Auto-Walksat: A Self-Tuning Implementation of *Walksat*

Donald J. Patterson [1]

University of Washington, Department of Computer Science and Engineering, Box 352350, Seattle WA 98195-2350

Henry Kautz

University of Washington, Department of Computer Science and Engineering, Box 352350, Seattle WA 98195-2350

Abstract

Stochastic search algorithms have proven to be very fast at solving many satisfiability problems [2,3,8]. The nature of their search requires careful parameter tuning to maximize performance, but depending on the problem and the details of the stochastic algorithm, the correct tuning may be difficult to ascertain [9]. In this paper we introduce *Auto-Walksat*, a general algorithm which automatically tunes any variant of the *Walksat* family of stochastic satisfiability solvers. We demonstrate *Auto-Walksat*'s success in tuning *Walksat*-SKC to the DIMACS benchmark problems with negligible additional overhead.

Key words: Satisfiability testing, invariant ratio, local search, stochastic algorithms, parameter tuning, Walksat, Auto-Walksat

1 Introduction

The ability of stochastic satisfiability solvers to successfully find a problem's solution depends on how the trade-off between random decisions and heuristic decisions is managed during the solution search [5–7]. This trade-off is controlled by a parameter setting, typically called the *noise*, which ranges from 0% to 100%. The optimal noise setting can vary greatly depending on the

[1] This work was supported by a National Defense Science and Engineering Graduate Fellowship, USA

specifics of the algorithm used and the problem being solved. For a particularly hard problem, whose solution is unknown, it would be very useful to know the optimal noise setting. If this were known, computational resources could be effectively allocated to solving the problem instance. This would result in less wasted computing cycles and a higher chance of producing a solution.

McAllester [5] presented empirical evidence that the optimal noise setting for stochastic satisfiability solvers is correlated to measurable properties of a given algorithm/problem pair. This paper describes our attempts to exploit these observations to improve the robustness of stochastic satisfiability solvers. We present an algorithm that uses a variant of *Walksat* [9] to probe the parameter space of noise settings for the value which will maximize the probability of finding a solution. Each probe takes a negligible amount of time compared to a complete run of *Walksat* on the problem instance.

The remainder of the paper is divided into five sections. Section 2 discusses the variant of *Walksat* that we used in this research. Section 3 discusses the minimization technique that *Auto-Walksat* applies. Sections 4 and 5 present and discuss the results of applying *Auto-Walksat* to the DIMACS [2] set of benchmark problems. Finally, we draw our conclusions in Section 6.

2 Background

We are concerned with solving Boolean satisfiability problems in conjunctive normal form (CNF). This problem can be described by a formula which is a conjunction of clauses. Each clause is a disjunction of literals, and each literal is a propositional variable or its negation. Solving such a formula consists of determining an assignment of truth values for each variable such that the formula evaluates to true.

Walksat is a family of stochastic algorithms [8] that assigns all variables a random truth assignment and then attempts to heuristically refine the assignment until the CNF formula evaluates to true. The specific method of varying the truth assignment defines the variant of *Walksat*. All variants share the common behavior of occasionally ignoring their heuristic and making a random refinement according to some fixed probability. *Walksat* algorithms are in general sound, but not complete.

In our validation of *Auto-Walksat* we used a variant called *Walksat*-SKC. Figure 1 briefly describes this algorithm.

The *objective function value* of a stochastic algorithm is defined as the value

[2] ftp://dimacs.rutgers.edu/pub/challenge

(1) Given a problem instance, F, in CNF, and a noise setting, N, choose a random truth assignment, S.
(2) **While** $F(S)$ evaluates to false and you have not reached a maximum number of iterations ("flips"):
   (a) Randomly choose an unsatisfied clause $c \in F(S)$.
   (b) **If** there is a literal in c whose value can be changed without causing any new clauses to become unsatisfied, let l be this literal.
   (c) **Else**
      (i) With probability N choose l in c randomly.
      (ii) With probability $1 - N$ apply the following heuristic: Choose l such that when its value is changed, the smallest number of satisfied clauses in $F(S)$ become unsatisfied.
   (d) Change the truth assignment of l in S.

Fig. 1. The *Walksat*-SKC algorithm

that is to be minimized during the search for a truth assignment. In *Walksat*-SKC, the objective function value is the number of unsatisfied clauses. The *mean objective function value* is defined as the average of the objective function value calculated at the end of several unsuccessful attempts to solve a formula. Each attempt is made with a different random seed. The *standard deviation of the objective function* is defined as the standard deviation of the objective function value over several unsuccessful attempts to solve a formula. The *invariant ratio* is the mean objective function value over the standard deviation of the objective function. McAllester [5] observed that the optimal performance of several stochastic algorithms occurs when the noise level is approximately ten percent above that which minimizes the invariant ratio. *Auto-Walksat* exploits this observation to estimate the optimal noise setting for a given problem and algorithm.

3 Minimization of the Objective Function

The invariant ratio is a practical value to minimize because it can be accurately determined without solving the satisfiability formula. By merely *probing* the problem, or attempting to solve the problem with a very small number of iterations, it is possible to empirically determine values of the invariant ratio. As part of a preprocessing phase, these values can be used to guide a search for the minimum invariant ratio which in turn leads to an estimated optimal noise level. The provided stochastic algorithm can then rigorously apply this noise level to the problem.

Auto-Walksat uses *Brent's method* [1] to adaptively search the invariant ratio space for the global minimum. This technique combines recursive brack-

Definition 1 ($Probe(F, A(N))$) *: Given a CNF problem F and an algorithm A, $Probe(F, A(N))$ calculates the invariant ratio at a noise level of N. The $Probe$ function attempts to solve the problem several times, collecting objective function statistics after executing 2000 flips plus one flip for every atom in F. $Probe$ terminates when the 95% confidence interval of the objective function mean is less than 0.05[3].*

(1) Let $left = 0$. P_{left} is undefined.
(2) Let $x = 0$. Let $P_x = Probe(F, A(x))$.
(3) Let $right = 100$. P_{right} is undefined. (By our assumptions we know that the minimum must lie between $left$ and $right$.)
(4) **While** the minimum is not tightly bracketed, find the next noise level to probe, n, as follows:
   (a) **Parabolic Interpolation**:
      If the parabola formed by ($left$,P_{left}), (x,P_x), ($right$,P_{right}) is well-formed let n be its minimum.
   (b) **Bracketed Search** (if the previous step fails):
      Choose n such that it is a golden section step from x toward $left$ if $(x - left) > (right - x)$ and toward $right$ otherwise. Intuitively this is the most effective recursive split that can be made as the number of levels of recursion goes to infinity. The motivation for this is discussed in detail in [1].
   (c) Let $P_n = Probe(F, A(n))$.
   (d) Reassign $left$,x,$right$ such that the bracket surrounding the minimum shrinks.
(5) Run $A(n)$

Fig. 2. The *Auto-Walksat* algorithm

eted search (Golden Section Search) and parabolic interpolation. In general, bracketed search is sufficient to find the minimum, but, in practice, including parabolic interpolation speeds up the search. Parabolic interpolation is not sufficient by itself because it is not robust to changes in the invariant ratio which are caused by the stochastic nature of *Walksat*. Using bracketed search as a fall-back adds the necessary robustness. Recursive improvements are made until successive refinements are less than 1% of the current value.

Figure 2 describes the *Auto-Walksat* algorithm and figure 3 graphically demonstrates steps (1)–(4) in three problem domains. In all three domains *Auto-Walksat* quickly converges, iterating until the solution is sufficiently refined. Despite the wide variety of the problem domains, the invariant ratio curve is well-formed and well-suited to bracketed search and parabolic interpolation. Vertical perturbations in the probed values, which are caused by the stochastic nature of the probes, do not adversely affect the value of the noise level to which *Auto-Walksat* converges.

Despite the complexity of completely solving these three problems, *Auto-*

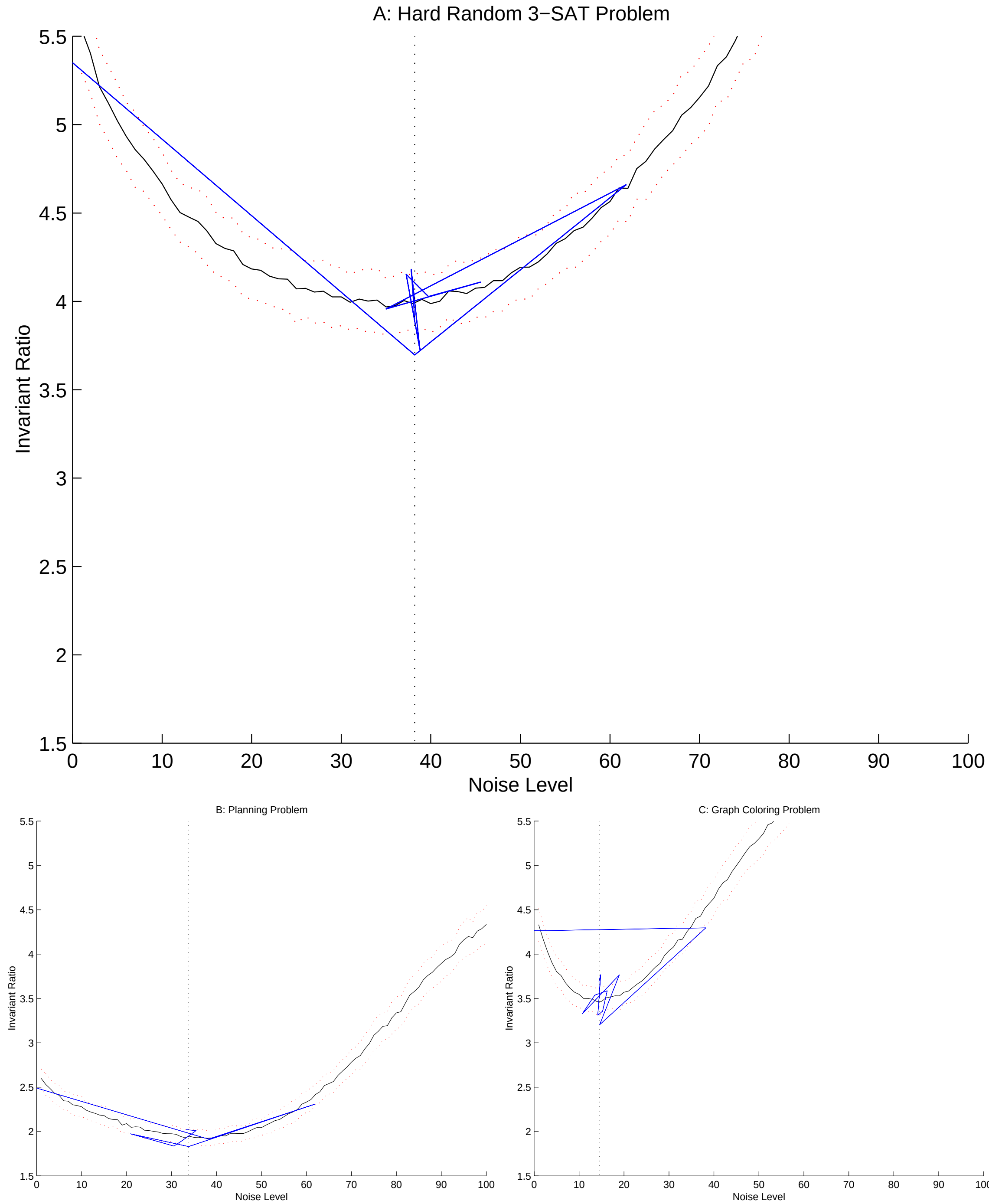

Fig. 3. Visualization of *Auto-Walksat* preprocessing on three CNF problems. In all three graphs, the solid line bordered by the dotted lines indicates the mean $\pm$ one standard deviation for 100 probes at each noise level and were generated for visualization purposes. The jagged solid lines are traces of *Auto-Walksat*'s minimization process. Each trace begins with a probe at $noise = 0\%$ and concludes at the indicated noise setting. A) A hard random 3-SAT problem with 4250 clauses and 1000 variables [4]. Estimated optimal noise setting was 39%. B) A Planning problem (huge.cnf) with 459 clauses and 7054 variables. Estimated optimal noise setting was 33%. C) A graph coloring problem (gr_125.17.cnf) with 2125 clauses and 66272 variables. Estimated optimal noise setting was 14%

Walksat was only required to probe 9–12 times to minimize the noise invariant. This corresponds to an additional overhead of approximately one minute (2 million flips) and is negligible compared to the possible time required to solve any given problem.

4 Results

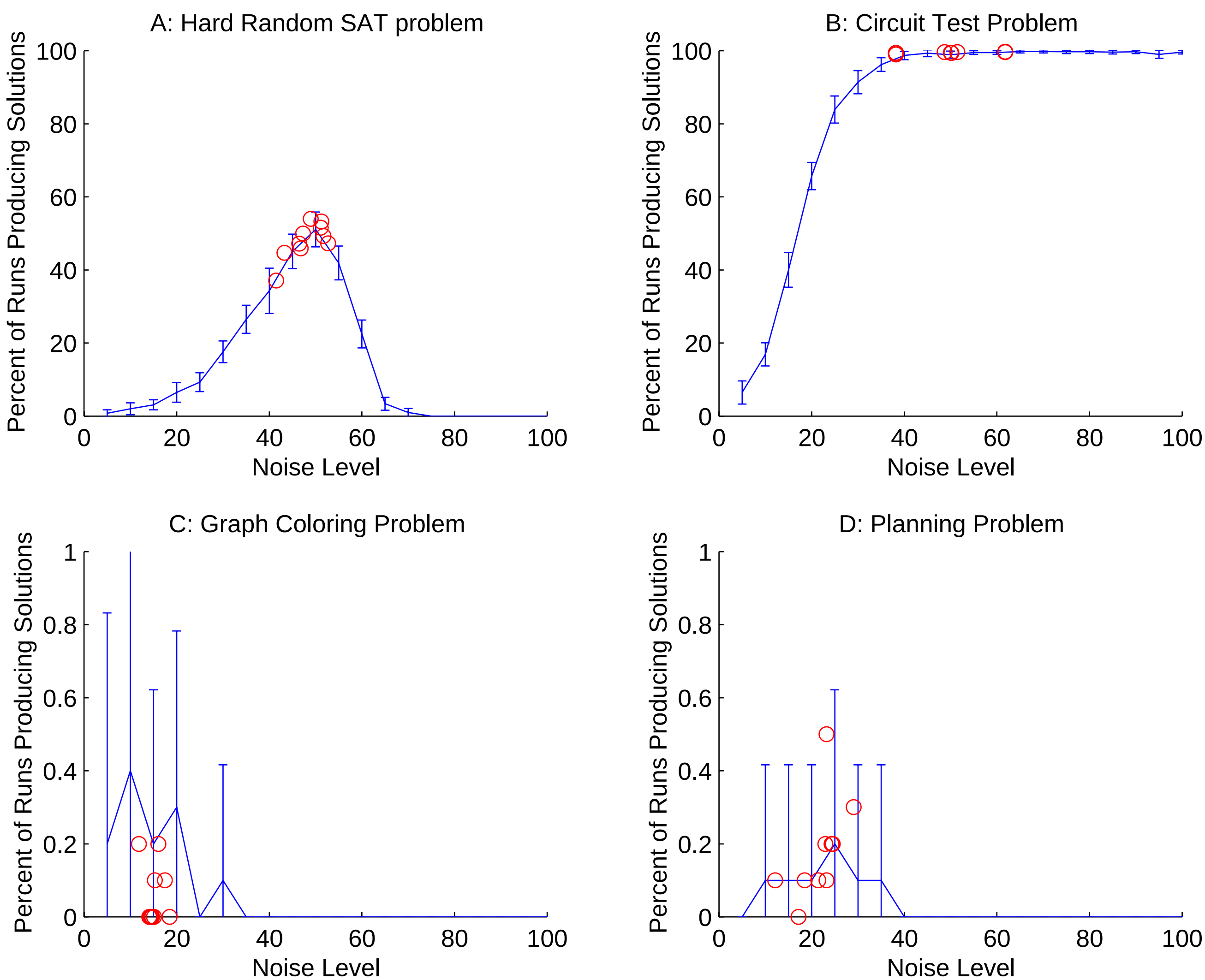

Fig. 4. Results of running *Auto-Walksat* with *Walksat*-SKC on four DIMACS benchmark problems. A) f600 B) SSA-7552-15-8 C) g125 D) bw-large-c . The solid line shows the result of running *Walksat*-SKC exhaustively with a variety of noise levels and is included for visualization purposes. Each noise level was sampled 10 times and the mean is plotted with error bars indicating ± one standard deviation. Each circle represents a complete run of *Auto-Walksat* on the same problem. In all cases, *Auto-Walksat* estimated the optimal noise setting by probing and then did a complete *Walksat*-SKC run with 1000 restarts to find solutions. In all cases the maximum number of flips per restart was 100,000. The percent of restarts which produced solutions are indicated.

Figure 4 shows the result of running *Walksat*-SKC and *Auto-Walksat* on four different DIMACS benchmark problems. In all four graphs a poorly chosen noise level could lead to equally poor performance. However, the proper noise level is not consistent between problems. In the case of figures 4-C and 4-D, noise settings greater than 40% would produce no solutions at all. On the other hand anything less than 40% in figure 4-B is suboptimal. Without any knowledge of the problem domain and with little extra overhead, *Auto-Walksat* was able to empirically choose a noise setting which maximized the chance of finding a solution. Previously this was only possible with careful hand tuning of parameters.

5 Discussion

The left graph of figure 5 shows the relationship between the invariant ratio and performance, parameterized by noise level. In the diagram on the left, maximizing performance requires estimating the point on the curve which minimizes the invariant ratio, noting the noise level, then slightly increasing the noise to maximize performance. McAllester suggested a 10% increase [5]. In all of the DIMACS benchmark problems that were sensitive to the noise setting, this was successful.

Our results indicated two cases in which this technique failed for *Auto-Walksat*. The first is in problems which are intrinsically difficult for *Walksat*-SKC to solve. Selman [8] demonstrates that there are some problems that are pathological for *Walksat* and can only be solved by aggressively tuning several parameters, if they can be solved at all. While this is not a failure of *Auto-Walksat* per se, *Auto-Walksat* provides no additional value in this case.

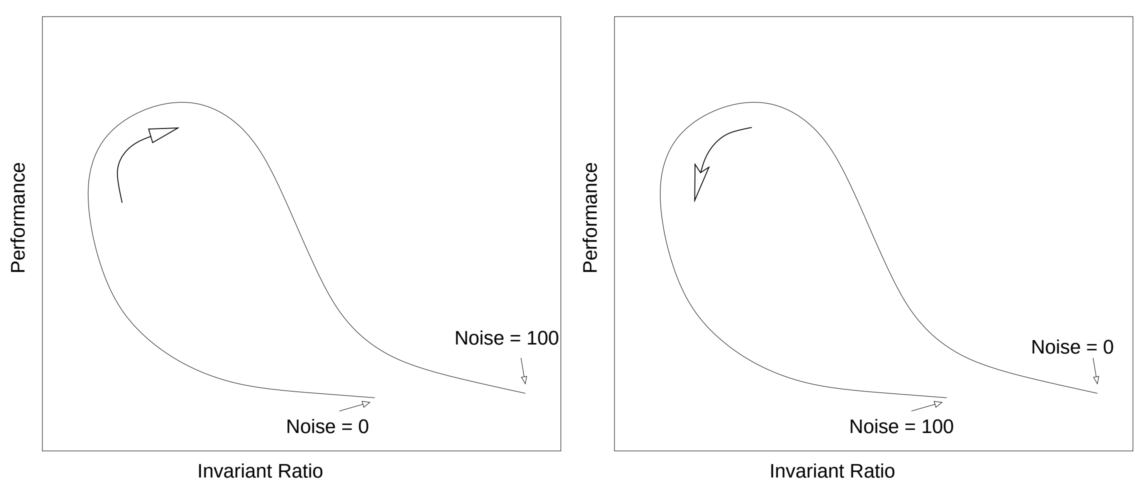

Fig. 5. Two possible relations between noise, performance and the invariant ratio. The left graph shows a generalization of data that McAllester presented in [5].

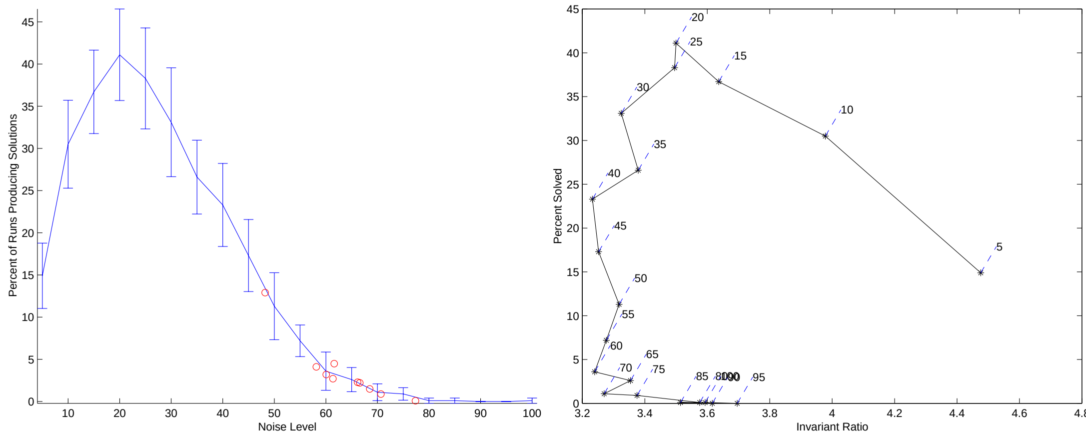

Fig. 6. Results of running *Auto-Walksat* with *Walksat*-SKC on a logistics problem (logistics.c.cnf). In the left graph, the solid line shows the result of running *Walksat*-SKC exhaustively with a variety of noise levels and is included for visualization purposes. Each noise level was sampled 10 times and the mean is plotted with error bars indicating ± one standard deviation. Each circle represents a complete run of *Auto-Walksat* on the same problem. In all cases, *Auto-Walksat* estimated the optimal noise setting by probing and then did a complete *Walksat*-SKC run with 1000 restarts to find solutions. The percent of restarts which produced solutions are indicated. The right graph shows a curve parameterized by noise level and is the same data that was generated from the exhaustive *Walksat*-SKC runs on the left.

A second curious class of problems are characterized by the graph on the right of figure 5. In this case once the invariant ratio is minimized, the noise must be decreased to maximize performance.

Figure 6 shows the result of running *Auto-Walksat* on a logistics problem. Other experiments (not shown) demonstrated that the logistics class of problems all demonstrate the same reverse-loop behavior. In figure 4 *Auto-Walksat* sets the noise level 10% higher than the value that minimized the invariant ratio, but in figure 6, this was the wrong choice. In figure 6's case, *Auto-Walksat* correctly minimized the invariant ratio at noise levels varying between 39% and 69%, but incorrectly increased the noise from that point. Instead a 30% decrease would have been appropriate to set the noise level at 9% to 39%.

It is not clear what differentiates the logistics class from the problems in the DIMACS benchmark set or how *Auto-Walksat* could decide whether to increase or decrease the noise level once the invariant ratio is minimized.

6 Conclusions

In this paper we investigated whether or not it was possible to exploit the invariant ratio presented by McAllester [5] to create a self-tuning variant of *Walksat*. Our algorithm *Auto-Walksat*, is able to successfully minimize the invariant ratio using a bracketed search supplemented with parabolic interpolation. The additional overhead of minimizing this ratio is very small, adding approximately one minute to the running time of the algorithm. Using a heuristic of adding ten percent noise to this value, *Auto-Walksat* then efficiently solves many problems which critically depend on a proper noise setting.

This investigation also revealed two areas of difficulty in applying this technique. The first is with problems that are pathological to *Walksat* and the second pertains to the proper heuristic technique following invariant ratio minimization. Specifically, it is not clear whether the noise level, once chosen should be increased or decreased to maximize performance.

Further investigation is warranted in determining what characteristics of satisfiability formulae should guide the choice of noise level setting following invariant ratio minimization.

References

[1] R. P. Brent. *Algorithms for Minimization without Derivatives.* Prentice–Hall, Englewood Cliffs, New Jersey, 1973.

[2] H. Kautz and B. Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In H. Shrobe and T. Senator, editors, *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pages 1194–1201, Menlo Park, California, 1996. AAAI Press.

[3] H. Kautz and B. Selman. Blackbox: a new approach to the application of theorem proving to problem solving, 1998.

[4] H. Levesque, D. Mitchell, and B. Selman. Hard and easy distributions of sat problems, 1992.

[5] D. A. McAllester, B. Selman, and H. A. Kautz. Evidence for invariants in local search. In *AAAI/IAAI*, pages 321–326, 1997.

[6] C. H. Nadya. Improved noise tolerance for walksat: Course project for AI 250a UCSD, http://citeseer.nj.nec.com/332054.html.

[7] A. J. Parkes and J. P. Walser. Tuning local search for satisfiability testing. In *AAAI/IAAI, Vol. 1*, pages 356–362, 1996.

[8] B. Selman, H. Kautz, and B. Cohen. Local search strategies for satisfiability testing. In *AAAI-92: Proceedings 10th National Conference on AI.* American Mathematical Society, 1995.

[9] B. Selman, H. A. Kautz, and B. Cohen. Noise strategies for improving local search. In *AAAI, Vol. 1*, pages 337–343, 1994.