kohli, Krishnamurti and Mirchandani 1996)
On average, the greedy heuristic for the minsat problem satisfies at most twice the number of clauses satisfied by an optimal truth assignment.

Proof. We prove the theorem by induction on the number of variables n. If $n = 1$, the expected number of satisfied clauses is

$$p_1 x_1 + (1 - p_1) y_1 = \frac{y_1}{n_1} x_1 + \frac{x_1}{n_1} y_1 = \frac{2x_1 y_1}{n_1},$$

where $p_1 = y_1/n_1$. Without loss of generality, assume that variable u_1 is true in an optimal assignment. Then the value of the optimal solution is $z = x_1$. Thus the value of the expected performance ratio for the probabilistic greedy heuristic is

$$E(r_n) = \frac{2x_1 y_1}{n_1} = \frac{2y_1}{n_1} \leq 2.$$

Let $l \geq 1$ be an integer such that

$$E(r_n) \leq 2 \text{ for } n = l.$$

We now show that

$$E(r_n) \leq 2 \text{ for } n = l + 1.$$

If the probabilistic greedy heuristic selects u_1 at step 1, the value of the optimal solution at the second step of the greedy heuristic is

$z - x_1$, where z is the optimal solution value of the minsat problem with n variables. However, if the greedy heuristic selects $\bar{u}_1$ at step 1, the value of the optimal solution at the second step is bounded from above by z. Hence the expected number of clauses satisfied by the probabilistic greedy heuristic is bounded from above by

$$p_1\Big(x_1 + E(r_l)(z - x_1)\Big) + (1 - p_1)\Big(y_1 + E(r_l)z\Big).$$

As $E(r_l) \leq 2$ by the induction hypothesis, the value of the above expression is no greater than

$$p_1(x_1 + 2(z - x_1)) + (1 - p_1)(y_1 + 2z).$$

Thus, an upper bound on the expected performance ratio for the probabilistic greedy heuristic is

$$\begin{aligned} E(r_{l+1}) &\leq \frac{1}{z}\Big(p_1(x_1 + 2(z - x_1)) + (1 - p_1)(y_1 + 2z)\Big) \\ &= \frac{1}{z}\Big(2z - p_1(x_1 + y_1) + y_1\Big) \\ &\leq 2. \end{aligned}$$

□

To prove the above bound is tight, consider the following example with $n = 2$ variables and m clauses. Let:

$$\begin{aligned} c_1 &= u_1 \vee u_2, \\ c_2 &= \bar{u}_1, \\ c_i &= \bar{u}_2, \ 3 \leq i \leq m. \end{aligned}$$

The optimal assignment sets both u_1 and u_2 true and satisfies one clause, c_1. The probabilistic greedy heuristic sets u_1 true or false with the same probability (which equals 1/2) at its first step. If it sets u_1 true, then it obtains the optimal solution, setting u_2 true with probability 1 at its second step. Otherwise, at the second step, it sets u_2 true with probability $1 - (1/(m - 1))$, satisfying 2 clauses, c_1 and c_2; and sets u_2 false with probability $1/(m - 1)$, satisfying $(m - 1)$ clauses, $c_2, \ldots, c_m$. Thus, the expected performance ratio (which equals the expected number of satisfied clauses) for the probabilistic greedy heuristic is

$$\frac{1}{2} \cdot 1 + \frac{1}{2}\Big(\frac{m-2}{m-1} \cdot (2) + \frac{1}{m-1} \cdot (m-1)\Big) = 2 - \frac{1}{m-1}.$$

As m tends to infinity, the value of this expression approaches from below the bound derived in Theorem 2. Note that if we interchange u_2 and $\bar{u}_2$ in the above example, then each clause has at most one unnegated truth variable. Such clauses are called Horn clauses, and so it follows that the above bound on the average performance of the probabilistic greedy heuristic remains tight if we restrict the problem instances to Horn clauses. Also note that $n = 2$ in this example and that the optimal clause c_1 contains $s = 2$ variables. Thus, the bound on the average performance of the probabilistic greedy heuristic does not depend on m or s.

3 Using Probabilistic Algorithms

An average performance analysis does not tell us how well a probabilistic algorithm will do if we stop it after a single run for a given problem instance. It is very possible that the solution obtained will be far removed from the average bound. Why then should we be interested in the average performance of a probabilistic algorithm? The answer is that we do not need to stop after a single run. If we run the algorithm a very large number of times (say N), then the average of the solution values will asymptotically converge to a value no smaller than the bound on the average performance in Theorem 1 or Theorem 2. The maximum value across these N runs can of course be no smaller than their average. We can thus select this best solution and be assured that, with a very high probability, it has a value that is no worse than the bound on the average performance of the algorithm. The value of N does not have to be enormously large — in most instances, several thousand, if not several hundred runs, will suffice.

There is another way in which probabilistic algorithms can be used. Suppose we run the algorithm a large number of times (say N), and select the best solution. We can repeat the process a large number of times (say M). We then have a sample of M solution values, each the best solution among N solutions. If we plot the distribution of these M best solutions, we obtain an empirical distribution of the best solution. This is a useful distribution, from which we can make several important inferences. First, we note that as N and M become arbitrarily large, the distribution approaches the "true" distribution of the best solution across NM runs. This argument was first used by Fisher and Tippet (1928) to obtain closed form expressions for distributions of a maximum or minimum value from a sample when it is bounded on one or both sides. The main assumption for these distributions to hold is that the samples be drawn from a continuous distribution. This is not strictly valid for such discrete problems as satisfiability, and so the exact distributions need not hold. But the essence of the argument still remains, and one can use repeated runs to obtain a reference distribution which is asymptotically the entire distribution of interest. This procedure is akin to bootstrapping methods in statistics, except that the sampling process uses a probabilistic algorithm. One can use the reference distribution to make probabilistic statements about a solution value — that is, obtain estimates for the probability that the best solution across many runs will exceed a certain numerical value. Such analyses are not possible for deterministic algorithms; but they are natural for probabilistic algorithms, although we are not aware of literature reporting their use.

4 Conclusion

There are other probabilistic algorithms for the maxsat problem. The simplest of these is a 'random' algorithm that sets each variable true or false with probability 1/2. It is easy to see that the average performance of this algorithm is 1/2 (Kohli and Krishnamurti, 1989). Johnson's (unweighted) deterministic greedy algorithm with a performance ratio of 1/2 can be considered as a derandomization of this simple random algorithm (Vazirani, 2003). A randomized algorithm for maxsat using linear programming rounding, where the fractional value assigned to the variable corresponding to a truth variable is used as the probability of setting the truth variable true, provides a performance guarantee of $1 - 1/e$ (Goemans and Williamson, 1993). The better of the 'random' algorithm and the randomized rounding algorithm using linear programming provides a performance guarantee of 3/4. Approaches using semidefinite programming for the maxsat problem have yielded improved approximation ratios (Goemans and Williamson, 1995). However, algorithms using linear programming and semidefinite programming have high running times. One benefit of using simple probabilistic algorithms is that the running time is low, and the probabilistic algorithm may be run many times on a given problem instance. By retaining the best solution across all the runs, one obtains a solution that, with high probability, is no worse than the bound on the average performance of a probabilistic algorithm.

Several possible problems remain open, including the following. How do different probability rules affect the expected performance of probabilistic greedy heuristics? Is there an optimal choice of the β parameter for the two rules considered in the paper? Are there some types of problems for which one or another probabilistic greedy heuristic does better than others? And are there ways in which resampling methods like bootstrapping (Efron 1979) be used to construct reference distributions from which one can make probabilistic assertions about an obtained solution relative to an optimal solution?

References

Efron, B., “Computers and the theory of statistics: thinking the unthinkable,” *SIAM Review,* 21, 1979, 460–80.

Even, S., A. Itai and A. Shamir, “On the complexity of timetable and multicommodity flow problems,” *SIAM Journal on Computing,* 5, 1976, pp. 691–703.

Fisher, R.A. and L.H.C. Tippet, “Limiting forms ofthe frequency distribution of the largest or smallest member of a sample,” *Proceedings of the Cambridge Philosophical Society,* 24, 1928, 180–190.

Garey, M.R. and D.S. Johnson *Computers and intractibility: a guide to the theory of NP-Completeness,* 1979, San Francisco, CA: Freeman.

Garey, M.R., D.S. Johnson and L. Stockmeyer, “Some simplified NP-complete graph problems,” *Theoretical Computer Science,* 1, 1976, pp. 237–267.

Goemans, M.X. and D.P. Williamson, “Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming,” *Journal of the Association of Computing Machinery,* 42, 1995, 1115–1145.

Goemans, M.X. and D.P. Williamson, “New 3/4-approximation algorithms for the maximum satisfiability problem,” *SIAM Journal on Discrete Mathematics,* 7, 1994, 656–666.

Johnson, D. S., “Approximation algorithms for combinatorial problems,” *Journal Comput. Syst. Sci.,* 9, 1974, pp. 256–278.

Kohli, R. and R. Krishnamurti, “Average performance of heuristics for satisfiability,” *SIAM Journal on Discrete Mathematics,* 2, 1989, pp. 508–523.

Kohli, R., R. Krishnamurti and P. Mirchandani, “The minimum satisfiability problem,” *SIAM Journal on Discrete Mathematics,* 7, 1994, 275–283.

Marathe, M. V. and S. S. Ravi, “On approximation algorithms for the minimum satisfiability problem,” *Information Processing Letters,* 58, 1996, 23–29.

Maddala, G.S., *Limited-Dependent and Qualitative Variables in Econometrics,* Econometric Society Monographs, New York: Cambridge University Press, 1999.

Mahajan, S. and H. Ramesh, “Derandomizing approximation algorithms based on semidefinite programming,” *SIAM Journal on Computing,* 28 (5), 1999, 1641–1663.

Vazirani, V. V., *Approximation Algorithms,* New York: Springer, 2003.