On the Relative Merits of Simple Local Search Methods for the Max Sat Problem

Denis Pankratov[1] and Allan Borodin[2]

[1] Department of Computer Science, University of Toronto
Currently, Department of Computer Science, University of Chicago
pankratov@uchicago.edu
[2] Department of Computer Science, University of Toronto
bor@cs.toronto.edu

Abstract. Algorithms based on local search are popular for solving many optimization problems including the maximum satisfiability problem (MAXSAT). With regard to MAXSAT, the state of the art in performance for universal (i.e. non specialized solvers) seems to be variants of Simulated Annealing (SA) and MaxWalkSat (MWS), stochastic local search methods. Local search methods are conceptually simple, and they often provide near optimal solutions. In contrast, it is relatively rare that local search algorithms are analyzed with respect to the worst-case approximation ratios. In the first part of the paper, we build on Mastrolilli and Gambardella's work [14] and present a worst-case analysis of tabu search for the MAX-k-SAT problem. In the second part of the paper, we examine the experimental performance of determinstic local search algorithms (oblivious and non-oblivious local search, tabu search) in comparison to stochastic methods (SA and MWS) on random 3-CNF and random k-CNF formulas and on benchmarks from MAX-SAT competitions. For random Max-3-SAT, tabu search consistently outperforms both oblivious and non-oblivious local search, but does not match the performance of SA and MWS. Initializing with non-oblivous local search improves both the performance and the running time of tabu search. The better performance of the various methods that escape local optima in comparison to the more basic oblivious and non-oblivious local search algorithms (that stop at the first local optimum encountered) comes at a cost, namely a significant increase in complexity (which we measure in terms of variable flips). The performance results observed for the unweighted MAX-3-SAT problem carry over to the weighted version of the problem, but now the better performance of MWS is more pronounced. In contrast, as we consider Max-k-Sat as k is increased, MWS loses its advantage. Finally, on benchmark instances, it appears that simulated annealing and tabu search initialized with non oblivious local search outperform the other methods on most instances.

1 Introduction

The maximum satisfiability problem is of great interest in both theoretical[8] and applied computer science[18]. The maximum satisfiability problem is NP-

hard. Current state-of-the-art algorithms can solve the problem optimally within a reasonable amount of time only for input instances of moderate size[5]. These methods are mostly based on branch and bound techniques with rather sophisticated rules that try to exploit the structure of the problem. MAX-SAT applications often involve instances of far larger scale than what exact solvers can handle. As a result, a number of approximation algorithms have been developed [6]. Local search based algorithms have gained popularity for their conceptual simplicity and approximation performance.

In the exact MAX-k-SAT problem each clause is restricted to have exactly k literals, and the same variable cannot repeat within the same clause whether negated or not. In the weighted version of the problem, each clause has a real-valued positive weight and the objective is to find a truth assignment that maximizes the total weight of satisfied clauses.

A "basic" local search algorithm for the MAX-SAT problem starts with an arbitrary truth assignment. The neighborhood of a solution consists of all the truth assignments obtained by flipping the truth value of one or a small number of the variables. Most local search algorithms simply use a Hamming distance 1 neighborhood. At each step, the basic local search algorithm looks through the neighborhood for a truth assignment that increases the number of satisfied clauses. If it finds such an assignment, the algorithm flips the value of the corresponding variable(s) and continues. If it does not find an improving assignment, the algorithm terminates.

Many heuristic methods are aimed at improving the performance of the basic local search, such as tabu search, random restarts, plateau moves, boosting, and other methods for escaping local optima [1]. In spite of the popularity and success of local search methods, these algorithms are rarely analyzed with respect to either worst-case or "average case" performance. Mastrolilli and Gambardella seem to be the first to analyze the worst-case performance of tabu search for the unweighted exact MAX-2-SAT problem[14]. In the first part of this paper, we extend their work to the weighted exact MAX-k-SAT problem. Tabu search guarantees a better approximation ratio than "oblivious local search" but loses significantly ta the Khanna et al [13]) "non-oblivious local search" that uses a related potential function to search through the neighborhood. In the second part of the paper, we study the experimental performance of local search methods for the MAXSAT problem. Our experiments indicate that tabu search and the stochastic local search methods consistently outperform both the oblivious and non-oblivious versions of basic local search. However, initializing tabu search with the truth assignment obtained by non-oblivious local search leads to results more competitive with the stochastic methods. But perhaps of equal interest is the fact that tabu search and the stochastic methods require substantially more time, and if the goal is simply to obtain a reasonable approximation then the basic methods (and especially non-oblivious local search) have an advantage in terms of significantly reduced time complexity.

2 Local Search Algorithms for MAX-k-SAT

The input for each of the following algorithms is a boolean formula in CNF with m clauses over n variables. For weighted MAXSAT, a weight function is also part of the input.

2.1 Oblivious Local Search

For a given truth assignment τ, its one-flip neighborhood is the set of all truth assignments at Hamming distance one from τ. Oblivious local search starts with an arbitrary fixed truth assignment. At each step, it searches the one-flip neighborhood of the current assignment for neighbors that achieve a better value of the given objective function. If such a neighbor exists, the algorithm replaces the current truth assignment with a truth assignment from the one-flip neighborhood that satisfies the most number of clauses. Ties are broken arbitrarily. If such a neighbor does not exist, oblivious local search terminates.

For unweighted MAXSAT the objective function is simply the number of satisfied clauses. The running time of oblivious local search is polynomial in this case, as each step improves the value of the objective function by at least one, and the optimal value is bounded above by m. For weighted MAXSAT the objective function is the total weight of the satisfied clauses. Without any restrictions on weights, the running time of oblivious local search is no longer necessarily polynomial. This can be remedied by insisting that improvements at each step are sufficiently large.

2.2 Non-Oblivious Local Search

The idea behind non-oblivious local search [13] is to introduce a related potential function and use it in the neighborhood search. This potential function gives preference to the clauses satisfied by many literals, as they are likely to stay satisfied even if the algorithm flips many variables in the future. For example, let C_j denote the set of clauses satisfied by j literals. Then the potential function for MAX-2-SAT is $3/2|C_1| + 2|C_2|$. For the case of MAX-k-SAT with $k > 2$, the reader is referred to Khanna et al paper [13]. Replacing $|C_j|$ by the total weight of C_j in the potential function provides a natural extension of this approach to the weighted case of MAXSAT. The running time analysis of this algorithm is similar to that of oblivious local search.

2.3 Tabu Search

Oblivious and non-oblivious local search terminate as soon as they achieve a local optimum for the given objective (respectively, the related potential) function. Tabu search offers a determinstic method for attempting to improve upon the current local optimum. Each iteration of tabu search consists of two stages. In the first stage, given a current truth assignment, oblivious local search is used

to compute a local optimum. In the second stage, tabu search maintains an additional data structure - a list of size t. This list is commonly referred to as a *taboo list*. and t is commonly called *taboo tenure*. When tabu search reaches a local optimum, it tries to escape this local optimum in phase 2 as follows. The algorithm records (x_i, t_i) pairs for the last t steps in the list, where x_i is a variable flipped at step t_i. During some of these steps, the current truth assignment can worsen, during other steps it can improve. If at some point, the current truth assignment improves over the local optimum found in stage 1, then phase 1 is repeated starting from the improved truth assignment. If during phase 2, tabu search does not find a solution that improves over the current local optimum, the algorithm terminates. Each step of tabu search in the escape phase consists of flipping a variable. The algorithm follows the following rules (in given order) to decide which variable to flip.

aspiration condition - if flipping a variable improves the best value of the objective function found so far, then the best such variable is chosen;

taboo - if there are variables that appear in unsatisfied clauses and that were not flipped in the last t steps, i.e. they are not in the taboo list, the algorithm chooses the best such variable;

LRU - if all variables that appear in unsatisfied clauses also appear in the taboo list, then the least recently used such variable is selected.

In the above rules, the best variable means that flipping it results in the largest increase of the objective function, or smallest decrease in the objective function, if no variable can improve it. All ties are broken arbitrarily. Taboo tenure controls the number of allowed steps during the escape phase. Mastrolilli and Gambardella argued that n is a reasonable choice for taboo tenure. The proof of a worst-case approximation ratio of tabu search relies on this requirement.

The same algorithm can be used for solving the weighted MAX-k-SAT problem, except that the weights of clauses are used in consideration of which variable to flip. In the unweighted case, it is easy to see that tabu search with taboo tenure of n runs in polynomial time. It improves the objective by at least one in every n steps, and m is a bound on the optimal value of the objective function. As before, in the weighted case we can guarantee strongly polynomial running time by considering only large enough improvement for the stopping conditions in phase 1.

2.4 Simulated Annealing

We present a version of simulated annealing (SA) that appears in [17] and was found to work well for the satisfiability problem. a randomized algorithm. SA was motivated by an analogous physical process, and the parameters of this algorithm have a corresponding semantic meaning. SA keeps track of the current "temperature". Initially, the temperature is high and the algorithm explores the solution space uniformly at random. As the temperature starts to cool down, SA gradually starts to prefer solutions that achieve better values of the objective

function, concentrating on more promising parts of the solution space. The rule that specifies how temperature changes with time is called *temperature schedule*. Our implementation is specified by three parameters: 1) MT - maximum temperature that the algorithm starts with, 2) DR - decay rate, which controls by how much the temperature drops from step to step, 3) mT - minimum temperature, at which SA stops. SA is initialized with an arbitrary truth assignment and proceeds in steps. At step s SA computes the current temperature $T = MT \exp(-s \cdot DR)$. If $T < mT$, then SA terminates. Otherwise, it computes $p_i = 1/(1+\exp(-\Delta(i)/T))$, where $\Delta(i)$ is the change in the objective function if variable i is flipped. It then flips variable i with probability p_i. After all variables have been processed, SA moves to the next step $s+1$. We use the parameters $MT = 0.3, mT = 0.01$, and $DR = 1/n$ as suggested in [17].

2.5 MaxWalkSat

There are many variants of maxwalksat algorithms. In general, given any current truth assignment for the unweighted MAXSAT problem, an unsatisfied clause is chosen uniformly at random among all unsatisfied clauses. Various heuristics are then used to select a literal from this clause and the truth value of that literal is flipped. Our experiments indicate that overall, the "productsum" heuristic performs best, and hence we restrict our attention to this heuristic. Suppose MWS decides to choose a literal in a clause $C = z_1 \vee z_2 \ldots \vee z_k$. Let $b(i) =$ the number of clauses that become unsatisfied if the literal z_i is flipped. Then "productsum" assigns a value $v_i = \left(\prod_{j \neq i} b(j)\right)\left(\sum_{j \neq i} b(j)\right)$ for each literal z_i in clause C and the flips literal z_i with probability $\frac{v_i}{\sum_{1 \leq j \leq k} v_j}$. In the weighted case, MWS considers clauses of highest weight to be "hard" clauses. Given any truth assignment, it chooses a random unsatisfied hard clause and applies "pickproductsum" heuristic to it with $b(i) =$ the weight of clauses that become unsatisfied if z_i is flipped. If all hard clauses are satisfied, MWS chooses a random unsatisfied clause and applies "pickproductsum" heuristic. After this step, some hard clauses might become unsatisfied, and MWS will try to fix them in the very next step.

3 Background

Unless otherwise stated, MAX-k-SAT will mean exact MAX-k-SAT. Oblivious local search with 1-flip neighborhood achieves the approximation ratio of $\frac{k}{k+1}$ for the unweighted (and weighted) MAX-k-SAT problem, and this ratio is tight [8]. Non-oblivious local search provides a better worst-case guarantee of $\frac{2^k-1}{2^k}$ for the same problem [13]. For the MAX-2-SAT problem, tabu search was shown to have the tight approximation ratio of $\frac{3}{4}$ [14]. This ratio matches that of non-oblivious local search for MAX-2-SAT, which raises a question as to whether the two algorithms have the same approximation ratio for MAX-k-SAT for all $k \geq 2$. This paper answers this qusestion in negative, showing that tabu search

has a weaker approximation guarantee than non-oblivious local search for MAX-k-SAT with $k > 2$.

A general inapproximability result says that if $P \neq NP$, then $\frac{2^k-1}{2^k}$ is (essentially) the best possible approximation ratio achievable by any polynomial time algorithm for MAX-k-SAT with $k > 2$ [9]. A simple randomized algorithm, the one that picks a truth assignment uniformly at random, satisfies $\frac{2^k-1}{2^k}$ of all clauses in the exact MAX-k-SAT formula in expectation. Derandomization of this algorithm leads to a simple greedy algorithm achieving the approximation ratio of $\frac{2^k-1}{2^k}$ [11]. The case $k = 2$ is special in the MAX-SAT world. Currently, the best approximation ratio for MAX-2-SAT is .931 (Feige and Goemans[7]) using an algorithm based on semidefinite programming relaxation and rounding. An inapproximability result for MAX-2-SAT states that for any $\epsilon > 0$ it is NP-hard to approximate MAX-2-SAT within a factor of $\frac{21}{22} + \epsilon \approx 0.955 + \epsilon$[9].

A natural extension of the 1-flip neighborhood is a larger p-flip neighborhood for $p > 1$. The size of this neighborhood is $\sum_{j=1}^{p} \binom{n}{j}$ for a formula over n variables. Even for "small" constant values of p, it still requires a substantial amount of time to search through the entire neighborhood, and experimentally the quality of solutions seems to be not much better than those obtained through a 1-flip neighborhood. From the worst-case point of view, [13] shows that oblivious local search with an $o(n)$-flip neighborhood has the tight approximation ratio of $\frac{2}{3}$ for MAX-2-SAT - the same as achieved with a 1-flip neighborhood. In general, these larger neighborhoods are not practical, and so this paper focuses on algorithms with the 1-flip neighborhood.

The second part of this paper deals with an empirical evaluation of different algorithms based on local search. We consider both benchmark examples from the "Second Evaluation" of MAX-SAT solvers (see [10] for a detailed description of these benchmark instances) and random k SAT instances. Random exact k SAT instances were generated by choosing formulas uniformly at random with the clause density around the estimated phase transition. There is a discrepency between what has been proven rigorously about the threshold values for k SAT in contrast to what has been experimentally shown (and justified by well motivated analysis). See [4], [15] and [3] for current results concerning threshold values. The situation for 3-SAT represents a glaring gap in our current knowledge. namely, the best lower bound c_3 (for which clause density $c < c_3$ implies satisfiability with high probability) is a constructive bound $c_3 > 3.52$ obtained by a myopic (i.e. greedy) algorithm [12]. The provable upper bound is $c_3 < 4.51$. Experimentally, the conjectured threshold is approximately 4.24 (see [16]).

4 Worst-Case Analysis of Tabu Search

Our version of tabu search, as described in Section 2, uses the length of taboo list equal to the number of variables. Tabu search contains oblivious local search as a subroutine, so an analysis of oblivious local search occurs as a part of worst-case analysis of tabu search. In the unweighted (and also weighted) case of the exact MAX-k-SAT problem, oblivious local search achieves a worst-case

approximation ratio of $\frac{k}{k+1}$. In fact, Khanna et al [8] prove the following stronger result on the "totality ratio".

Lemma 1 *At a local optimum, oblivious local search satisfies at least $\frac{k}{k+1}$ of the total number of clauses in the formula.*

Khanna et al show that the $\frac{k}{k+1}$ bound is tight. Adapting the proof of Lemma 1 to tabu search, Mastrolilli and Gambardella showed a $\frac{3}{4}$ approximation guarantee of tabu search for the MAX-2-SAT problem. We extend their result to the MAX-k-SAT problem for all $k \geq 2$.

Theorem 1 *Tabu search outputs a truth assignment that satisfies at least $\frac{k+1}{k+2}$ of the total number of clauses.*

Proof. Suppose, oblivious local search is given a formula ϕ in k-CNF form with m clauses over n variables. The initial truth assignment is X^0. Let X^s be the truth assignment output by tabu search and let C_j^t denote the set of clauses that have exactly j literals satisfied by X^s at step t. By the halting condition, the algorithm terminated at step $s+n$. There exist t, such that $0 < t \leq n$, and each variable from an unsatisfied clause at step $s+t$ was flipped exactly once during the last t steps. To prove this claim, consider two possibilities at step $s+n$. If n variables were flipped during last n steps, then the claim follows trivially with $t = n$. If less than n variables were flipped between steps s and $s+n$, then by the pigeonhole principle at least one variable was flipped twice during last n steps. In this case, choose a variable that is flipped the second time at the earliest step and let that step be $s + d + 1$.

Then $t = d$ satisfies the claim. To see that, consider step $s + d$. This is the step immediately before the chosen variable was flipped the second time. The algorithm had to repeat a variable, because all variables from unsatisfied clauses were in the taboo list at step $s + d$. In particular, each of these variables was flipped at least once during the d last steps. The truth values could not be flipped more than once, since the earliest step, when a variable is flipped for the second time, is $s + d + 1$. Then at step $s + d$ all the variables occuring in unsatisfied clauses were flipped exactly once during last d steps, as required.

Taking t as in the above claim, a clause with an unsatisfied literal at step s is satisfied by that literal at step $s+t$. Then a clause at step $s+t$ is unsatisfied by all literals only if it is satisfied by all literals at step s, i.e. $C_0^{s+t} \subseteq C_k^s$. This provides a lower bound on the number of C_k-clauses at the solution: $|C_0^s| \leq \left|C_0^{s+t}\right| \leq |C_k^s|$, where the first inequality follows because the solution does not improve in between steps s and $s + n$. Together with X^s being a local optimum and the Lemma 1 , we get $m = \Sigma_{i=0}^{k} |C_i^s| \geq |C_0^s| + |C_1^s| + |C_k^s| \geq |C_0^s| + k\,|C_0^s| + |C_0^s| = (2 + k)\,|C_0^s|$. Thus $|C_0^s| \leq \frac{m}{k+2}$ and the theorem follows.

The approximation ratio guaranteed by tabu search for MAX-2-SAT matches that of non-oblivious local search, suggesting that tabu search might have the same worst-case performance as non-oblivious local search. However, we show that for $k > 2$, although tabu search improves over oblivious local search, it has a significantly weaker approximation guarantee than non-oblivious local search.

Theorem 2 *The worst-case approximation (and totality) ratio of tabu search with tabu tenure n is at most $\frac{3k-3}{3k-2}$ of the total number of clauses.*

Proof. Fix k. The goal is to construct a satisfiable formula, such that the truth assignment that tabu search finds satisfies $\frac{3k-3}{3k-2}$ of the total number of clauses in the formula. The formula is over $2k-1$ variables, which we denote $x_1, x_2, \ldots, x_{2k-1}$. The formula consists of 5 sets of clauses:

$S_1 = \{\bar{x_1} \vee \bar{x_2} \vee \ldots \vee \bar{x_k}\}$,

$S_2 = \bigcup_{i=1}^{k} \{x_i \vee \bigvee_{j=i+1}^{k+i-1} \bar{x_j}\}$,

$S_3 = \bigcup_{i=k+1}^{2k-2} \{x_i \vee \bigvee_{j=i+1}^{2k-1} \bar{x_j} \vee \bigvee_{j=1}^{k-i} x_j\}$,

$S_4 = \bigcup_{i=1}^{k-2} \{\bar{x_i} \vee \bigvee_{j=i+1}^{i+k-1} x_j\}$,

$S_5 = \{x_1 \vee x_2 \vee \ldots \vee x_{k-1} \vee x_{2k-1}\}$.

Once this formula is given, the adversary chooses the initial truth assignment and a variable to flip in case of a tie. The initial truth assignment is the one with all variables set to true. Under this truth assignment one clause from S_1 is unsatisfied, and all the other clauses in the formula are satisfied. The claim is that during the next $n = 2k - 1$ steps tabu search does not improve upon the initial truth assignment, i.e. at least one clause remains unsatisfied at all times. This would prove the theorem, since tabu search stops after n steps, the formula contains $|S_1| + |S_2| + |S_3| + |S_4| + |S_5| = 1 + k + (k-2) + (k-2) + 1 = 3k - 2$ clauses, and an optimal truth assignment satisfies all the clauses. For example, the truth assignment that assigns value true to x_1 and false to all the other variables is an optimal one, as can be readily checked.

To prove the claim, we trace the execution of tabu search step by step.

Table 1. Execution of tabu search step by step

Step No	Allowed variables to flip	Chosen variable	Taboo list
0	$x_1, x_2, \ldots, x_k$	x_1	$\emptyset$
$1 \leq i \leq k$	$x_{i+1}, x_{i+2}, \ldots, x_{i+k-1}$	x_{i+1}	$\{x_1, \ldots, x_i\}$
$k+1 \leq i \leq 2k-2$	$x_{i+1}, x_{i+2}, \ldots, x_{2k-1}$	x_{i+1}	$\{x_1, \ldots, x_k, \ldots, x_i\}$

In Table 1, allowed variables to flip are the variables that occur in unsatisfied clauses, but not in the taboo list. The chosen variable is the one chosen by the adversary. It is straightforward to verify that at each step exactly one clause is unsatisfied, that flipping any of the allowed variables does not change this condition, and that aspiration conditions never hold. During the execution of tabu search, the truth values of variables will be flipped in order $x_1, x_2, \ldots, x_{2k-1}$. In general, S_1 contains an initially unsatisfied clause. Clauses from S_2 are required for flipping truth assignments of variables $x_1, \ldots, x_k$. Clauses from S_3 are

required for flipping truth values of variables $x_{k+1}, \ldots, x_{2k-1}$. Clauses from S_4 guarantee that aspiration conditions are never met. Finally, the clause from S_5 is unsatisfied after the execution of tabu search, proving the claim.

5 Experimental Results

In addition to the three deterministic and two randomized algorithms specified in section 2, we also consider tabu search when first initialized with a truth assignment found by non-oblivious local search [3]. We consider a system-independent definition of the running time of a local search algorithm, namely simply counting the number of variable flips. The complexity of our determinstic algorithms is determined when a local optimum is reached. In contrast, for simulated annealing (SA), the complexity is bounded by the setting of parameters and for MaxWalkSat (MWS), the stopping time is determined by an ad-hoc limit on the number of flips. All algorithms will immediately terminate if a satisfying assignment is found. Termination for the SA and MWS algorithms does not generally coincide with a local optimum.

The relative performance of all algorithms is evaluated with respect to both benchmark and random exact MAX-k-SAT instances. To generate a random formula, we first fix n = number of variables, m = number of clauses, and k = number of literals per clause. Then for each clause, k variables are chosen uniformly at random without replacement, and each of these variables is negated with probability $\frac{1}{2}$.

We first compare the performance of algorithms for random instances of the unweighted MAX-3-SAT problem. The number of variables in a formula varies from 50 to 1100 in increments of 50. The number of clauses m is always chosen to be slightly above the conjectured phase transition for 3-SAT, more specifically, $m = 4.25n$. For a given n, the performance of each algorithm is averaged over 500 trials. Each trial is an execution of the algorithm on a random formula starting at the all-variables-false truth assignment. For large n, it is not feasible to compute the exact solution and, consequently, the true approximation ratio. Instead, we calculate the "totality ratio" of the satisfied clauses by a given algorithm to the total number of clauses. This is only a lower bound on the approximation ratio, but given our choice of clause density around the phase transition, the formulas are "almost" satisfiable, so the true maximum is around m, and the computed bound is a good estimate of the true approximation ratio. All algorithms are excuted on the same formulas with the same initial truth assignment which allows for a relative comparison of performance. Given that the totality ratios are close to 1, we compare performance in terms *unsat ratio*, the "unsatisfiability ratio" defined as the ratio of the number of unsatisfied clauses to the total number of clauses.

[3] We excluded experimental results of a simple greedy algorithm based on derandomizing the naive randomized method since the greedy algorithm did not compare favorably to any of the other methods. Furthermore initalizing other methods using this greedy algorithm did not substantially improve performance.

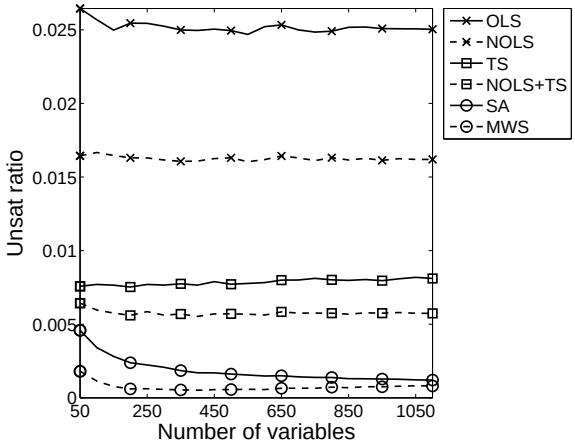
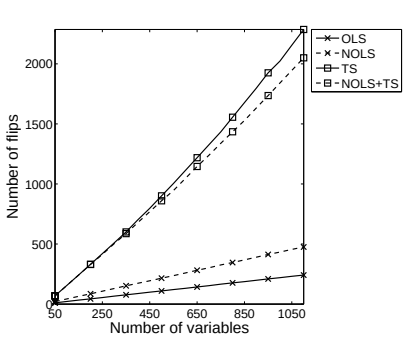


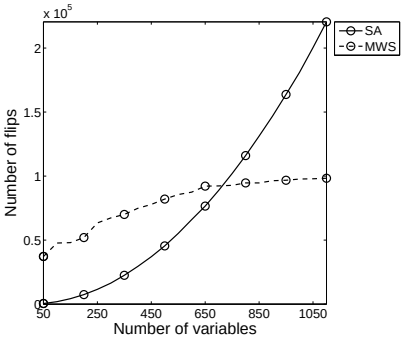
Fig. 1. Average performance when executing on random instances of exact MAX-3-SAT.

Figure 1 presents the performance results for random MAX-3-SAT instances. All the techniques are clearly separated from each other in terms of their performance. The behavior of non-oblivious local search and its oblivious counterpart matches their relative standings in the worst-case scenario. However, in spite of a weaker worst-case guarantee, tabu search beats non-oblivious local search very comfortably. In addition, if tabu search is initialized with a truth assignment found by non-oblivious local search, the resulting hybrid method outperforms plain tabu search. Simulated annealing and MaxWalkSat are the overall leaders and they get very close (on average) to the optimal 0 unsat ratio. The fact that for SA and MSW the unsat ratio is highest for small n is due to the relatively small number of total clauses. For $n \geq 150$, the unsat ratio for MWS is at most .00082. As we will see in Figures 2 and 3 the better performance of the SA and MSW algorithms comes at a greater computational cost.

It is not suprising that techniques giving better results tend to require more time. An exception to this rule is the hybrid of non-oblivious local search with tabu search, which finds better truth assignments than regular tabu search and for large enough formulas uses somewhat fewer computations. The running time for all the determinstic techniques scale quite reasonably with an increase in the size of the formula. The running time of simulated annealing (for the given temperature schedule) blows up dramatcally and MaxWalkSat was given a fixed stopping time of 100,000 flips. The fact that the average running time of MWS is less than 100,000 flips for a small number of variables indicates that the method obtains a satisfying assignment for many instances. Figure 3 depicts the normalized performance of algorithms relative to the four deterministic methods. That is, we measure the normalized performance "A/B" of algorithm A relative to algorithm B by terminating A at the point that it uses the number of flips used by B. The normalized performance indicates that the non-oblivious local



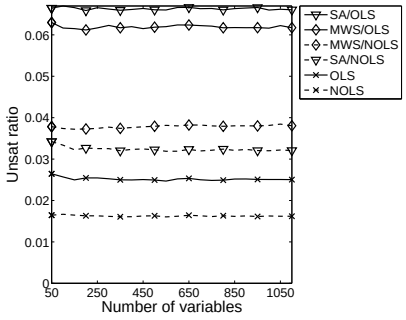
(a) Average completion time for the four determinstic algorithms.



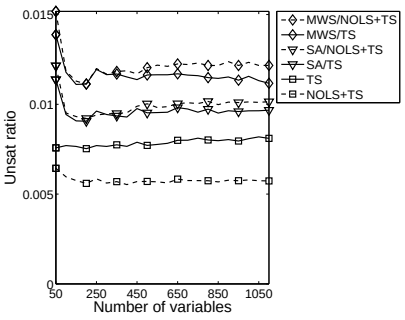
(b) Average completion time for the two stochastic algorithms.

Fig. 2. Average completion time for executing on random instances of Max-3-Sat.

search and the hybrid method might be efficient choices when only a "good" approximation is needed.



(a) Average performance when MWS is normalized with respect to OLS and NOLS



(b) Average performance when MWS is normalized with respect to TS and NOLS+TS

Fig. 3. In this experiment, algorithms run for exactly the same number of flips as the specified deterministic algorithm

We next consider weighted MAX-3-SAT instances. Here we fix the number of variables to be $n = 500$, and the number of clauses is again $4.25 \times n$. A random formula is generated, and then for each clause a weight value is chosen uniformly at random between one and a prescribed maximum weight value. The unsat ratio now refers to the ratio of the weight of unsatisfied clauses to the total weight of all clauses. This maximum weight will be the only parameter that varies in figure 4. As before, the performance of each algorithm is averaged over 500 trials. We observe that the performance of MWS now becomes dramatically better than

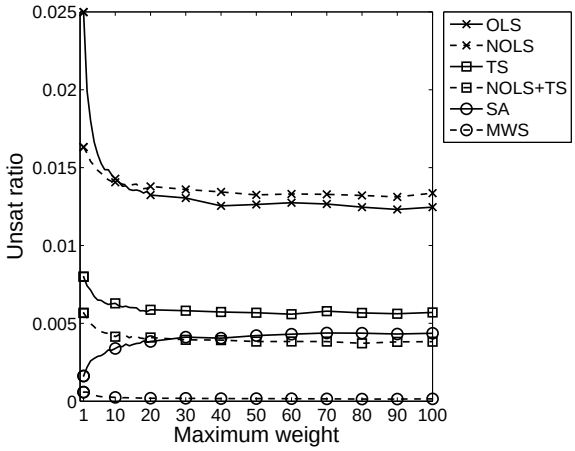
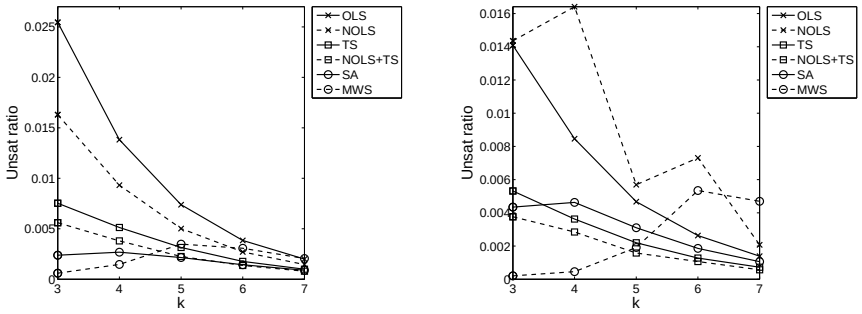


Fig. 4. Average performance when executing on random weighted instances of exact Max-3-Sat.

the other algorithms. As explained in section 2.5, MWS is designed to focus on weighted clauses and is successful in this regard. The unsat ratio is for the most part decreasing as a function of the specified max weight W having (for example) ratio .000245 at $W = 10$ and ratio .000153 at $W = 100$.

The performance of all deterministic methods and MWS improves as the maximum weight attainable in a formula increases from 1 to 30. In contrast, over the same range of maximum weight values the performance of simulated annealing declines, and as the weights grow, the deterministic hydrid method slightly outperforms SA although it remains significantly worse that MWS. Another phenomenon concerns relative performances of oblivious and non-oblivious local searche. Oblivious local search somewhat outperforms its non-oblivious counterpart in formulas with large enough weights. As the weights of clauses grow, the scaling weights used in the potential function have less and less effect, to the point that they hurt the performance of the algorithm.

Finally with regard to random SAT instances we consider Max-k-Sat for $k > 3$. Similar to the random Max-3-Sat instances, we choose the number of clauses to be $m = c \cdot n$ where c is slightly larger that c_k, the estimated threshold [15] for random k-SAT. Achlioptas et al [2] analyze how random 3-SAT differs from larger values of k as the landscape of satisfiable random k-SAT formulas fracture (into many small connected components) around the threshold value for larger values of k. We observe in figure 5 a dramatic change in the relative performance of algorithms as k increases. Of course, for large k, since a random assignment will (in expectation) satisfy all but a fraction $\frac{1}{2^k}$ of the clauses in an exact k-SAT formula, the unsat ratio approaches 0 as k grows. For $k \geq 5$, tabu search and the hybrid method outperform all other methods. As further evidence that the performace of MWS suffers as k increases, in figure **??** we consider weighted instances of Max-k-Sat with weights chosen uniformly in the



(a) Unweighted clauses (b) Maximum clause weight of 10

Fig. 5. Average performance when executing on random instances around the threshold of exact k-SAT ($3 \leq k \leq 7$).

range $[1, 10]$. Again in marked contrast to the weighted Max-3-Sat case, MWS begins to be outperformed at $k \geq 5$ and again tabu search and the hybrid method yield the best performance. The alternating performance (between even and odd k) performance of non oblivious local search is an interesting phenomena that is overcome when followed by tabu search.

Moving away from random SAT instances we considered the relative performance of algorithms on benchmark instances. In contrast to many of the results concerning random instances, MWS does not fare as well as SA or our hybrid algorithm. We ran the six algorithms on the benchmarks from the Second Evaluation of MAX-SAT Solvers (MAX-SAT 2007) and recorded how many times one technique improved over another one. The benchmarks contain instances generated in many different ways. Some are random just like the ones considered in the previous experiments, others were obtained by encoding different problems (for example, MAX-CUT) as an instance of the MAX-SAT problem. In contrast to many of the results concerning random instances, MWS does not fare as well as SA or our hybrid algorithm. In table 5 we see two off-diagonal zeros where one technique is subsumed by the other, namely, oblivious local search is a part of tabu search, and non-oblivious local search is a part of the hybrid algorithm. All the other off-diagonal entries are non-zero. For some instances even oblivious local search, arguably the weakest of the considered algorithms, improves over simulated annealing and MWS, arguably the strongest of the algorithms for random 3 SAT instances. The hybrid algorithm improves over the basic non-oblivious local search in most instances, which shows the usefulness of the tabu phase. As for the two major rivals, simulated annealing and the hybrid algorithm, their performances are similar with simulated annealing having an advantage. The hybrid method improves over oblivious local search. non-oblivious local search and MWS slightly more often than does simulated annealing while simulated annealing improves over tabu search more often than the hybrid method.

	OLS	NOLS	TS	NOLS+TS	SA	MWS
OLS	0	457	741	744	730	567
NOLS	160	0	720	750	705	504
TS	0	21	0	246	316	205
NOLS+TS	8	0	152	0	259	179
SA	30	50	189	219	0	185
MWS	205	261	453	478	455	0

Table 2. MAX-SAT 2007 benchmark results. Total number of instances is 815. The tallies in the table show for how many instances a technique from the column improves over the corresponding technique from the row.

6 Future work

We conclude with several open questions suggested by this work. A tight bound on the approximation or totality ratio of tabu search still requires closure. For all local search methods, rather than worst case approximation (totality) ratios, it would be more insightful to be able to computer expected ratios where the expectation is taken over random initial assignments. A more challenging direction is to provide theoretical results corresponding to the experiments from the second part of the paper. For example, what is the expected approximation ratio achieved by any of the deterministic local search based methods under a uniform random model of k SAT formulas with clause densities near the hypothesized threshold? In particular, for densities above the known algorithmic lower bound [12] can anything be said about the expected MAXSAT approximation? If the length of the taboo list is infinite, tabu search enters a cycle. What is the expected number of steps that tabu search makes before entering a cycle and what is the expected length of a cycle? Is there a theoretical explanation for why non-oblivious local search seems to provide such a subtantial improvement when used to initialize tabu search but does not seem to help (for example) MaxWalkSat.

References

1. E. Aarts and J. Lenstra, editors. *Local Search in Combinatorial Optimization*. Princeton University Press, second edition, 2003.
2. D. Achlioptas, A. Naor, and Y. Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435:759–764, 2005.
3. D. Achlioptas, A. Naor, and Y. Peres. On the maximum satisfiability of random formulas. *JACM*, 54(2), 2007.
4. D. Achlioptas and Y. Peres. The threshold for random k-sat is $2^k log 2 - o(k)$. *JAMS*, 17(2):947–973, 2004.
5. J. Argerlich, C. M. Li, F. Manya, and J. Planes. The first and second max-sat evaluations. *Journal on Satisfiability, Boolean Modeling and Computation*, 4:251–278, 09/2008 2008.
6. T. Asano and D. P. Williamson. Improved approximation algorithms for max sat. *J. Algorithms*, 42(1):173–202, 2002.

7. U. Feige and M. Goemans. Approximating the value of two power proof systems, with applications to max 2sat and max dicut. In *ISTCS '95: Proceedings of the 3rd Israel Symposium on the Theory of Computing Systems (ISTCS'95)*, page 182, Washington, DC, USA, 1995. IEEE Computer Society.
8. P. Hansen and B. Jaumard. Algorithms for the maximum satisfiability problem. *Computing*, 44(4):279–303, 1990.
9. J. Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.
10. F. Heras, J. Larrosa, S. de Givry, and T. Schiex. 2006 and 2007 max-sat evaluations: Contributed instances. *JSAT*, 4(2-4):239–250, 2008.
11. D. S. Johnson. Approximation algorithms for combinatorial problems. In *STOC '73: Proceedings of the fifth annual ACM symposium on Theory of computing*, pages 38–49, New York, NY, USA, 1973. ACM.
12. A. Kaporis, L. M. Kirousis, and E. G. Lalas. The probabilistic analysis of a greedy satisfiability algorithm. *Random Structures & Algorithms*, 28(4):444–480, 2006.
13. S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. *SIAM J. Comput.*, 28(1):164–191, 1999.
14. M. Mastrolilli and L. M. Gambardella. Max-2-sat: How good is tabu search in the worst-case? In *AAAI*, pages 173–178, 2004.
15. S. Mertens, M. Mezard, and R. Zecchina. Threshold values for random k-sat from the cavity method. *Random Structures & Algorithms*, 28(3):340–373, 2006.
16. D. M. Pennock and Q. F. Stout. Exploiting a theory of phase transitions in three-satisfiability problems. In *In Proc. AAAI '96*, pages 253–258. AAAI Press/MIT Press, 1996.
17. W. M. Spears. Simulated annealing for hard satisfiability problems. In *In, Workshop*, pages 533–558. American Mathematical Society, 1993.
18. H. Xu, R. A. Rutenbar, and K. Sakallah. sub-sat: a formulation for relaxed boolean satisfiability with applications in routing. In *ISPD '02: Proceedings of the 2002 international symposium on Physical design*, pages 182–187, New York, NY, USA, 2002. ACM.

7 Appendix

Appendix

There are many additional experiments that we performed that have not been included because of space limitations.

7.1 Many random initializations

A deterministic local search algorithm starts with an initial assignment and then converges to a local optimum. If the same method is initialized with many different random truth assignments, it may converge to better local optimum. To normalize the performance of our deterministic methods agianst that of Simulated Annealing, we ran the deterministic methods on many initial random assignments so as to match the number of flips by SA. iFor small n, this simple randomization does improve the performance of the considered algorithms, as Figure 6 shows. However, with growing n the search space becomes exponentially large. Each technique requires many executions with different initial truth assignments to explore such a search space with any thoroughness and it would appear that "good initializations" are not very dense in the space of all solutions. Simulated Annealing (and also MaxWalkSat) has a more intelligent guidance in its search through the solution space, so it uses computational resources efficiently. Figure 6 shows that SA achieves a better approximation ratio than any of the deterministic algorithms with multiple restarts for values of n larger than 200.

7.2 Lenthening the taboo list

One of the reasons why non-oblivious local search followed by tabu search failed to perform as well as MaxWalkSat in the previous experiment is that random restarts do not take into account previous search history. Tabu search algorithms have one extra parameter that may affect both the running time and the approximation ratio. This parameter is the length of taboo list. In our next experiment, the length of taboo list for the two tabu search methods is adjusted to guarantee that each technique performs at least as many computations as simulated annealing does.

Increasing the length of taboo list does not improve the performance of the algorithms. Recall that the version of tabu search considered in this paper records only recently flipped variables in the taboo list. Thus, the algorithm might enter a loop long before visiting every truth assignment in the search space. To see that this is exactly what happens in this experiment, we modified the stopping condition of tabu search, so that it terminates when a loop is detected. Theoretically this makes the worst-case running time exponential, although in practice the modified algorithms always terminated quickly. The altered tabu search has a longer running time than simulated annealing, but even for the formulas with 500 variables, the modified tabu search terminated in about 25,000,000 computations of the delta function, which is nowhere close to the worst-case running

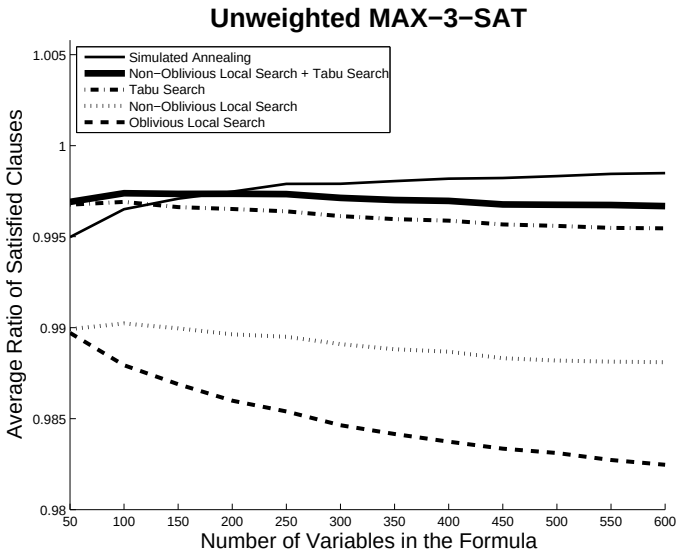


Fig. 6. Performance of simulated annealing versus other techniques with multiple restarts.

time of 2^{500}. The performance of the altered algorithms did not change compared to their standard implementations described in Section 2. This suggests that the length of taboo list does not provide a smooth tradeoff between the running time and the performance of a tabu search algorithm.