An Adaptive Noise Mechanism for WalkSAT

Holger H. Hoos
University of British Columbia
Computer Science Department
2366 Main Mall
Vancouver, BC, V6T 1Z4, Canada
hoos@cs.ubc.ca

Abstract

Stochastic local search algorithms based on the WalkSAT architecture are among the best known methods for solving hard and large instances of the propositional satisfiability problem (SAT). The performance and behaviour of these algorithms critically depends on the setting of the noise parameter, which controls the greediness of the search process. The optimal setting for the noise parameter varies considerably between different types and sizes of problem instances; consequently, considerable manual tuning is typically required to obtain peak performance. In this paper, we characterise the impact of the noise setting on the behaviour of WalkSAT and introduce a simple adaptive noise mechanism for WalkSAT that does not require manual adjustment for different problem instances. We present experimental results indicating that by using this self-tuning noise mechanism, various WalkSAT variants (including WalkSAT/SKC and Novelty$^+$) achieve performance levels close to their peak performance for instance-specific, manually tuned noise settings.

Introduction and Background

The WalkSAT family of algorithms (Selman, Kautz, & Cohen 1994; McAllester, Selman, & Kautz 1997) comprises some of the most widely studied and best-performing stochastic local search (SLS) algorithms for the propositional satisfiability problem (SAT). WalkSAT algorithms are based on an iterative search process that in each step selects a currently unsatisfied clause of the given SAT instance at random (according to a uniform probability distribution), selects a variable appearing in that clause and flips it, *i.e.*, changes its truth value from true to false or vice versa. Different methods are used for the variable selection within unsatisfied clauses, giving rise to various WalkSAT algorithms (McAllester, Selman, & Kautz 1997; Hoos 1999; Hoos & Stützle 2000a). All of these use a parameter called the *noise parameter* to control the degree of greediness in the variable selection process, *i.e.*, the degree to which variables are likely to be selected that, when flipped, lead to a maximal decrease in the number of unsatisfied clauses.

The noise parameter, which for all WalkSAT algorithms except for WalkSAT/TABU represents a probability and hence takes values between zero and one, has a major impact on the performance of the respective algorithm, as measured by the probability of finding a solution, *i.e.*, a model of the given formula, within a fixed number of steps, or by the expected number of steps required for finding a solution. Not only is there a significant quantitative impact of the noise parameter setting on performance, but the qualitative behaviour of the algorithm can be different depending on the noise setting. In particular, it has been shown that for sufficiently high noise settings, the other important parameter common to all WalkSAT algorithms, the number of steps after which the search process is restarted from a randomly selected variable assignment (also called *cutoff parameter*) has little or no impact on the behaviour of the algorithm (Parkes & Walser 1996; Hoos & Stützle 1999). For low noise settings, however, finding an appropriate cutoff setting is typically crucial for obtaining good performance (Hoos & Stützle 2000a). Fortunately, for many of the most prominent and best-performing WalkSAT algorithms, including WalkSAT/SKC, WalkSAT/TABU, Novelty$^+$, and R-Novelty$^+$, peak performance is obtained for noise settings high enough that the cutoff parameter does not affect performance unless it is chosen too low, in which case, performance is degraded. This leaves the noise setting to be optimised in order to achieve maximal performance of these WalkSAT algorithms.[1]

Unfortunately, finding the optimal noise setting is typically a difficult task. Because optimal noise settings appear to differ considerably depending on the given problem instance, this task often requires experience and substantial experimentation with various noise values (Hoos & Stützle 2000a). We will see later that even relatively minor deviations from the optimal noise setting can lead to a substantial increase in the expected time for solving a given instance; and to make matters worse, the sensitivity of WalkSAT's performance *w.r.t.* the noise setting seems to increase with the size and hardness of the problem instance to be solved. This complicates the use of WalkSAT for solving SAT instances as well as their evaluation, and hence the development, of new WalkSAT algorithms.

One obvious approach for developing a self-tuning mechanism for the noise parameter in WalkSAT is to build on McAllester *et al.*'s "invariants" that relate optimal noise parameter settings to certain statistics of the number of unsatisfied clauses over a (partial) WalkSAT trajectory (McAllester,

[1]It may be noted that Novelty$^+$ and R-Novelty$^+$ have an additional secondary noise parameter, which, however, seems to have less impact on performance than the primary noise parameter. Furthermore, one uniform setting of this parameter seems to achieve excellent performance for a broad range of SAT instances and instance types (Hoos 1999; Hoos & Stützle 2000a).

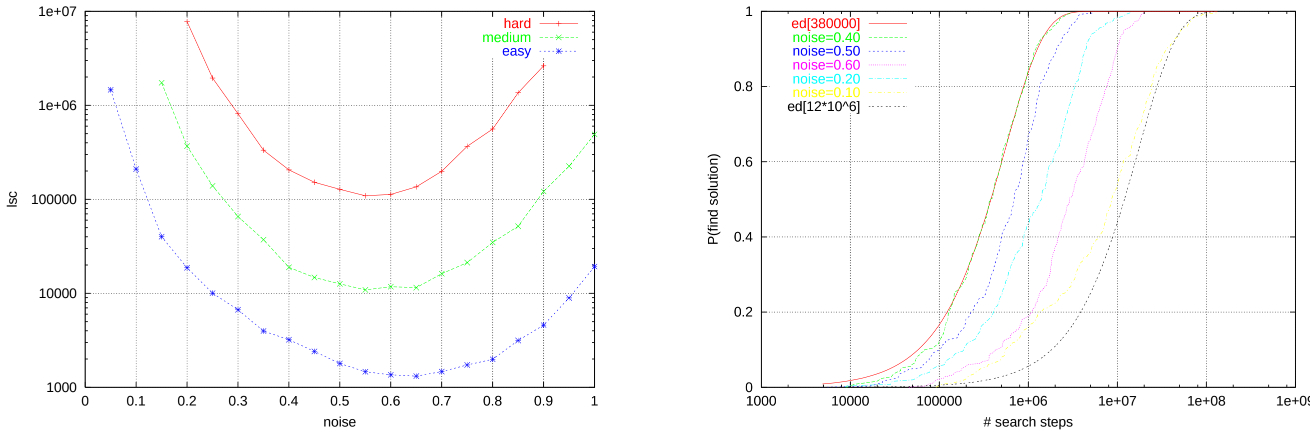

Figure 1: *Left:* Noise response for Novelty$^+$ on easy, medium, and hard instances from test-set `flat100-239-100`. *Right:* RTDs for for WalkSAT/SKC on SAT-encoded block world planning instance `bw_large.b` for approx. optimal, lower and higher noise settings.

Selman, & Kautz 1997). Recently, it has been demonstrated that these invariants can be used as the basis for automatically tuning the noise parameter in WalkSAT/SKC (Patterson & Kautz 2001). It should be noted, however, that these relationships are of an approximate nature and that thus far, they have only be established for WalkSAT algorithms.

The approach followed in this paper is based on a more general principle that can easily be generalised to SLS algorithms other than the WalkSAT architecture and to hard combinatorial problems different from SAT. It substantially differs from the method proposed in (Patterson & Kautz 2001), which optimises the noise setting for a given problem instance prior to the actual (unmodified) search process, during which the noise parameter setting is held fixed. The key idea behind our noise mechanism is to use high noise values only when they are needed to escape from stagnation situations in which the search procedure appears to make no further progress towards finding a solution. This idea is closely related to the motivation behind Reactive Tabu Search (Battiti & Tecchiolli 1994) and Iterated Local Search (Lourenço, Martin, & Stützle 2000), two high-performing SLS algorithms for combinatorial optimisation. Applied to WalkSAT algorithms such as Novelty$^+$, this approach not only achieves a remarkably robust and high performance, in some cases it also improves over the peak performance of the best previously known WalkSAT variant for the respective problem instance.

The Noise Response

We use the term *noise response* to refer to the functional dependency of the local search cost on the setting of the noise parameter. The noise response captures the characteristic impact of the noise setting on the performance of a given algorithm for a specific problem instance. *Local search cost* (abbreviated *lsc*) is defined as the expected time required by a given algorithm (for specific parameter settings) to solve a given problem instance. We estimate *lsc* by taking the average of an empirical run-time distribution (RTD). Since the variance of WalkSAT RTD is typically very high, stable estimates of *lsc* require empirical RTDs based on a large number of successful runs. Unless specifically stated otherwise, the *lsc* measurements reported in this paper are based on at least 250 successful runs. Furthermore, random restart within runs was generally disabled by setting WalkSAT's cutoff parameter effectively to infinity. As we will see later, this does not affect the peak performance of the algorithms studied here.

Measuring the noise response for more than 300 SAT instances (most of which were taken from the SATLIB Benchmark Collection), including SAT-encoded planning and graph colouring problems, we found that the noise response for WalkSAT/SKC, Novelty$^+$, and R-Novelty$^+$ always has the same characteristic, concave shape: There exists a unique optimal noise setting minimising *lsc*; for noise higher than this optimal value, *lsc* increases monotonically; likewise, *lsc* increases monontonically as noise is decreased below the optimum value (typical examples are shown in Figure 1). The response curve is asymmetric, with a steeper increase in *lsc* for lower-than-optimal than for higher-than-optimal noise values, and there is no evidence for discontinuities in any of its derivatives.

As a consequence of this shape of the noise response curve, there is a certain robustness *w.r.t.* minor variations in the noise setting around the optimal value. Furthermore, lower-than-optimal noise values tend to cause significantly more difficulty in solving a problem instance than higher-than-optimal noise values. (This is particularly the case for some of the best-performing WalkSAT variants, such as Novelty$^+$ and R-Novelty$^+$.)

It has been previously observed that for optimal and higher-than-optimal noise settings, WalkSAT and other SLS algorithms for SAT show exponential RTDs (Hoos & Stützle 1999). For lower-than-optimal noise settings, RTDs indicate stagnation behaviour reflected in an increase in the variation coefficient (mean/stddev) with decreasing noise (Hoos & Stützle 2000a). (Typical RTDs are shown in Figure 1.) Because of the effect of the initial search phase that is most pronounced for relatively easy problem instances (relative to their size) around the optimal noise value, the variation coefficient can also slightly increase as the noise is increased beyond its optimal value.

There has also been some evidence in the literature that for

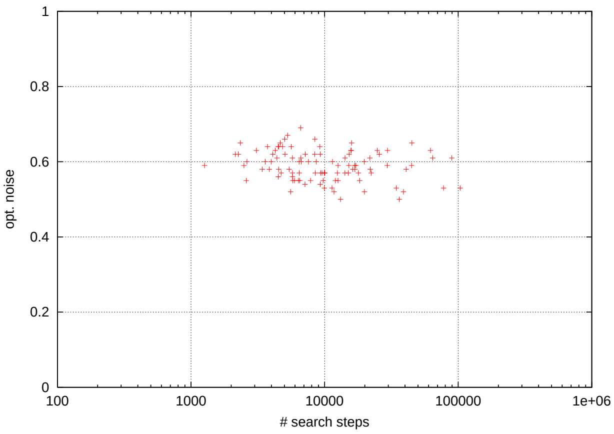

Figure 2: Approx. optimal noise values *vs.* *lsc* for Novelty$^+$ on test-set `flat100-239-100`.

sets of syntactically very similar problem instances, in particular for test-sets sampled from Uniform Random-3-SAT distributions (Cheeseman, Kanefsky, & Taylor 1991), the optimal noise values for WalkSAT/SKC are very similar (Hoos & Stützle 1999). This observation appears to hold for other sets of syntactically similar problem instances as well as for other WalkSAT variants. A typical example is shown in Figure 2; note that despite the syntactical similarity of the instances there are substantial differences in local search cost, which, however, are not significantly correlated with optimal noise settings. It may be noted that even at 250 tries per instance the *lsc* estimates, and hence our estimates for optimal noise settings, are often not very stable. For the test-set used in Figure 2, differences in search cost between the extreme optimal noise values obtained were smaller than a factor of 1.5.

However, optimal noise settings vary considerably with instance type and size (McAllester, Selman, & Kautz 1997; Hoos & Stützle 2000a). This is particularly noticable for the widely used SAT-encoded blocksworld planning instances (Kautz & Selman 1996; Hoos & Stützle 2000b), where the optimal noise values appear to decrease monotonically with problem size. For other instance types, including Uniform Random-3-SAT instances and SAT-encoded Flat Graph Colouring instances, the optimal noise value is apparently not affected by instance size. Overall, it appears that for those types of SAT instances where optimal noise changes with instance size, larger instances tend to have smaller optimal noise values (*cf.* Table 1).

Finally, there are significant differences in optimal noise levels between different WalkSAT variants. This is not surprising, considering the differences in how the noise parameter is used within these variants; but it is relevant in this context because it means that when comparing the performance of the variance for a given set of problem instances, the noise parameter setting needs to be optimised for each variant individually. This observation is particularly relevant in the context of recent findings that no single WalkSAT variant generally outperforms all others (Hoos & Stützle 2000a).

These observations suggest the following approach to manually tuning the noise parameter: For two initial guesses for the optimal noise value, empirical RTDs are measured and *lsc* values are calculated from these. These two initial noise values are guessed in such a way that they are likely to be slightly higher than the optimal noise value. Assuming that the *lsc* measurements are reasonably accurate, and exploiting the typical concave shape of the noise response curve, a simple iterative method can be used to narrow down the optimal noise value by measuring additional *lsc* values for appropriately chosen noise settings. Typically, RTDs for no more than four noise values need to be evaluated in order to obtain a noise setting for which the *lsc* value is no more than 20% above the minimum. The initial guesses are often based on obvious structural properties of the problem instances, such as the ratio of clauses to variables, or background knowledge about the origin of the problem instances, including the transformations used for encoding them into SAT.

The drawback of this method is that it requires solving the problem instance under consideration hundreds, maybe thousands of times. This only makes sense when tuning an algorithm for a whole class of problem instances in a scenario where a large number of similar problem instances have to be solved subsequently. According to our observation that for several widely studied classes of SAT instances the optimal noise settings seem to be very similar or identical over whole distributions of problem instances, this situation is not unrealistic (especially in the context of comparative studies of SLS algorithms over a wide range of problem instances).

WalkSAT with Dynamic Noise

Given the observations made in the previous section, it appears very desirable to have a mechanism that automatically adjusts the noise parameter in such a way that manual parameter tuning is no longer necessary for obtaining optimal performance.

There are at least four types of information that can potentially be used by such a mechanism:

(a) background knowledge provided by the algorithm designer; this knowledge might reflect extensive experience with the algorithm on various types of instances or theoretical insights into the algorithm's behaviour;

(b) syntactic information about the problem instance; for SAT instances, this may include the number of clauses and variables as well as information about clause lengths, *etc.*;

(c) information collected over the run of the algorithm so far; in particular, this includes information about the search space positions and objective function values encountered over the (incomplete) search trajectory;

(d) information collected by specific mechanisms (or agents) that perform certain types of "semantic" analyses on the given problem instances; this can include active measurements of properties of the underlying search space, such as autocorrelation lengths for random walks (Weinberger 1990) or density of local optima (Frank, Cheeseman, & Stutz 1997).

Obviously, a self-tuning noise mechanism can integrate various types of information. In the following, we study a technique that is based on information of type (a), (b), and (c). Ultimately, we believe that information of type (d) should also be integrated, leading to a more robust and even better performing algorithm. However, from a scientific perspective as well as from an engineering point of view, it seems preferable to start with rather simple self-tuning algorithms before studying complex combinations of techniques.

Our approach is based on a simple and fairly general idea: Based on the effect of the noise setting on the search process, as described previously, and consistent with earlier observations by McAllester *et al.* (1997), it appears that optimal noise settings are those that achieve a good balance between an algorithms ability to greedily find solutions by following local gradients, and its ability to escape from local minima and other regions of the search space that attract the greedy component of the algorithm, yet contain no solutions. From this point of view, the standard static noise mechanism that performs non-greedy (or not-so greedy) search steps required to escape from situations in which the search would otherwise stagnate with a constant probability, seems to be a rather crude and wasteful solution. Instead, it appears much more reasonable to use this escape mechanism only when it is really needed.

This leads to our adaptive noise approach, in which the probability for performing greedy steps (or noise setting) is dynamically adjusted based on search progress, as reflected in the time elapsed since the last improvement in the objective function has been achieved. At the beginning of the search process, we use greedy search exclusively (noise=0). This will typically lead to a series of rapid improvements in the objective function value, followed by stagnation (unless a solution to the given problem instance is found). In this situation, the noise value is increased. If this increase is not sufficient to escape from the stagnation situation, *i.e.*, if it does not lead to an improvement in objective function value within a certain number of steps, the noise value is further increased. Eventually, the noise value should be high enough that the search process overcomes the stagnation, at which point, the noise can be gradually decreased, until the next stagnation situation is detected or a solution to the given problem instance is found.

Our first implementation of the adaptive noise mechanism uses very simple techniques for the basic components of stagnation detection, noise increase, and noise decrease. As an indicator for search stagnation we use a predicate that is true iff no improvement in objective function value has been observed over the last $\theta \cdot m$ search steps, where m is the number of clauses of the given problem instance and $\theta = 1/6$. Every incremental increase in the noise value is realised as $wp := wp + (1 - wp) \cdot \phi$. the decrements are defined as $wp := wp - wp \cdot 2\phi$, where wp is the noise level and $\phi = 0.2$.

The asymmetry between increases and decreases in the noise setting is motivated by the fact that detecting search stagnation is computationally more expensive than detecting search progress and by the earlier observation that it is advantageous to approximate optimal noise levels from above rather than from below. After the noise setting has been increased or decreased, the current objective function value is stored and becomes the basis for measuring improvement, and hence for detecting search stagnation. As a consequence, between increases in noise level there is always a phase during which the trajectory is monitored for search progress without further increasing the noise. No such delay is enforced between successive decreases in noise level.

It may be noted that this adaptive noise mechanism uses two internal parameters, θ and ϕ, that control its behaviour. While it appears that this merely replaced the problem of tuning one parameter, wp, by the potentially more difficult problem of tuning these new parameters, the values of θ and ϕ used in this study were determined in preliminary experiments and then kept fixed throughout the rest of this study. In particular, the same values for θ and ϕ were used for all problem instances used in our performance evaluation. As we will see in the next section, various WalkSAT algorithms, when using the adaptive noise mechanism introduced here, achieve very impressive performance for the same fixed values of θ and ϕ, while the same algorithms, for the same fixed value of wp perform substantially worse. This indicates that, while our adaptive mechanism has some possible internal adjustments, these adjustments do not have to be tuned for each problem instance or instance type to achieve good performance.

Experimental Results and Discussion

The adaptive noise mechanism described in the previous section can be easily integrated into existing implementations of WalkSAT. In order to evaluate its performance against peak performance as obtained for manually tuned static noise, we conducted extensive computational experiments on widely used benchmark instances for SAT obtained from SATLIB (Hoos & Stützle 2000b). The benchmark set used for our evaluation comprises SAT-encoded blocksworld and logistics planning instances, two types of SAT-encoded graph colouring problems, critically constrained Uniform Random-3-SAT instances, and SAT-encoded all-interval-series problems. In addition, primarily to assess scaling behaviour, we generated a new test-set of 100 critically constrained, satisfiable Uniform Random-3-SAT instances with 400 variables and 1700 clauses each. The instances labelled `uf*-hard` are those instances from the respective critically constrained Uniform Random-3-SAT test-sets with the highest *lsc* for WalkSAT using manually tuned static noise.

As can be seen from Table 1, Novelty$^+$ with dynamic noise performs very well, considering the fact that it used no instance-specific parameter tuning, and keeping in mind that when using the standard static noise mechanism, especially for hard and large instances, even relatively small deviations from the optimal noise setting can easily lead to increases in *lsc* of more than an order of magnitude. It may be noted that the weakest performance is observed for the large DIMACS graph colouring instances, `g125_17` and `g125_18`. Additional experiments (not shown here) indicated that by using a different stagnation criterion, performance on these instances can be significantly improved; this stagnation criterion, however, does not perform as well on the other instances tested here. Similarly, we observed that for different parameter settings θ and ϕ of the dynamic noise mechanism, the performance on almost all instances can be further improved. These observations suggest that more sophisticated mechanisms for adjusting the noise should be able to achieve overall performance improvements and in some cases are likely to exceed the performance of the best known SLS algorithms for SAT.

It is worth noting that in three cases, dynamic noise achieves better performance than approx. optimal static noise. At first glance, this might appear surprising; however, it should be noted that the adaptive noise mechanism does not merely attempt to find the optimal static noise level, but is rather based on the idea of using noise only when it is

instance	nov+opt		nov+dyn		dyn / opt
	lsc	noise	*lsc*	noise	*lsc* ratio
bw_large.a	9,388	0.40	12,156	0.47 ± 0.07	1.29
bw_large.b	197,649	0.35	212,671	0.30 ± 0.05	1.08
bw_large.c	$7.57 \cdot 10^6$	0.20	$8.77 \cdot 10^6$	0.19 ± 0.02	1.16
log.c	123,984	0.40	141,580	0.34 ± 0.03	1.14
flat100-hard	139,355	0.60	111,772	0.44 ± 0.07	0.80
g125_18	8,634	0.45	32,498	0.55 ± 0.04	3.76
g125_17	$0.84 \cdot 10^6$	0.25	$1.41 \cdot 10^6$	0.26 ± 0.03	1.68
uf100-hard	38,473	0.55	41,733	0.46 ± 0.07	1.08
uf250-hard	$3.71 \cdot 10^6$	0.55	$2.92 \cdot 10^6$	0.37 ± 0.02	0.79
uf400-hard	$22.9 \cdot 10^6$	0.55	$22.8 \cdot 10^6$	0.32 ± 0.01	1.00
ais10	$1.96 \cdot 10^6$	0.40	$1.72 \cdot 10^6$	0.33 ± 0.04	0.88

Table 1: Novelty$^+$ with approx. optimal static noise *vs.* dynamic noise mechanism on individual benchmark instances. *lsc* estimates are based on at least 250 runs for all instances except for uf400-hard, for which only 100 runs have been conducted. For Novelty$^+$ with dynamic noise, the mean and standard deviation of the noise over all runs are reported.

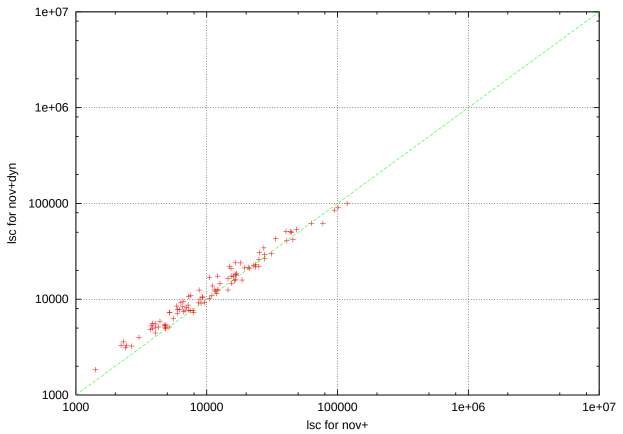

Figure 3: Correlation between *lsc* for Novelty$^+$ with optimal static noise *vs.* dynamic noise mechanism on test-set flat100-239-100.

actually needed. Nevertheless, as can be seen from comparing the approx. optimal static noise levels and the statistics over the noise levels used by the dynamic variants, there is a correlation between the noise levels used in both cases. An interesting exception can be observed for the hard Random-3-SAT instances, for which the adaptive noise mechanism uses noise levels that are significantly lower than the optimal static noise setting. Generally, the low variation in noise level for the dynamic mechanism indicates that the noise levels used within runs on an individual instance are very consistent.

Table 2 shows the relative performance obtained by Novelty$^+$ with dynamic *vs.* approx. optimal static noise across four of the test-sets of instances used in our evaluation. Interestingly, the dynamic noise variant achieves a significantly lower variation in *lsc* across all test-sets, as can be seen by comparing the respective variation coefficients (vc). Furthermore, as illustrated in Figure 3, there is a very strong correlation between the performance of both variants, with a small but significant tendency for dynamic noise to achieve lower *lsc* than static noise on hard instances. This is consistent with the intuition that the adaptive noise mechanism requires a certain amount of time before reaching good noise levels. (This "homing in" phenomenon can be observed from traces of the actual noise level over search trajectories of the algorithm, not shown here.)

As noted earlier, a significant advantage for conventional WalkSAT algorithms including WalkSAT/SKC, Novelty$^+$, R-Novelty$^+$ with static noise lies in the fact that they show memory-less behaviour for optimal noise levels. This makes their performance robust *w.r.t.* the cutoff parameter and provides the basis for achieving optimal speedup using a straight-forward multiple independent tries parallelisation. It turns out that the WalkSAT variants with dynamic noise also have this property. In all cases, the respective RTDs can be approximated well with exponential distributions, which is indicative of the same memory-less behaviour as observed for approx. optimal static noise.

So far, we have only compared run-times in terms of individual variable flips. But obviously, the time-complexity of these search steps also needs to be taken into account when assessing the performance of the new WalkSAT variants with dynamic noise. The time-complexity of search steps was measured on a PC with dual Pentium III 733MHz CPUs, 256MB CPU cache, and 1GB RAM running Redhat Linux Version 2.4.9-6smp. It was found that for Novelty$^+$, R-Novelty$^+$, and WalkSAT/SLK on a broad set of benchmark instances, the CPU-time per variable flip was typically about 5–10% higher for the dynamic noise variant compared to the respective versions with standard static noise. This confirms that even when using a straight-forward implementation, the dynamic noise mechanism causes only a minimal overhead *w.r.t.* the time-complexity of search steps. It may be noted that in some cases, such as for WalkSAT/SKC when running on bw_large.c, search steps were up to 20% faster for dynamic than for static noise. This is caused by the fact that the time-complexity of WalkSAT search steps depends on the number of unsatisfied clauses, which in these cases drops more rapidly in the initial search phase when using the adaptive noise mechanism.

Due to space constraints, in this paper we report performance results for Novelty$^+$ with dynamic noise only. We obtained, however, empirical evidence indicating that the same adaptive noise mechanism appears to work well for WalkSAT/SKC and R-Novelty$^+$. Using the same values for θ and ϕ as in the present study, the performance (*lsc*) achieved by

test-set	*lsc* for nov+opt			*lsc* for nov+dyn		
	mean	cv	median	mean	cv	median
`flat100-239`	17,205	1.18	10,497	21,102	1.07	13,231
`flat200-479`	495,018	1.70	241,981	573,176	1.47	317,787
`uf100-430`*	2512.8	2.98	898.5	2550.9	2.16	1121.8
`uf250-1065`	53,938	5.26	8,755	64,542	4.72	13,015

Table 2: Novelty$^+$ with dynamic *vs.* approx. optimal static noise on various sets of benchmark instances. ($*$) The data for test-set `uf100-430` was computed for 100 randomly selected instances from that set. 'cv' denotes the coefficient of variation, *i.e.*, stddev/mean, of the distribution of *lsc* across the respective test-sets.

R-Novelty$^+$ with dynamic noise is within a factor of 1.5 of the performance obtained using approx. optimal static noise settings for 8 of the 11 instances listed in Table 1; in four of these cases, using dynamic noise results in substantially better performane than using approx. optimal static noise. Even better performance can be achieved for slightly different θ and ϕ settings. Similar results were obtained for WalkSAT/SKC; full reports on these experiments will be included in an extended version of this paper (currently available as a technical report).

Conclusions

We have characterised the noise response of WalkSAT algorithms and introduced an adaptive noise mechanism that achieve very good performance on a broad range of widely used benchmark problems when compared to the peak performance of traditional variants of WalkSAT with static noise.

In principle, this adaptive noise mechanism is easily applicable to a much wider range of stochastic local search algorithms for SAT and other combinatorial problems. This is particularly attractive for other high-performance algorithms, such as WalkSAT/TABU (McAllester, Selman, & Kautz 1997) and GSAT with tabu lists (Selman, Kautz, & Cohen 1994), DLM (Wu & Wah 1999), or ESG (Schuurmans, Southy, & Holte 2001), which all have parameters that are in many ways analogous to the noise parameter in the WalkSAT variants studied here. While the implementation of our adaptive noise strategy for these algorithms is rather straightforward, its effectivity in terms of achieving good and robust performance remains to be shown.

Another avenue for further investigation is the development and analysis of different and improved criteria for search stagnation which can be used within our generic adaptive mechanism. We strongly believe that the simple stagnation criteria studied here can be substantially improved, *e.g.*, by including measures such as search mobility (Schuurmans & Southy 2000) or the ones used in McAllester *et al.*'s invariants (McAllester, Selman, & Kautz 1997). Further improvements of the noise adaption mechanism and of adaptive SLS algorithms in general could possibly be achieved by integrating simple search space analysis techniques into the search control. Another promising avenue for further investigation is to study the use of machine learning techniques for identifying features that are effective for detecting search stagnation or for predicting optimal noise values.

Finally, it should be noted that the deeper reasons underlying the characteristic shape of the noise response curve for WalkSAT algorithms and the shape of the corresponding runtime distributions are unknown. Since these are intimitely connected to crucial aspects of SLS behaviour, further investigation in this direction could lead to improvements in our understanding of current SLS algorithms and in the design of future methods.

References

Battiti, R., and Tecchiolli, G. 1994. The reactive tabu search. *ORSA J. on Computing* 60(2):126–140.

Cheeseman, P.; Kanefsky, B.; and Taylor, W. M. 1991. Where the Really Hard Problems Are. In *Proc. IJCAI-91*, 331–337.

Frank, J.; Cheeseman, P.; and Stutz, J. 1997. When Gravity Fails: Local Search Topology. *(Electronic) Journal of Artificial Intelligence Research* 7:249–281.

Hoos, H., and Stützle, T. 1999. Towards a Characterisation of the Behaviour of Stochastic Local Search Algorithms for SAT. *Artificial Intelligence* 112:213–232.

Hoos, H., and Stützle, T. 2000a. Local search algorithms for SAT: An empirical evaluation. *J. Automated Reasoning* 24:421–481.

Hoos, H., and Stützle, T. 2000b. SATLIB: An Online Resource for Research on SAT. In I.P. Gent, H. M., and Walsh, T., eds., *SAT 2000*, 283–292. IOS Press.

Hoos, H. 1999. On the run-time behaviour of stochastic local search algorithms for SAT. In *Proc. AAAI-99*, 661–666.

Kautz, H., and Selman, B. 1996. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proc. AAAI-96*, 1194–1201.

Lourenço, H.; Martin, O.; and Stützle, T. 2000. Iterated Local Search. Technical Report AIDA–00–06, Technische Universität Darmstadt. (To appear in F. Glover and G. Kochenberger, editors, Handbook of Metaheuristics, Kluwer, 2002).

McAllester, D.; Selman, B.; and Kautz, H. 1997. Evidence for invariants in local search. In *Proc. IJCAI-97*, 321–326.

Parkes, A. J., and Walser, J. P. 1996. Tuning Local Search for Satisfiability Testing. In *Proc. AAAI-96*, 356–362.

Patterson, D. J., and Kautz, H. 2001. A Self-Tuning Implementation of Walksat. *Electronic Notes on Discrete Mathematics* 9. (Presented at the LICS 2001 Workshop on Theory and Applications of Satisfiability Testing).

Schuurmans, D., and Southy, F. 2000. Local search characteristics of incomplete SAT procedures. In *Proc. AAAI-2000*, 297–302.

Schuurmans, D.; Southy, F.; and Holte, R. 2001. The exponentiated subgradient algorithm for heuristic boolean programming. In *Proc. IJCAI-01*, 334–341.

Selman, B.; Kautz, H. A.; and Cohen, B. 1994. Noise strategies for improving local search. In *Proc. AAAI-94*, 337–343.

Weinberger, E. 1990. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics* 63:325–336.

Wu, Z., and Wah, B. 1999. Trap Escaping Strategies in Discrete Lagrangian Methods for Solving Hard Satisfiability and Maximum Satisfiability Problems. In *Proc. AAAI-99*, 673–678.