A Characterization of GSAT's Performance on a Class of Hard Structured Problems

Holger Hoos and Thomas Stützle

Intellektik!

Forschungsbericht AIDA–96–20

Darmstadt

Fachgebiet Intellektik, Fachbereich Informatik
Technische Hochschule Darmstadt
Alexanderstraße 10
D–64283 Darmstadt
Germany

A Characterization of GSAT's Performance on a Class of Hard Structured Problems

Holger H. Hoos* Thomas Stützle*

Abstract

Randomized algorithms for solving hard satisfiability problems have been successfully applied to many application areas, but are still poorly understood. In this work, we show that, applied to SAT-encoded Hamilton circuit problems, GSAT with Random Walk (GWSAT) displays a very regular behaviour. In particular, the dependency of the success-rate on the maxSteps parameter for optimal walk probability can be charactarized by a simple algebraic function. We found an almost identical situation for GHC, a GSAT-variant adapted to the structure of the given problems class. Based on this observation, runtime estimates for given test-sets and parameter settings can be obtained. Furtheron this result shows that optimal speedup can be achieved for GWSAT (and GHC) when performing tries in parallel. The observed behaviour gives also rise to a novel interpretation of the search performed by GWSAT (and GHC) as an iterated blind probing in a much smaller, "virtual" searchspace the size of which is a measure for the search algorithms performance. We have evidence, that this concept can be also applied to other problem classes and local search algorithms.

1 Introduction

Developing and studying algorithms for general logical problems like the satisfiability problem in propositional logic (SAT) is strongly motivated by the fact that many problems from different domains can be formulated as logical problems. Much work in the field of AI is based on the assumption that these encoded problems can be adequately solved by general logical methods. This general approach seems particularly interesting for computationally hard (i.e. $\mathcal{NP}$-hard) problems for which no specialized, efficient algorithms are available.

GSAT [13], an incomplete SAT algorithm based on local hill-climbing, and its variants have been successfully applied to a number of problems from different application areas [12, 9] but understanding of its behaviour is still very

*Affiliation: FG Intellektik, FB Informatik, TU Darmstadt; Alexanderstr. 10, D-62483 Darmstadt, Germany; email: {hoos,stuetzle}@informatik.tu-darmstadt.de

limited. In this paper we investigate the performance of some GSAT-variants on SAT-encoded instances of the Hamilton circuit problem in directed graphs (HCP). These problems form a class of highly structured SAT instances which has turned out to be very difficult for GSAT with Random Walk (GWSAT) [6]. Our aim here is not to devise a competitive method for solving HCP, but to gain insight into the behaviour of local search algorithms like GWSAT on hard, structured problems, which is not only needed to decide how these algorithms can be most effectively applied, but might also help to significantly improve them.

The results of a detailed empirical analysis show that GWSAT's performance on this problem class is surprisingly regular and predictable. Actually, we can give a functional characterization of the relation between the maxSteps parameter (which essentially gives an upper bound for the run-time of the algorithm) and success-rate. We show that GWSAT is an approximately complete method for these problems and point out some implications of our results on parameter-settings and parallelization. We further develop a new interpretation for GWSAT's search which can be also applied to other search algorithms displaying a similar behaviour. We are convinced that from these results theoretical insights in the underlying search-space structure (which is induced by the encoding method) can be obtained, which in turn might be relevant for the improvements of SAT algorithms as well as encoding methods.

The remainder of this paper is structured in the following way: In the next section we will shortly review a method for transforming HCP instances into SAT problems [7] and describe the GSAT algorithm [13]. Sect.3 introduces GHC, a modification of GSAT which exploits structural aspects of the transformed problems. In Sect.4 we report the results of a detailed empirical analysis of GSAT's and GHC's performance on this problem class. These results and their implications are discussed in Sect.5; finally, in Sect.6 we sum up the main results of this work and point out directions for further research.

2 Solving Hamilton Circuit Problems with GSAT

In this section we first describe, how HCP-instances can be efficiently transformed into SAT-instances. Then we present GSAT, a powerful, model-generating algorithm for the SAT-problem.

2.1 Transforming HCP into SAT

For a given graph G (directed or undirected), the Hamilton circuit problem is the task of finding a cyclic tour (Hamilton circuit) in G visiting every vertex exactly once. This problem is $\mathcal{NP}$-complete (like SAT) and therefore, unless $\mathcal{P} = \mathcal{NP}$, not solvable by polynomial-time algorithms. The method we use for the SAT-encoding of HCP instances is due to Iwama and Miyazaki [7]; it is based on the general idea of transforming HCP instances into CNF formulae by focusing on tours of length n in the graph G, where n is the number of vertices of G. This is done by assigning an integer value to each vertex, which defines its position in the tour. The function which assigns such a *vertex position* to each vertex of the graph is called a *vertex assignment.* A vertex assignment encodes a Hamilton circuit of the given graph, iff:

- no vertex is visited more than once,
- nodes with successive positions in this tour are always connected by an edge,
- there is an edge connecting the first vertex of the tour to the last one.

Using this characterization, any given instance of HCP can be easily transformed into a constraint satisfaction problem (CSP) on integers (i.e. vertex positions) which suggests a SAT-transformation in a straightforward way: namely by coding every vertex position by a number of propositional variables using the binary representation of integers. This allows every constraint on vertex positions to be translated into a CNF-clause such that the integer CSP is finally transformed into a SAT problem.[1]

Based on this idea, it is easy to devise an encoding algorithm with polynomial time complexity (wrt. n) which for any given directed graph G with n vertices generates a CNF-formula α with $(n-1)\lceil\log_2(n-1)\rceil$ propositional variables that is satisfiable iff G contains a Hamilton circuit [7, 6]. This is done in such a way that models of α directly correspond to the solutions of the HCP.

Now consider a more straight-forward approach where there is one propositional variable for each edge of the graph and a Hamilton circuit is represented by an assignment which makes exactly those variables true which correspond to the edges of that circuit. Clearly, such an encoding would require up to n^2 variables for highly connected graphs and thus, considering the complexity of the problem, cause a tremendous blow-up of the problem-size. Because the Iwama-Miyazaki-Transformation on the other hand avoids this while also still displaying a time complexity which is small wrt. the time complexity of the problem itself, we call it an "efficient transformation" [7].

A simple argument shows that one cannot expect to find SAT-transformations of HCP using significantly fewer propositional variables (see [7] for details). The CNF-formulae which are obtained from HCP-instances using this algorithm will be referred to as *HC-formulae* in the sequel.

2.2 GSAT – a Fast, Incomplete SAT-Algorithm

GSAT is a heuristic SAT-algorithm proposed by Selman, Levesque and Mitchell [13] which is based essentially on greedy local search in the space of all variable assignments of the input CNF-formula. Starting with a randomly chosen assignment A, in every step of the procedure the value of one propositional variable is

[1] Note that by the encoding algorithm presented in [7, 6] symmetric solutions are eliminated.

flipped, i.e. changed from true to false or vice versa. The selection of the variable to be flipped is based on the difference in the number of currently satisfied clauses the flip would cause (which is called the variable's *score*). In every step, a variable with maximal score is flipped; if there are several variables with maximal score, one of these is randomly selected. Thus, GSAT basically performs a kind of local hill-climbing on the number of satisfied clauses, which can be viewed as the global optimization of an objective function. When all clauses are satisfied, i.e. a model of the input formula is found, GSAT terminates. If after a given number maxSteps of flips no solution is found, the algorithm restarts its local search by randomly choosing a new variable assignment. If after a given number maxTries of these tries still no model is found, GSAT terminates unsuccessfully. This can be formalized in the following way:

```
procedure GSAT(α)
  for i := 1 to maxTries do
    A := randomly chosen assignment of the variables
           appearing in α;
    for j := 1 to maxSteps do
      if A satisfies α then return(A);
      else
        x := randomly chosen variable of α with
               max. score under current assignment A;
        flip value of x in A;
        end if;
      end for;
    end for;
  return (''no model found!'');
  end GSAT;
```

Note, that due to the random choices for the initial assignment and between equally good flips, GSAT is a non-deterministic algorithm. Another important property of GSAT is its incompleteness: However the parameters maxTries and maxSteps are chosen, if the input formula is satisfiable, there is no guaranty that GSAT finds a model. So in contrast to complete SAT-algorithms like the Davis-Putnam procedure (DP), GSAT can never determine the *unsatisfiability* of a given formula. However, it has been shown, that variants of the GSAT algorithm can outperform complete systematic search methods like DP on large classes of SAT-problems [12, 9].

Another difficulty with local search methods like GSAT is that they can "get stuck" in local optima of the objective function which do not represent solutions. A simple technique to cope with that is the restart mechanism which is already realized in basic GSAT. An important extension of the basic algorithm is the "Random Walk" strategy [12]: Here, in each step with a fixed probability walkProb a randomly chosen variable appearing in a currently unsatisfied clause is flipped instead of follwing the usual greedy strategy. Therefore, by each walk-step at least one formerly unsatisfied clause becomes satisfied. In contrast to ordinary hill-climb-steps, a walk-step may cause a decrease in the total number of satisfied clauses even if improving flips would have been possible. In consequence, by using Random Walk the local search becomes somewhat less directed which increases the chances of escaping from local optima. Experimental analysis indicates, that for most problem classes GSAT with Random Walk (GWSAT) shows a significantly improved performance when compared to basic GSAT [12].

3 GHC - Exploiting Structure of HC-Formulae

In this section we present GHC, a problem specifc modification of GSAT which exploits the structure of HC-formulae. Due to space limitations, we can present only the general idea here, a detailed description will be given elsewhere.

We recall that the SAT transformation used for HC instances encodes ever vertex position by a set of $\log n$ variables, where n is the number of vertices in the given graph. A single GSAT step therefore allows only direct transitions between vertex positions with Hamming distance one wrt. their binary representations. GHC is based on the idea of extending GSAT, such that arbitrary changes in one vertex position can be done in one step. This involves simultaneously changing the value of a set of variables (so-called "multiflips"). However, GHC allows only multiflips within the set of variables encoding the same vertex position. Since the results of the single flips within such a multiflip are not independent from each other, we have to consider n^2 assignments in each GHC step ($2^{\log n}$ for every vertex) as opposed to $n \log n$ for a GSAT step.

Because of complete variable dependency, another important structural aspect of HC-formulae which essentially states that flipping any single variable might affect the score of all others, this is not as bad as it seems [6]. As we use the score-by-need method (cf. [5]) for our GSAT implementation, we have in both cases to update the score of up to all $n \log n$ variables after each step. Since these updates are much more time consuming steps than the selection of the next assignment (even in the extended neighbourhood structure of GHC), the time complexity of GHC steps is identical to that of GSAT steps.

We also use a generalized Random Walk strategy in GHC, where a walk step is realized as a GHC multiflip such that at least one formerly unsatisfied clause becomes satisfied (this is strictly analogous to Random Walk for GSAT). In our implementation of GHC we further never consider variable assignments which encode invalid vertex positions. In general, this does not change the behaviour of GHC in a significant way but allows for a more efficient implementation.Finally, instead of actually transforming the given HCP instance into SAT, we apply GHC directly to the graph representation and only simulate the desired behaviour on the SAT transformed problem.[2] As a consequence, our implementation of GHC is about 10 times faster than GSAT and needs far less memory. However, conceptually GHC can still be regarded as a GSAT variant; we will therefore use GSAT or GWSAT in a broader sense to refer to both the GWSAT and GHC algorithm in certain contexts.

[2]This generic method, which we call *implicit clause encoding (ICE)* can also be easily applied to other problem classes.

4 Characterizing GSAT's Performance on HC-Formulae

Having described how GSAT-variants can be used to generate solutions for the HCP in general, we concentrate now on the actual performance of these algorithms on the given problem class. As known from [6], HC-formulae for directed graphs are very difficult for GSAT and several extensions to the basic algorithm (like Clause Weighting [11], or RBR extension [8] as a preprocessing step) which significantly improve its performance on other classes of hard structured problems [11, 2] but fail to be effective here. Actually, when compared to Random-3-SAT problems from the phase transition region with the same number of variables, the subclasses of HC-formulae considered here tend to be significantly harder to solve for these GSAT variants. So despite the fact that there are algorithms to more efficiently solve these problems, they are very hard for GSAT and some of its most successful variants.

However, we do not claim that the encoding method that has been used here is very favourable for the application of GSAT. The important point we wish to make here is rather that the HC-formulae form a class of SAT problems which is derived from a "real world" problem, the instances of which are highly structured and which is particularly hard for some otherwise very successful GSAT variants [6]. So our main objective here is neither to find a better SAT encoding for HCP, nor to devise a better algorithm for solving this subclass of SAT but rather to gain insight into GSAT's behaviour on hard structured problems in general.

For our empirical study we applied GSAT and GHC with and without Random Walk to several sets of SAT-transformed HCP instances. Our experimental approach is based on a method which is frequently applied in the analysis of SAT-algorithms (see [4, 1]). Since no suitable benchmark problems were available, we generated several sets of solvable instances.[3] In order to achieve a certain homogeneity of the test-sets, we restricted the experimental analysis to problem classes $\mathrm{HCF}_2(n)$ consisting of the solvable HCP instances with n vertices and out-degree 2. Test-sets were randomly generated for various problem sizes n between 10 and 30 by constructing first a Hamilton circuit of the given size and afterwards adding randomly selected edges, until every vertex had exactly the specified out-degree.[4] To abstract from random differences between the instances, each test-set consists of 100 independently generated instances.

As performance criteria we concentrate on the success-rate $sr = 100 \cdot st/tt$ (where st denotes the number of successful tries and tt the total number of tries for a specific experiment) here. For incomplete methods like GSAT and GHC this is an important performance criterion as unsuccessful tries are very

[3] Because both GSAT and GHC are incomplete algorithms, it doesn't make much sense to apply it to potentially unsolvable problem instances for evaluation purposes.

[4] We are aware of the fact that these problems might be intrinsically easier than "unforced problems" (which are not guaranteed by construction to have a solution). However, as stated above, they have been found to be very hard for algorithms like GSAT.

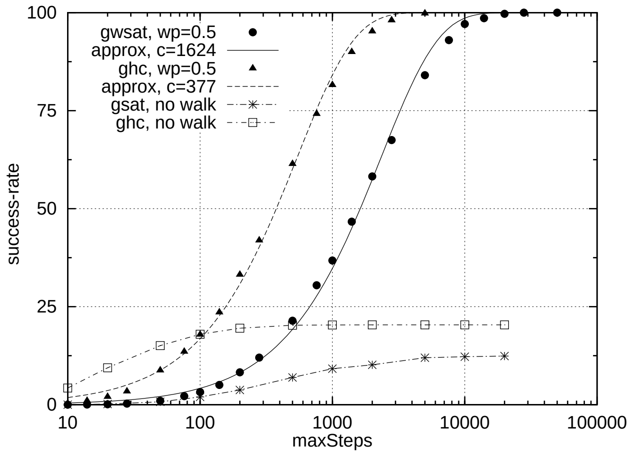

Figure 1: success-rate vs. maxSteps for walkProb of 0 resp. 0.5 on $\mathrm{HCF}_2(10)$; maxTries $=100$

expensive. Also, the expected number of tries that is needed to solve a problem with a fixed probability can be easily predicted from the success-rate.

GWSAT's (and GHC's) behaviour is controlled by 3 parameters: maxTries, maxSteps and walkProb. Since the individual tries (i.e. executions of the outer loop) are mutually independent, the influence of this parameter on both algorithms' performance can be easily predicted and needs not to be studied empirically. As for walkProb, it is known that there is an optimal setting of walkProb=0.5 for the given problem class [6].

Here, we concentrate on the relation between maxSteps and the success-rate. The results of our experiments for $\mathrm{HCF}_2(10)$ are given in Fig.1: Clearly, GHC shows a significantly improved performance wrt. GWSAT. Furtheron, qualitative different performance characteristics for walkProb=0 (which is GSAT resp. GHC without Random Walk) and walkProb=0.5 (the optimal walkProb-value) can be observed: Whereas for optimal walkProb the success-rate converges towards 100% for growing maxSteps, without walk it rises never beyond approximately 12% for GSAT and 20% for GHC. Since the structure of the formulae is not dependent on problem size, this qualitative difference should hold for arbitrary problem sizes (as our experiments confirm).

So we find, despite both GWSAT and GHC are generally incomplete algorithms, that by increasing maxSteps (i.e. the upper bound on solution costs) they can be used to solve each solvable HCF_2 instance with arbitrarily high probability by choosing maxSteps accordingly.[5] This property, which we call *approximate completeness*, critically depends on the Random Walk strategy, it does not hold for basic GSAT or GHC without Random Walk. Of course, sim-

[5]With HCF_2 we denote the union of all $\mathrm{HCF}_2(n)$.

ilar behaviour can be generally observed for all GSAT variants when increasing the number of tries. However, at this point it is not clear that this would be of much use because generally the number of tries needed to obtain reasonable success-rates would be prohibitively high. However, we will see shortly that this is not the case for the problem class considered here.

Fitting the curves for walkProb=0.5 using the Marquart-Levenberg algorithm reveals that both can be well approximated by the function $f_{sr}/100 = 1-2^{-ms/c}$ with $c = 1624$ for GSAT and $c = 377$ for GHC (see Fig.1). Furthermore, it turns out that for different problem sizes, the relation between maxSteps and success-rate for optimal walkProb can be characterized by the same function only using different values for c. Therefore we call this relation (abstracting from c) the *sr/ms-characteristic* of GWSAT resp. GHC on HCF_2. Summing up, we get the following result:

On HCF_2 problems, for optimal settings of walkProb *we find an identical relation between* maxSteps *and success-rate (sr/ms-characteristic) for both GSAT and GHC. This relation is qualitatively independent of the problem size and can be approximated by a simple function, $f_{sr}(ms) = 100 \cdot (1-2^{-ms/c})$.*

Using this result, success-rates for arbitrary maxSteps settings can be predicted when c is known; also for any given success-rate the maxSteps value that is needed to obtain it can be derived. Note, that for maxSteps=c we find $f_{sr}(c) = 50\%$, which means that for this parameter setting we expect 50% of the solvable instances to be actually solved by GWSAT resp. GHC. Furtheron, c can be measured by only one experiment (given that maxSteps is not too small). Relating the c values for different problem sizes one finds an approximately exponential dependence on the problem size as could be expected from the complexity of HCF_2. This approximation finally allows to predict success-rates for arbitrary problem sizes and maxSteps settings as well as to derive tighter upper bounds on the run-time as were previously available. The predictions for success-rates could be empirically confirmed which supports the assumption that the underlying approximations are indeed correct.

5 Discussion

Up to this point, we have shown that, for optimal walk probability, GWSAT's performance on HCF_2 can be characterized by a simple algebraic function and how this result can be used to estimate run-time and success-rate for experimental setups. However, further examination of the *sr/ms*-characteristic leads to some more interesting implications.

5.1 Optimal Parallelization of GWSAT

Looking at the approximation $f_{sr}/100 = 1-2^{-ms/c}$ we find that performing one try with maxSteps $= 2k$ yields exactly the same success-rate as two (independend) tries with maxSteps $= k$. As our experiments have shown, after an initial phase (which can be interpreted as the hill-climb-phase of [3]) the measured sr/ms-characteristic is almost identical to our approximation. We therefore conclude that after this initial phase the starting point for local search has almost no influence on the success-rate. From this observation we immidiately get:

For optimal walkProb*, GWSAT can be parallelized with optimal speedup by simultaneously running independent tries and setting* maxSteps *to the value from which on the sr/ms-characteristic fits closely the approximation.*

Differing from [3] we conclude, that at least for this problem class the plateau-phase of the local search should be cut off since here, no real progress is made anymore.

Therefore, e.g. on $\mathrm{HCF}_2(10)$ for GWSAT a setting of maxSteps $= 200$ should be used; to obtain a success-rate of approximately 80%, with maxTries $= 1$ a setting of maxSteps $= 3800$ would be required, whereas by performing 19 tries with maxSteps $= 200$ in parallel yields exactly the same success-rate while achieving a speedup of 19. The same result can be obtained by using GHC with walkProb $= 0.5$, maxTries $= 19$, and maxSteps $= 50$. Remarkably, for short tries GHC without Random Walk performs even better; to get 80% success-rate it suffices to select maxTries $= 19$ and maxSteps $= 20$. Thus, by exploiting problem structure and maximally parallelizing the algorithm based on our results regarding the sr/ms-characteristic we can achieve a speedup of approximately 190 in the given example.

5.2 Searchspace Reduction

We will now show that, given our approximation f_{sr} of the sr/ms-characteristic, the local search performed by GWSAT can be interpreted in a new way, namely as a series of independent, blind guesses in a much smaller searchspace.

As shown above, the success-rate obtained by GWSAT with maxSteps $= ms$ can be approximated by $f_{sr} = 100 \cdot (1 - 2^{-ms/c})$. But exactly the same kind of behaviour would be observed for an algorithm A which in each step blindly selects a randomly chosen variable assignment, terminating successfully, when stumbling upon a satifying assignment and unsuccessfully when after ms steps no solution has been found.[6] Therefore, we now interpret GWSAT's observed behaviour on a given HCF_2 instance with n propositional variables (in the searchspace S consisting of all variable assignments) as the behaviour of algorithm A in another, "virtual" searchspace S'.

Since most HCF_2 instances have only one satisfying assignment, it is reasonable to assume that there is also only one solution in the virtual searchspace S'. With s' denoting the size of S', algorithm A would therefore in each single step hit a solution with probability $1/s'$, i.e. $f_1/100 = 1 - 2^{-1/c} = 1/s'$. But in the above approximation for f_{sr} the constant c was empirically determined so that we can finally evaluate the size s' of the virtual searchspace S'.

[6] This corresponds to GSAT with maxSteps $=0$ and maxTries $=ms$.

Note, that this interpretation is applicable to every local search algorithm displaying a sr/ms-characteristic that can be approximated by a function of type $f_{sr} = 100 \cdot (1 - 2^{-ms/c})$, such as GWSAT or GHC. Given the same problem class, the size s' of the virtual searchspace wrt. such an algorithm B gives an indication for the effectiveness of its search method: the smaller s', the more effective is B. This notion is captured by the definition of the *searchspace reduction factor (srf)* r of an algorithm B as $r := s/s'$ where s denotes the size of the original searchspace S and s' the size of the virtual searchspace for B as introduced above.

From our results for GWSAT and GHC on $HCF_2(10)$ we can thus derive virtual searchspace sizes of $s' \approx 2343$ for GWSAT and $s' \approx 544.4$ for GHC. The corresponding original searchspace sizes are $s = 2^{36}$ and $s = 9^9$ respectively.[7] From this we get *searchspace reduction factors* of $r \approx 2.93 \cdot 10^7$ for GWSAT and $r \approx 7.11 \cdot 10^5$ for GHC. Of course this doesn't mean that GWSAT is more effective here, because for GHC the original searchspace S is much smaller. However, comparing different algorithms wrt. their *srf* does make sense if their original searchspaces are of the same size.

5.3 Generalizing the Results

Of course all this is of very limited interest, as long as it holds only for GWSAT on the very specific problem class HCF_2. Thus perhaps the most important question is, whether our results generalize to other problem classes and algorithms.

Further experimentation, which cannot be presented here due to the limited space available, shows clearly that for certain "homogeneous" sub-classes of unforced Random-3-SAT Formulae, this is actually the case. For these, we have found that GWSAT displays exactly the same type of sr/ms-characteristic for optimal walk-probability. However, this does not hold for arbitrary subsets of Random-3-SAT. Apparently a certain homogeneity wrt. the difficulty of the problem instances seems to be essential for the behaviour we observed. For the HCF_2 formulae this is probably enforced by the encoding mechanism and the uniform out-degree of the graph; whereas the difficulty of Random-3-SAT problem varies greatly even for fixed clauses/variable ratio. Certainly, the homogeneity which is required to reproduce our results for other problem classes will be reflected by the structural properties of the underlying searchspaces. Yet, this is one topic that we leave for further work.

We have also studied different SAT encodings for HCP; applying the empirical methodology presented in this paper shows that while there are encodings on which GWSAT performs significantly better than on the one discussed here, our observations and results still hold for all these encodings.

We are currently also investigating the behaviour of heuristics for Constraint Satisfaction Problems (CSPs). More specifically, we are considering randomly

[7] Note, that GHC allows only valid vertex positions and both methods fix the position of one vertex to eliminate symmetrical solutions.

generated binary CSPs and hard graph coloring problems. For the solution of these problems we are applying variants of the *Min-Conflicts Heuristic (MCH)* [10] enhanced by Random Walk and Tabu Search strategies. Again, for "homogeneous" classes of instances we observe the same qualitative *sr/ms*-characteristic as for GWSAT and GHC on HCF_2.

Summing up, we have strong indication that our results are neither specific for the problem presented here, nor for the encoding or the algorithm which was used. As apparently the concept of measuring searchspace reduction (as detailed above) can be applied to a variety of algorithms, problems and encodings, we believe this to be a fairly general approach for characterizing the behaviour of search algorithms.

6 Conclusion and Further Work

In this work, we have shown that, applied to SAT encoded Hamilton circuit problems, GSAT with Random Walk (GWSAT) shows a very regular behaviour. Using empirical methods, the dependency of the success-rate on the maxSteps parameter (*sr/ms-characteristic*) for optimal walk probability could be characterized by a simple algebraic function. We found an almost identical situation for GHC, a GSAT-variant adapted to the structure of HC-formulae. We argued that based on this observation estimates for the runtime of the algorithm on a given test-set of problems can be made. Furtheron both algorithms are *approximately complete* when the walkProb parameter is set to 0.5; however, GHC needs far lesser steps than GSAT to obtain similar success-rates. Moreover, it turns out that the observed *sr/ms*-characteristic allows for achieving *optimal speedup* of GWSAT (and GHC) when doing a number of tries in parallel.

Finally, we pointed out that the observed behaviour gives rise to a novel interpretation of the search performed by GWSAT (and GHC) as an iterated blind probing in a much smaller, *"virtual" searchspace.* Based on this interpretation we introduced the *searchspace reduction factor* as a measure for the efficency of search algorithms displaying the characteristic kind of behaviour which we observed for GWSAT here. We finally gave some evidence that our results can be generalized for other problem classes, different encodings and a related local search methods such as Min-Conflicts on Constraint Satisfaction Problems.

Of great interest in this context is the question for theoretical explanations of the behaviour we have characterized in this work. Particularly interesting here is the fact that the ideal *sr/ms*-characteristic is only observed for certain problem classes and given the optimal walk-probability. This suggests, that there are structural features of the searchspace which are responsible for the observed behaviour. To identify these could give valuable insights into the very nature of the problem class under consideration as well as on the algorithm that has been applied. In this respect we suggest that the work presented here might be regarded as a first step into the direction of learning more about the intricate interrelationship between problem representation and solution mechanisms which we regard as one of the central aspects of AI research.

References

[1] A. Beringer, G. Aschemann, H.H. Hoos, M. Metzger, and A. Weiß. GSAT versus Simulated Annealing. In *ECAI'94*, pages 130–134. John Wiley & Sons, 1994.

[2] B. Cha and K. Iwama. Performance test of local search algorithms using new types of random CNF formulas. In *IJCAI'95*, pages 304–310, 1995.

[3] I. P. Gent and T. Walsh. An empirical analysis of search in GSAT. *(Electronic) Journal of Artificial Intelligence Research*, 1:47–59, 1993.

[4] I. P. Gent and T. Walsh. Towards an understanding of hill–climbing procedures for SAT. In *AAAI'93*, pages 28–33. MIT press, 1993.

[5] S. Hölldobler, H. H. Hoos, A. Strohmaier, and A. Weiß. The GSAT/SA-Familiy – Relating greedy satisifability testing to simulated annealing. Technical Report AIDA-94-17, TH Darmstadt, 1994.

[6] H. H. Hoos. Solving hard combinatorial problems with GSAT — a case study. In *KI-96*, volume 1137 of *Lecture Notes in AI*, pages 107–119. Springer Verlag, 1996.

[7] K. Iwama and S. Miyazaki. SAT-variable complexity of hard combinatorial problems. In *Proceedings of the World Computer Congress of the IFIP*, pages 253–258 (Volume 1). Elsevier Science B.V., North Holland, 1994.

[8] Kalev Kask and Rina Dechter. SAT and local consistency. In *IJCAI'95*, pages 612–622, 1995.

[9] H. Kautz and B. Selman. Pushing the Envelope: Planning, Propositional Logic, and Stochastic Search. In *AAAI'96*, volume 2, pages 1194–1201. MIT Press, 1996.

[10] S. Minton, M.D. Johnston, A.B. Philips, and P. Laird. Minimizing conflicts: A heuristic repair method for constraint satisfaction and sceduling problems. *Artificial Intelligence*, 52:161–205, 1992.

[11] B. Selman and Henry A. Kautz. Domain-independent extensions to GSAT: Solving large structured satisfiability problems. In *IJCAI'93*, pages 290–295, 1993.

[12] B. Selman, Henry A. Kautz, and Bram Cohen. Noise strategies for improving local search. In *AAAI'94*, pages 337–343. MIT press, 1994.

[13] B. Selman, H. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In *AAAI'92*, pages 440–446. MIT press, 1992.