Towards Efficient Sampling: Exploiting Random Walk Strategies

Wei Wei, Jordan Erenrich, and Bart Selman
Department of Computer Science
Cornell University
Ithaca, NY 14853
{weiwei, erenrich, selman}@cs.cornell.edu

Abstract

From a computational perspective, there is a close connection between various probabilistic reasoning tasks and the problem of counting or sampling satisfying assignments of a propositional theory. We consider the question of whether state-of-the-art satisfiability procedures, based on random walk strategies, can be used to sample uniformly or near-uniformly from the space of satisfying assignments. We first show that random walk SAT procedures often do reach the full set of solutions of complex logical theories. Moreover, by interleaving random walk steps with Metropolis transitions, we also show how the sampling becomes near-uniform.

Introduction

We consider the problem of sampling from the set of satisfying assignments ("models") of a propositional theory. Perhaps the best known methods for sampling from combinatorial spaces are Markov Chain Monte Carlo (MCMC) methods, such as the Metropolis algorithm and simulated annealing (Madras 2002; Metropolis *et al.* 1953; Kirkpatrick *et al.* 1983). These methods are based on setting up a Markov chain with a predefined stationary distribution. One can draw samples from the stationary distribution by running the Markov Chain for a sufficiently long time. Unfortunately, on many hard combinatorial problems, the Markov chain takes exponential time, in terms of problem size, before reaching its stationary distribution.

These methods can also be used to find optimal solutions to combinatorial problems. For example, simulated annealing (SA) uses the Boltzmann distribution as the stationary distribution. By lowering the temperature to near zero, the distribution becomes concentrated around the minimum energy states, which correspond to the solutions of the combinatorial problem under consideration. For example, to find a satisfying assignment of a propositional theory, one can use an energy function that assigns to each truth assignment an energy (or cost) equal to the number of violated logical constraints. Zero cost solutions correspond to satisfying assignments.

Unfortunately, standard SA on a propositional theory is of little practical use, because on theories of practical interest, SA at low temperature does not reach the stationary distribution in a reasonable number of steps. Several alternative local search methods have been discovered that find assignments for large and complex propositional theories, involving up to a million variables and several million constraints. These methods exploit properties of random walks. We will refer to them as random walk strategies.

An intriguing question to consider is whether these random walk strategies can also be used *to sample uniformly or near uniformly from the set of satisfying assignments of a propositional theory.* If so, this would have interesting implications for a wide range of probabilistic reasoning tasks, because of the close connections between sampling models from a propositional theory and probabilistic inference. In particular, if one could sample propopositional models efficiently, then one obtains an efficient method for Bayesian inference and a range of other forms of probabilistic reasoning. The full technical details on the connections between sampling from a logical theory and probabilistic reasoning are not overly complicated but not the focus of this paper. We refer to, e.g., (Roth 1996; Littman *et al.* 2001; Park 2002).

We will show that one can indeed exploit ideas from random walk based methods to obtain effective near-uniform sampling of the models of certain propositional theories. More specifically, we show that a random walk strategy which incorporates a greedy bias, called WalkSAT, samples nearly uniformly on several structured propositional theories. We will also see that on hard random formulas, WalkSAT does find all assignments, but the sampling is far from uniform. We then consider a hybrid strategy which interleaves simulated annealing (SA) steps and WalkSAT steps. This hybrid strategy restores the sampling to near uniform. See the scatterplots in Figure 1 for a preview of this result. Each data point corresponds to a solution of the theory (2531 total). The vertical axis gives the frequency with which each solution was reached. The top panel shows that the ratio between least frequent and most frequently found solutions for WalkSAT is around a factor of 10^4; the bottom panel shows how interleaving simulated annealing steps reduces the non-uniformity by around a factor of 1,000 (a factor 10 between most frequent and least frequent remains). (Note that SA by itself can generally not reach any solution on these theories.) We also present a formal analysis in support of our empirical

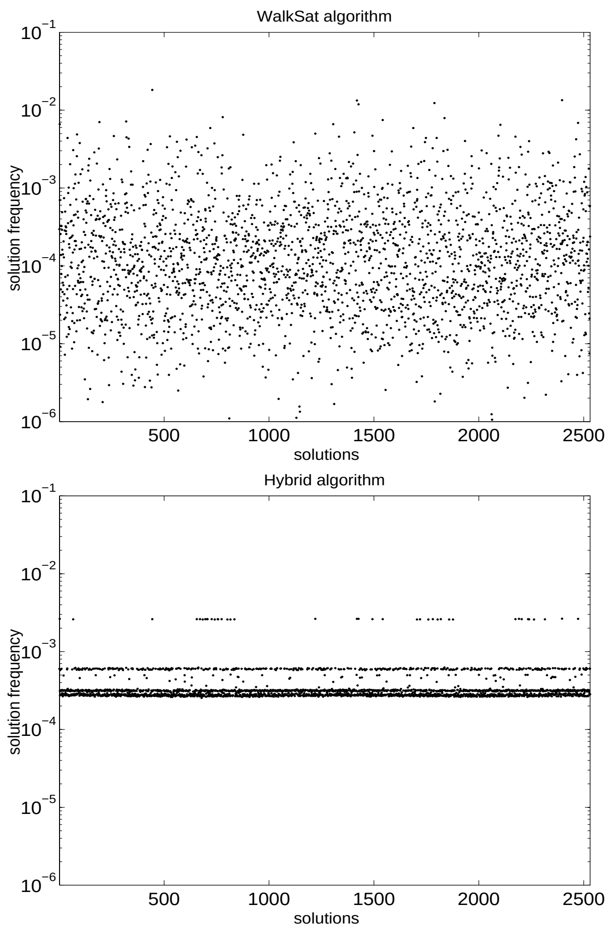

Figure 1: Sampling of hard random 3SAT instance. Top: Walksat. Bottom: Hybrid.

findings.

We believe that our results are a promising first step in the use of random walk strategies for sampling.

Random Walk Strategies

Without loss of generality, we will assume that our propositional theory is given in Conjunctive Normal Form (CNF). That is, the theory consist of a conjunction (logical "and") of disjunctions. Each disjunction (also called a "clause") consists of the logical "or" of a set of literals, where each literal is a Boolean variable or its negation. The goal is to find a setting to the Boolean variables such that there is at least one literal in each disjunction that evaluates to `true`.

The problem of determining whether a CNF formula has a satisfying assignment is also referred to as the satisfiability or SAT problem. SAT is the prototypical NP-complete problem (Cook 1971). Therefore, it is generally believed that in the worst-case one has to search through the full space of all 2^N assignments to find a satisfying assignment if one exists, where N is the number of Boolean variables of the theory. Because the SAT problem is NP-complete, it has a tremendous range of applications (Garey & Johnson 1979). The problem of counting the number of satisfying assignments is #P-complete (Valiant 1979), and believed to be significantly harder than NP-complete (again, worst-case).

We refer to a CNF formula with exactly k literals in each clause as k-CNF; the associated satisfiability problem is referred to as k-SAT. For $k \geq 3$, the satisfiability problem is NP-complete. However, 2-SAT can be solved in linear time. Interestingly, counting the number of 2-SAT assignments is #P-complete. This suggests that counting is significantly harder than determining satisfiability.

A *random walk procedure* for satisfiability is a deceptively simple search technique. Such a procedure starts with a random truth assignment. Assuming this randomly guessed assignment does not already satisfy the formula, one selects one of the unsatisfied clauses at random, and flips the truth assignment of one of the variables in that clause. This will cause the clause to become satisfied but, of course, one or more other clauses may become unsatisfied. Such flips are repeated until one reaches an assignment that satisfies all clauses or until a pre-defined maximum number of flips are made. This simple strategy can be surprisingly effective. In fact, Papadimitriou (1991) showed that a pure (unbiased) random walk on an arbitrary satisfiable 2SAT formula will reach a satisfying assignment in O(N^2) flips (with probability going to 1). More recently, Schoening (1999) showed that a series of short unbiased random walks on a 3-SAT problem will find a satisfying assignment in O(1.334^N) flips (assuming such an assignment exists), much better than O(2^N) for an exhaustive check of all assignments.

To gain some further insight into the behavior of random walk strategies for SAT, let us briefly consider an elegant argument introduced by Papadimitriou showing polytime behavior on 2SAT. Consider a satisfiable 2SAT formula F on N variables. Let T denote a satisfying assignment of F. The random walk procedure starts with a random truth assignment, T'. On average, this truth assignment will differ from T on the assignment of $N/2$ letters. Now, consider an unsatisfied clause in F (if no such clause exists, then T' is a satisfying assignment). Without loss of generality, we assume that the unsatisfied clause is of the form $(a \vee \neg b)$. Since this clause is unsatisfied, T' must assign both literals in the clause to False (which means that a is assigned False and b True). Also, the satisfying assignment T is such that at least one of the literals in the clause is assigned to True. Now, randomly select a variable in the clause and flip its truth value in T'. Since we have only two variables in the clause, we have at least a 50% chance of selecting the variable corresponding to the literal set to True in T. (Note that if T satisfies exactly one literal in the clause, we will select that literal with 0.5 probability. If T satisfies both literals, we select the "right" literal with probability 1.0.) It follows that with at least 50% chance, the Hamming distance of our new truth assignment to T will be reduced by 1, and with at most 50% chance, we will have picked the wrong variable and will have increased the Hamming distance to the satisfying assignment. Finally, we appeal to a basic result for one-dimensional symmetric random walks, which says that after L^2 steps, the random walker will on average have traveled a distance of L from the starting point. Given that our random walk starts a distance $N/2$ from the satisfying truth assignment T, after order N^2 steps, the walk will hit a satisfying assignment, with probability going to one. Note that although the walk may at first wander away from the satisfying assignment, it will "bounce off" the reflecting barrier at distance N. Also, our analysis is worst-case, *i.e.*, it holds for *any* satisfiable 2SAT formula.

It is instructive to consider what happens on a 3SAT formula, *i.e.*, a conjunctive normal form formula with 3 literals per clause. In this case, when flipping a variable selected at random from an unsatisfied clause, we may only have 1/3 chance of fixing the "correct" variable (assuming our satisfying assignment satisfies exactly one literal in each clause). This leads to a random walk heavily biased away from the solution under consideration. The theory of random walks tells us that reaching the satisfying assignment under such a bias would take an exponential number of flips. And, in fact, in practice we indeed see that a pure random walk on a hard random 3SAT formula performs very poorly.

However, we can counter this "negative" bias by considering the gradient in the overall objective function we are trying to minimize. The idea is that the gradient may provide the random walk with some additional information "pointing" towards the solution, and thus towards the right variable to flip. In SAT, we want to minimize the number of unsatisfied clauses. So, in selecting the variable to flip in an unsatisfied clause, we can bias our selection towards a variable that leads to the greatest decrease in the overall number of unsatisfied clauses. Introducing a bias of this form leads us to the WalkSAT algorithm and its variants. For the details of these algorithms, we refer to (Selman *et al.* 1994; McAllester *et al.* 1997). WalkSAT remains one of the most effective local-search style methods for SAT.

Table 1: Summary statistics from coverage experiments

INST.	RUNS ($\times 10^6$)	HITS RARE	HITS COM	RATIO WSAT	RATIO HYBRID
RAND.	50	53	9×10^5	17000	10
LOG.	1	84	4×10^3	50	17
VERIF.	1	45	318	7	4

Sampling results

Given the effectiveness of current random walk style procedures for finding satisfying assignments, we now turn our attention to the question of how well such procedures can sample from the set of all satisfying assignments of a logical theory. Note that since the theory behind random walk SAT procedures relies on hitting an extremal point (corresponding to a solution) as quickly as possible, there is no a priori guarantee that we will reach all solutions or that we will sample from solutions uniformly. Contrast this with an equilibrium based sampling approach such as SA: using the number of unsatisfied clauses as a cost function and with temperature going to zero, SA is guaranteed to sample uniformly from the satisfying assignments, provided it approaches the stationary distribution (which is of course a problem in practice).

We consider a hard random 3-SAT instance, a more structured problem from a planning application, as well as a structured instance from microprocessor verification domain. In order to generate a random 3-SAT instance, we first created 100 uniform random 3-SAT instances, with 70 variables and 301 clauses (Mitchell *et al.* 1992). We chose an instance with 2531 solutions for our detailed experiments. (Relsat (Bayardo & Pehaushek 2000) was used to systematically generate all solutions.) For the logistics planning formula, we started with *logistics.d.cnf* available from Satlib with over 10^{10} solutions. To facilitate solution set sampling experiments, we added 52 unit clauses, constraining the instance to 1600 solutions. For the verification domain, we chose a formula from Miroslav Velev's formula suite SSS-SAT.1.0. We added 287 unit clauses to reduce the number of solutions to 6144.

Table 1 shows our experimental results for sampling solutions of these three instances. For the random 3-SAT (see row marked "rand."), logistic (row marked "log."), and verification (row marked "verif.") instances, we performed 50×10^6, 1×10^6, and 1×10^6 runs of WalkSAT, respectively. The first key observation is that WalkSAT ultimately finds every solution for all instances. So, apparently, in terms of "solution set coverage", the random walk strategy works already quite well. In Table 1, the column marked "hits rare" gives the number of times the rarest solution was reached; "hits com." gives the number of times the most common solution was found. The column marked "ratio Wsat" gives the ratio of hits of the most common to the rarest solution for the WalkSAT algorithm. So, for example, on the random instance, we see that the most frequently visited solution is sampled 1.7×10^4 times more often than the satisfying assignment that is sampled the least frequent. The sampling of the verification and planning formulas by WalkSAT is much more uniform (ratio of 7 and 50, respectively). The last column of the table shows how we can still significantly improve the sampling ratio by using a Hybrid sampling strategy. We discuss this strategy in the next section.

The scatterplot in the top panel of Figure 1 gives the actual solution frequency (#hits/#runs) for each of the 2531 solutions of the random formula. (One data point per solution.) The figure further confirms the results from Table 1: WalkSAT's solution sampling can be highly non-uniform. Notice the log scale on the y-axis: 4 orders of magnitude variance. The Kullback-Leibler(KL) divergence (Bishop 1995) between this data and uniform distribution is 1.2431. (A similar figure for the logistic problem shows the points in a relatively narrow band.)

Improving uniformity: Hybrid strategy

In order to improve the uniformity of the sampling by WalkSAT, we explored a hybrid strategy. In this strategy, we select with probability p a random walk style move and with probability $1 - p$, a simulated annealing step. We used fixed temperature annealing (*i.e.*, Metropolis) moves (Sait & Youssef 2000).[1] More specifically, for each SA move, we

[1]The temperature was tuned by hand to obtain the most uniform sampling. We used T = 0.1. The hybrid strategy is not very

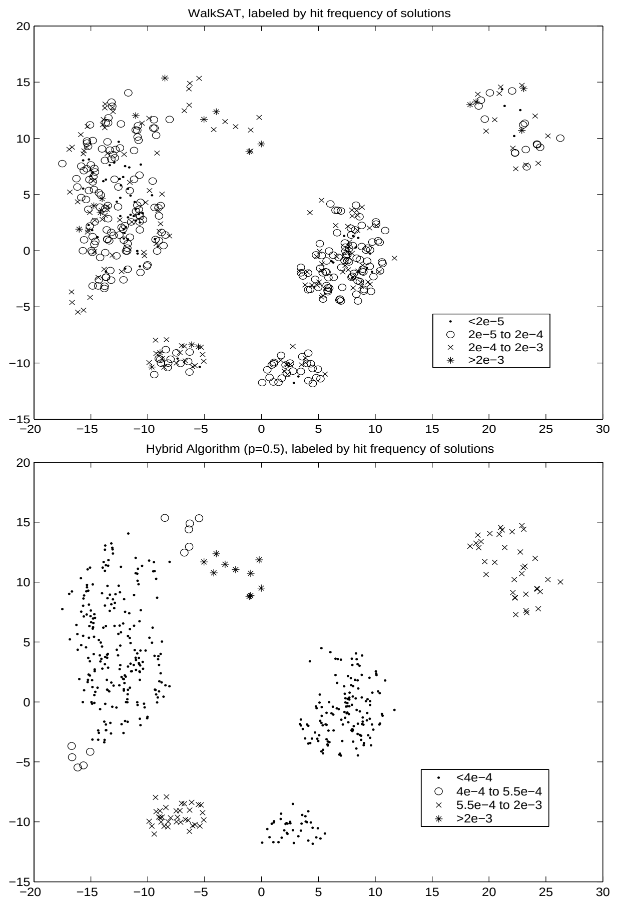

Figure 2: Solution space and the solution frequency for each solution.

consider a randomly selected neighbor of the current assignment. (A neighbor is an truth assignment that differs from the current truth assignment on a single letter.) When the neighbor satisfies the same or more clauses as the current assignment, the algorithm moves to the neighbor assignment. When the neighbor satisfies fewer clauses than the current assignment, we consider the $\Delta Cost$, which gives the decrease in the number of satisfied clauses. The algorithm moves with probability $e^{-\Delta Cost/Temp}$ to the neighbor (otherwise stays at the current assignment).

The bottom panel of Figure 1 gives our results. ($p = 0.5$; 5,000 steps of the mixed chain.) We see a dramatic improvement in the uniformity of sampling. The ratio between the most frequent and least frequently sampled solution is now around 10 (a factor 1000 improvement). The KL-divergence between the data and uniform distribution is reduced to 0.1495. The last column of Table 1 also shows significant improvements in uniformity for the two other domains. Apparently, the SA moves help make the sampling much more uniform. Note, however, that we do need WalkSAT transitions to actually reach solution states efficiently.

We further explored the interesting band structure of the bottom panel. We used a multi-dimensional scaling (Lee 2000) technique to project the high-dimensional solution space into two dimensions. Figure 2 shows the solution space with probabilities to reach each solution by WalkSAT and the Hybrid algorithm. (Not all datapoints are printed to make the figure readable in black and white.) Generally, the closer two solutions appear in the Figure 2, the closer they are in Hamming distance. We can see from the figure that the solutions form clusters. For the WalkSAT algorithm, the number of times a solution is reached varies considerably from solution to solution, even within a cluster. However, for the Hybrid algorithm, solutions within the same cluster are sampled with a similar frequency. This suggests that a Hybrid algorithm can move around a cluster quite effectively, thereby "smoothing" out the probabilities. Overall, the Hybrid method samples much more uniformly than WalkSAT by itself. (SA samples even less effective, because it often does not reach any solution.)

sensitive to the temperature setting.

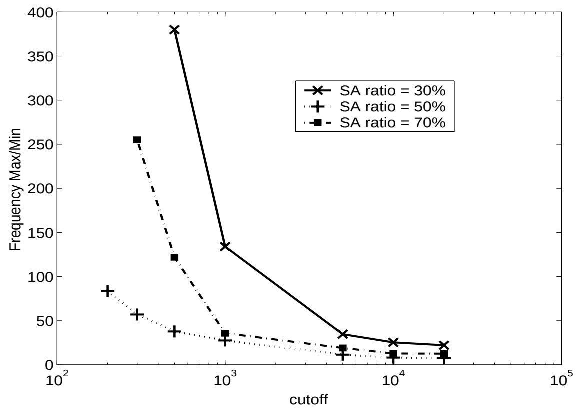

Figure 3: Ratio of least frequent versus most frequently sampled solution for hybrid.

In Figure 3, we show how the effectiveness of the sampling depends on the choice of p. We give the ratio of the most frequent solution to the least frequent solution as a function of the number of steps taking in the hybrid chain. The figure shows that there is a clear optimal ratio: 50% SA moves is better than 30% or 70%. In fact, for 30% and 70%, it becomes difficult to measure the ratio when the chain runs only for a small number of steps (less than 200), because not all solutions are found on such short runs even after a million runs.

Analysis

In this section, we present some analysis that illuminates the difference in sampling using a standard MCMC approach, such as SA, versus a random walk based strategy.

First, it is useful to highlight the fundamental difference between a Metropolis style algorithm and a random walk procedure for SAT. As we discussed, Papadimitriou showed that a random walk procedure for SAT will find a solution to any satisfiable 2CNF formula in $O(N^2)$ time, where N is the number of variables in the formula. This is not the case for SA. (We will use SA as a prototypical example of an MCMC method.)

Consider a formula $F^\star$ of the following form: $a \vee c_1, a \vee c_2, \ldots a \vee c_n, \neg a \vee b, \neg a \vee \neg b$.

Theorem 1 *SA at fixed temperature with probability going to 1 takes time exponential in n to find a satisfying assignment of $F^\star$.*

The details of the proof are somewhat tedious. Informally, the argument goes as follows. There are two satisfying assignments of the formula. They both have a assigned to `False`, in order to satisfy the last two clauses, and c_1 through c_n assigned to `True` (b can be assigned arbitrarily). We start with a random assignment. First assume that a is assigned `True` in this assignment. Any proposed flips of the variables c_1 through c_n will be accepted by SA because they do not create any new unsatisfied clauses (cost function does not increase). Flipping of c_1 through c_n amounts to a random walk on an n-dimensional hypercube formed by these variables. In such a walk, the probability of reaching an area where only a small number of variables, say, $\sqrt{n}$ of them, are assigned `False` is exponentially vanishing. With a non-vanishing fraction of the variables in c_1 through c_n assigned `False`, the probability that a proposal to flip a to `False` will be accepted is exponentially small (note that temperature is fixed and n goes to infinity), and finding the satisfying assignment takes exponential time. Informally, we are stuck in an area with local minima defined by a set to `True`. If the search starts with a assigned to `False`, with probability going to 1, it will be flipped to `True` in a polynomial number of flips, getting us to the previous scenario.

A random walk strategy finds a solution for $F^\star$ efficiently, since it is in 2CNF. Moreover, it will find the two satisfying assignments with equal probability due to symmetry in the formula. However, symmetry does of course not exist in all formulas. In fact, when the solution space is very asymmetrical, a random walk strategy can have vastly different probabilities of reaching different solutions. This is shown by considering the following formula $F^{\star\star}$: $x \vee y_1, x \vee y_2, \cdots x \vee y_n, \neg y_n \vee y_1, \neg y_1 \vee y_2, \neg y_2 \vee y_3, \cdots \neg y_{n-1} \vee y_n$. The final n clauses require y_1 through y_n to adopt the same value. Therefore, this formula has exactly three assignments. They are $S_1 : x = 1, y_1 = y_2 = \cdots y_n = 0$, $S_2 : x = 1, y_1 = y_2 = \cdots y_n = 1$, and $S_3 : x = 0, y_1 = y_2 = \cdots y_n = 1$. (0 denotes `False` and 1 denotes `True`.) We will show that in a pure random walk algorithm, the last assignment is reached with an exponentially small probability. (The other two solutions are symmetric and are reached with probability $0.5(1 - \epsilon)$, where ϵ is the probability of reaching S_3.)

Theorem 2 *Pure random walk strategy on $F^{\star\star}$ reaches assignment S_3 with an exponentially small probability.*

One observation is that during pure random walk search on $F^{\star\star}$, if variable x is ever assigned `True`, it will immediately satisfy all clauses in which it appears, so x will never be flipped to `False` again during the entire search. To show that a solution with x assigned `False` is rare, we only need to consider the case where x is initially assigned `False`.

At the beginning of the search, we can expect about $\frac{n}{2}$ variables among y_1 through y_n to be assigned `True`. Now we need to calculate the probability that the algorithm flips all of these n variables to `True`, while not flipping the values of x. Note that during the search, if the $(n + i)$th clause is unsatisfied, variable y_i must be `False`. Since x is also `False`, the ith clause is also unsatisfied. This implies that before x is flipped, at least half of the unsatisfied clauses come from the first n clauses. The pure random walk algorithm at each step first picks an unsatisfied clause at random, then flips a random variable in that clause. So at each step, there is at least $\frac{1}{4}$ probability that variable x is flipped. Since the number of steps needed is greater than $\frac{n}{2}$, the probability that x is kept `False` in all these steps is less than $(\frac{3}{4})^{-\frac{n}{2}}$. This means that the probability of reaching S_3 with x set to `False` is exponentially small.

So, in principle, MCMC methods such as SA sample uniformly from states with an equal cost value (number of unsatisfied clauses). However, Thm. 1, shows an example where reaching the stationary distribution (assume low temperature SA) takes exponential time, even on a 2-CNF formula. On the other hand, Random Walk (and its extension, WalkSAT) is much more effective at finding solutions but may sample quite non-uniformly (Thm. 2). This leads us to our Hybrid strategy introduced earlier. As we discussed, our empirical results show that the Hybrid strategy is indeed quite promising. Let us now try to obtain a better understanding as to why the hybrid strategy may work well. (At this point we do not have a full rigorous analysis of the hybrid strategy.)

By observing runs of the Hybrid algorithm, we found that the behavior can be characterized by two phases. Before the search finds a solution, the random walk strategy plays the dominant role. In this first phase, it is important to reach some solution. In Table 2, we compare the average number of steps it takes WalkSAT, Hybrid and SA algorithms to reach the first solution state. It shows that the Hybrid numbers are closer to the WalkSat numbers, and relatively far from the SA numbers. This is the first phase. (The table provides a good illustration of the limitations of a pure MCMC approach: it simply takes too long to reach any satisfying assignment.)

Table 2: Flips needed to reach first solution.

INSTANCE	WALKSAT	HYBRID $p = 0.5$	SA
RANDOM	382	677	24667
LOGISTIC	5.7×10^5	15.5×10^5	$> 10^9$
VERIF.	36	65	10821

After the algorithm reaches a solution, the search enters phase two where the SA moves dominate. In the second phase, the algorithm explores clusters of solutions. A cluster is a set of connected solutions, so that any two solutions within a cluster can be connected through a series of flips without leaving the cluster. In many domains of interest, solutions usually exist in clusters (Mézard *et al.* 2002). In phase two the algorithm behaves like SA because after the search enters a cluster of solutions, unsatisfied constrains no longer exist, and the random walk portion of the search is effectively turned off until an uphill move brings the search out of the cluster. Because SA has good properties in exploring a connected space, it samples near-uniformly and in practice often effectively explores the neighboring solution states.

SA has the following useful property when exploring a cluster of solutions:

Theorem 3 *SA at zero temperature samples all solutions within a cluster uniformly.*

Consider the Markov Chain formed inside a cluster of satisfying assignments. For a node (satisfying assignment) i, the transition probability from i to neighboring node j inside the cluster is $\frac{1}{n}$. If the selected neighbor in outside the cluster, however, SA will not make the move since temperature is 0. Therefore there is a self cycle with probability $\frac{n-d_i}{n}$ for node i, where d_i is the degree of node i inside its cluster. (In other words, d_i is the number of neighboring assignments (one truth value "flipped") of the current state that are also satisfying assignments.) This Markov chain is ergodic.

More formally, the transition matrix of the chain is $P = (p_{ij})$, where

$$p_{ij} = \begin{cases} \frac{1}{n}, & \text{if } i \text{ and } j \text{ neighbors in cluster;} \\ \frac{n-d_i}{n}, & \text{if } i = j; \\ 0, & \text{otherwise.} \end{cases}$$

The ith entry of $q \cdot P$ is $\frac{1}{n}\sum_{k=1}^{n} p_{ki} = \frac{1}{n}(d_i \cdot \frac{1}{n} + \frac{n-d_i}{n}) = \frac{1}{n}$ for all i. Therefore, $q = (\frac{1}{n}, \frac{1}{n}, \cdots, \frac{1}{n})$ is the stationary distribution of the underlying Markov chain.

A key question is how fast the stationary distribution is reached. This is determined by the mixing time. By using a Fourier analysis of the eigenvalues, one can show that the mixing time on a full hybercube is $O(n \log n)$, where n is the number of dimensions. So, sampling from a full hypercube is very efficient. However, a cluster of satisfying assignments forms a subset of the hypercube and rigorous analysis of the mixing time of a subset of the nodes of a hypercube is beyond current analysis (Morris & Sinclair 1999). Our empirical observations suggest that SA samples quite effectively from the solution clusters.

Given the algorithm's ability to sample from a cluster nearly uniformly, the probability mass of a solution is very near to the ratio of the probability of reaching the cluster that the solution belongs to and the size of that cluster. Let C denote a cluster of solutions, P_C denote the probability of reaching the cluster, and $|C|$ denote the size of C. We have for the probability P_a of reaching assignment a in cluster C

$$\forall a \in C, P_a \simeq \frac{P_C}{|C|}$$

Ideally, if P_C is proportional to $|C|$ for all clusters, this approach would achieve perfect sampling. However, this is not quite the case, which explains why our current hybrid does not sample fully uniformly. In particular, our empirical results show that larger clusters are reached with a higher probability, but not fully proportionally to their size. We are currently exploring other hybrid strategies that may further increase the uniformity of the sampling process.

Conclusions

We have considered the problem of sampling satisfying assignments of Boolean satisfiability problems. This problem is closely related to various probabilistic reasoning tasks, and is believed to be much harder than the already intractable problem of finding a single satisfying assignment.

We have shown how random walk style procedures provide a promising technique for sampling from the set of satisfying assignments. These procedures can reach solution clusters for much larger problem instances than standard MCMC methods. Moreover, by using a hybrid strategy, we demonstrated how one can get reasonably uniform sampling in several problem domains. Our findings point to a promising research direction for the development of a new class of sampling techniques that combine random-walk style procedures with MCMC methods.

An interesting direction for future research is to adapt complete search methods, such as Davis-Putnam style procedures, for sampling. We are currently exploring such an approach.

Acknowledgements

This work was supported in part by an Alfred P. Sloan Research Fellowship, a National Science Foundation Information Technology Research Grant IIS-0312910, and the Intelligent Information System Institute at Cornell University sponsored by Air Force Office of Scientific Research Grant F49620-01-1-0076.

References

B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, vol. 8, no. 3, pp. 121–123, 1979.

R. J. Bayardo, Jr. and J. D. Pehoushek. Counting Models Using Connected Components. In *Proc. AAAI-00*, 2000

C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, New York, pp. 58-59, 1995

S. A. Cook. The complexity of theorem proving procedures. In *Proc. Third Annual ACM Symposium on the Theory of Computing*, 1971.

M. R. Garey and D. S. Johnson. *Computers and Intractability*, W. H. Freeman and Company, 1979.

M. R. Jerrum, L. G. Valiant, V. V. Vazirani. Random Generation of Combinatorial Structures From a Uniform Distribution. *Theoretical Computer Science* 43, 169-188, 1986.

H. Kautz and B. Selman. Unifying SAT-based and Graph-based Planning. In *Proceedings of IJCAI-99*, Stockholm, 1999.

S. Kirkpatrick, C. D. Gelatt, Jr., M.P. Vecchi. Optimization by Simulated Annealing. *Science*, Number 4598, 1983.

M. D. Lee. Generating Multidimensional Scaling Representations. Software. Home page Michael Lee (software).

M. L. Littman, S. M. Majercik and T. Pitassi. Stochastic Boolean satisfiability. *Journal of Automated Reasoning*, Vol. 27, No. 3, 251-296, 2001.

N. Madras. *Lectures on Monte Carlo Methods*. Field Institute Monographs, Vol 16, 2002.

D. A. McAllester, B. Selman, and H. A. Kautz. Evidence for Invariants in Local Search. In *Proceedings of AAAI-97*, pages 321-326, 1997.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller and E. Teller. Equations of State Calculations by Fast Computing Machines. *J. Chem, Phys.* 21, 1087-1093, 1953.

M. Mézard, G. Parisi, and R. Zecchina. Analytic and Algorithmic Solution of Random Satisfiability Problems. *Science* 297, 812-815, 2002.

D. G. Mitchell, B. Selman, and H. J. Levesque. Hard and Easy Distributions of SAT Problems. In *Proceedings AAAI-92*, pages 459-465, 1992.

B. Morris, A. Sinclair. Random Walks on Truncated Cubes and Sampling 0-1 Knapsack Solutions. In *Proceedings FOCS-99*, pages 230-240, 1999.

C. H. Papadimitriou. On Selecting a Satisfying Truth Assignment. In *Proceedings of the Conference on the Foundations of Computer Science*, pages 163-169, 1991.

J. D. Park. MAP complexity results and approximation methods. In *Proceedings of the 18th Annual Conference on Uncertainty in AI (UAI)*, pages 388–396, 2002.

D. Roth. On the Hardness of Approximate Reasoning. *Artificial Intelligence* 82, 273-302, 1996.

S. M. Sait, H. Youssef, *Iterative Computer Algorithms with Applications in Engineering: Solving Combinatorial Optimization Problems*, Wiley-IEEE Computer Society Press, 2000.

U. Schoening. A Probabilistic Algorithm for k-SAT and Constraint Satisfaction Problems. In *Proceeedings of FOCS*, 1999.

B. Selman, H. Kautz, and B. Cohen. Local Search Strategies for Satisfiability Testing. *2nd DIMACS Challenge on Cliques, Coloring and Satisfiability*, 1994.

A. Sinclair. *Algorithms for Random Generation and Counting*. Birkhäuser, Boston. 1993.

L. G. Valiant. The Complexity of Enumeration and Reliability Problems. *SIAM Journal on Computing*, Vol. 8, No. 3, 410-421, 1979.

W. Wei and B. Selman. Accelerating Random Walks. In *Proc. CP-02*, 2002.