Stochastic Search and Phase Transitions: AI Meets Physics
(extended abstract)

Bart Selman
AT&T Bell Laboratories
Murray Hill, NJ 07974 U.S.A.
selman@research.att.com
http://www.research.att.com/orgs/ssr/people/selman/

Abstract

Computationally hard instances of combinatorial problems arise at a certain critical ratio of constraints to variables. At the critical ratio, problem distributions undergo dramatic changes. I will discuss how an analogous phenomenon occurs in phase transitions studied in physics, and how experiments with critically constrained problems have led to surprising new insights into average-case complexity and stochastic search methods in AI.

1 Introduction

Many AI formalisms, such as used in planning, reasoning, and learning, have been shown to be inherently intractable. These intractability results are generally based on a worst-case analysis, and hence, there has been much debate about their practical relevance. Average-case complexity analysis, on the other hand, would appear to be more directly applicable. However, such analysis requires a precise model of the distribution of input instances. Since we do not have a good understanding of real-world problem distributions, average-case results generally assume relatively simple distributions of randomly generated problem instances.

Average-case results based on such distributions show that almost all randomly generated instances of combinatorial problems are surprisingly easy to solve. This has led some AI researchers to dismiss many of the negative worst-case complexity results. However, as discussed below, recent work has shown that the positive average-case results are largely due to the particular choice of input distribution. By being more careful in generating instances, one can in fact quite easily obtain extremely hard search and reasoning problems. The key property of such hard random instances is that they have to be *critically constrained*. That is, they occur at a certain critical ratio of variables to constraints. At this ratio, the problem distributions undergo dramatic changes. I will discuss how an analogous phenomenon occurs in phase transitions studied in physics, and how tools from statistical mechanics can be used to analyze the transition phenomenon.

In the second part of my talk, I will show how the recent new insights into computationally hard problems have led to the development of powerful new stochastic search methods. These methods now offer a viable alternative to the more traditional systematic methods.

2 Hard problems and phase transitions

In [7], Cheeseman *et al.* study the computational cost of solving randomly generated graph coloring problems. Starting with a given number of nodes, a random graph is generated by adding edges between randomly chosen pairs of nodes [4]. In graph coloring, the goal is to assign a color to each node in such a way that no two nodes connected by an edge have the same color. Cheeseman *et al.* consider the computational cost of coloring random graphs with a fixed number of colors, using a backtrack-style algorithm.

The average cost of coloring random graphs was found to be directly dependent on the ratio of the number of edges to the number of nodes. When graphs contain relatively few edges, they tend to have many valid colorings, and a backtrack style algorithm can find a coloring quite easily — early on in its search. On the other extreme, when a graph contains many edges, a valid coloring often does not exist. Because of the many potential conflicts between the nodes, the global inconsistency of the coloring problem can be detected relatively easily. Finally, at a certain critical ratio of edges to nodes, it becomes quite difficult for a coloring procedure to determine whether the graphs are colorable or not. Intuitively, the instances are "critically constrained". The average computational cost of running the coloring procedure on the critically constrained graphs scales exponentially with the size of the graphs. So, determining whether such graphs are colorable becomes infeasible even for moderate size graphs.

In our own work on Boolean satisfiability (SAT) testing, we also observed a transition phenomenon [48]. Our initial interest in the satisfiability problem arose from early reports that many satisfiability problems are easily solvable. For example, Goldberg [25] describes a class of random SAT problems that are surprisingly easy for the Davis-Putnam satisfiability procedure [11]. Goldberg's work led to an extensive theoretical exploration of his particular random instance model. He considered CNF formulas. Each formula consists of a conjunction of disjunctions (clauses) of literals. Each clause is generated by selecting literals with some fixed probability. This leads to clauses of varying length. A rigorous analysis, reported in a series of papers [15; 16; 26; 54], has shown that in this model, the *average-case* complexity is polynomial for *almost all* choices of parameter settings. In other words, it is difficult to generate computationally hard problem instances. Note that this does not mean that hard instances do not exist; it is simply means that such instances

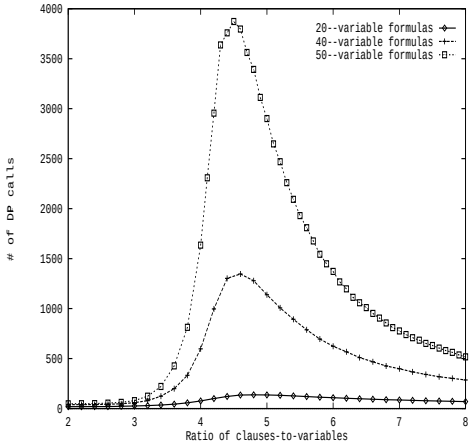

Fig. 1 Solving 3SAT problems.

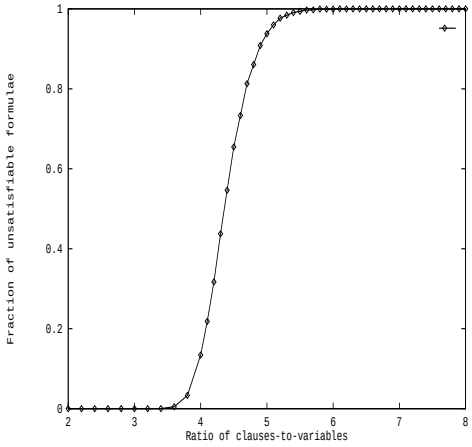

Fig. 2 Fraction of unsatisfiable 3SAT problems.

are very rare.

In [48], we show how by using a different model for generating random formulas, called the *fixed-clause-length* model, one can easily generate hard problem instances. Consider generating a random 3CNF formula. Each clause is generated by randomly selecting three variables from among N variables; each of these variables is negated with probability 0.5. We generate a total of M clauses. We found that the key in generating computationally hard instances is the ratio between M and N. Fig. 1 shows the median cost of solving randomly generated instances at different ratios of variables to clauses. We see that the cost peaks at around 4.3 clauses per variable.[1] Our experimental data shows that at this point the cost of determining satisfiability grows exponentially with the size of the formulas.

Fig. 2 gives the fraction of formulas that are unsatisfiable as a function of the ratio of clauses to variables. At low ratios, few clauses compared to the number of variables, almost all instances are satisfiable (*i.e.*, the unsatisfiable fraction is almost zero). At relatively high ratios of clauses to variables, in a sense too many constraints, almost all randomly generated instances are unsatisfiable. A sudden change occurs around the critical ratio of 4.3. At this ratio, there is a *phase transition*, from the mostly satisfiable phase to the mostly unsatisfiable phase. From Fig. 1, we see that the phase transition region coincides with the area with the hardest problem instances. In this region, the instances are again critically constrained. In the next section, we will take a closer look at what happens inside the phase transition region.

The randomly generated critically constrained problem instances have been used extensively in the study and development of algorithms for graph coloring and satisfiability testing. A key question is whether the results obtained for such instances are at all indicative of the behavior of the algorithms on more structured, real-world instances. The results of the recent DIMACS Challenge on Satisfiability Testing [68] suggest that the behavior of algorithms on hard random problems can indeed be representative of the behavior on more structured problems. The DIMACS Benchmark Problem Set contained several hard random instances and numerous more structured problems. The satisfiability algorithms fell in two categories: complete systematic procedures, and incomplete stochastic methods. These methods complement each other, in that there are problem classes where the stochastic methods are best, whereas on other problem classes the systematic methods are superior. However, within each category, algorithms that were fastest on the hard random instances usually also performed best on the more structured problems. Apparently, the hard random instances do exercise the various time critical parts of the algorithms. Therefore, the performance of algorithms on such hard random instances is a reasonably good indicator of the overall performance on a more diverse set of problem instances.

Aside from being useful as benchmark problems, there is also some indication that critically constrained problems may occur naturally in real-world applications. Nemhauser [50] studied a large airline scheduling problem, involving approximately 500 planes. After a substantial computational effort, his group found a provably optimal solution. Given that the original schedule was only approximately optimal, it was expected that an optimal solution would lead to a savings of one or more planes. However, quite surprisingly, the optimal schedule did not save a single plane. The explanation appears to be that the problem had become critically constrained: Because of economic factors, the airline had assigned additional routes to planes that were idle during parts of the day in the original schedule. So, external factors can give rise to critically constrained real-world planning and scheduling problems.[2]

3 A closer look at the phase transition

Fig. 3 shows the phase transition for 3SAT for several different values of N (the number of variables). Note how the threshold function sharpens up for larger values of N. In [40], we show that the threshold has characteristics typical of phase transitions in the statistical mechanics of disordered materials.

Physicists have studied phase transition phenomena in great detail because of the many interesting changes in a system's

[1] For large N, this ratio converges to around 4.25. [9; 40].

[2] The ratio of constraints to variables will probably differ from the critical ratios found in hard random instances, because of the inherent internal structure of real-world problems. In this abstract, I cannot do justice to the large amount of recent work in this area. The reader is encouraged to consult any of the following additional references [2; 5; 8; 9; 21; 22; 31; 43; 44; 70; 71].

macroscopic behavior that occur at phase boundaries. One useful tool for the analysis of phase transition phenomena is called *finite-size scaling*. This approach is based on rescaling the horizontal axis by a factor that is a function of N. The function is such that the horizontal axis is stretched out for larger N. So, in effect, rescaling "slows down" the phase transition for higher values of N, and thus gives us a better look inside the transition. Fig. 4a shows the result of rescaling the curves from Fig. 3. The original curves are rescaled into a single universal curve.[3] The fit becomes better for higher values of k (the clause length). See, for example, Fig. 4b for rescaled 4SAT data.

From the universal curve, applying the scaling function backwards, the actual transition curve for each value of N can be derived. This approach also localizes the 50%-satisfiable-point for any value of N, which allows us to generate the hardest possible SAT instances.

Finite-size scaling can also be used to study properties other than satisfiability in the critical region. See [64] for a rescaling of computational cost curves, and [57] for a rescaling of the prime implicate function.

4 Stochastic search

In Section 2, we discussed how we can generate satisfiability problems that are hard for the Davis-Putnam procedure. A natural question to consider is whether there are other methods that are better at solving such instances. Recent experimental work has shown that from among the *systematic* search procedures, the basic Davis-Putnam procedure is in fact the most effective [6; 9; 14; 17; 68]. In [65], we show however that a *stochastic* method can outperform the Davis-Putnam procedure. Our method, called GSAT, is based on a randomized local search strategy [46; 51].

The original impetus for trying a local search method on satisfiability problems was the successful application of such methods for finding solutions to large N-queens problems, first using a connectionist system [1], and then using greedy local search [47]. We originally assumed that simply indicated that N-queens was an *easy* problem, and felt that such techniques would fail in practice for SAT. In particular, it would seem that local search methods would easily get stuck in local minima, with a few clauses remaining unsatisfied. Our GSAT experiments have shown, however, that certain local search strategies often do reach global minima.

The basic GSAT procedure (Fig. 5) starts with a randomly generated truth assignment. It then changes ('flips') the assignment of the variable that leads to the greatest decrease in the total number of unsatisfied clauses. Such flips are repeated until either a satisfying assignment is found or a pre-set maximum number of flips (MAX-FLIPS) is reached. This process is repeated as needed, up to a maximum of MAX-TRIES times.

In [65], we show that GSAT substantially outperforms backtracking search procedures, such as the Davis-Putnam procedure, on various classes of formulas, including randomly generated formulas and SAT encodings of graph coloring problems [34].

[3] We do not yet have a closed form expression for the universal curve, but the function $e^{-2^{-\alpha'}}$, where α' is the rescaled M/N ratio, is a good approximation [40].

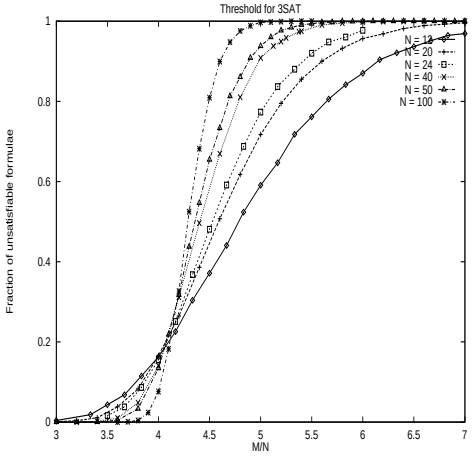

Fig. 3. The 3SAT phase transition sharpens up.

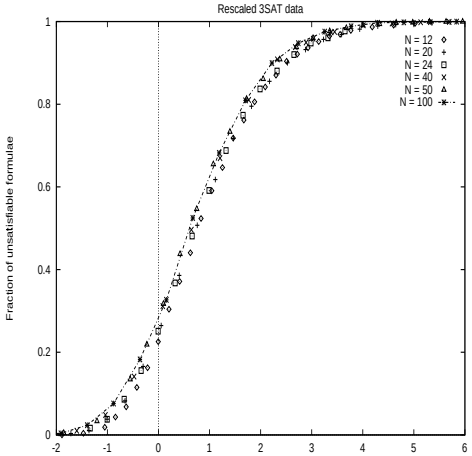

Fig. 4a. Rescaling the 3SAT phase transition. Along the rescaled horizontal axis is $\alpha' = N^{1/\nu}(\alpha - \alpha_c)/\alpha_c)$, where $\alpha = M/N$ (from Fig. 3), $\nu = 1.5, \alpha_c = 4.2$.

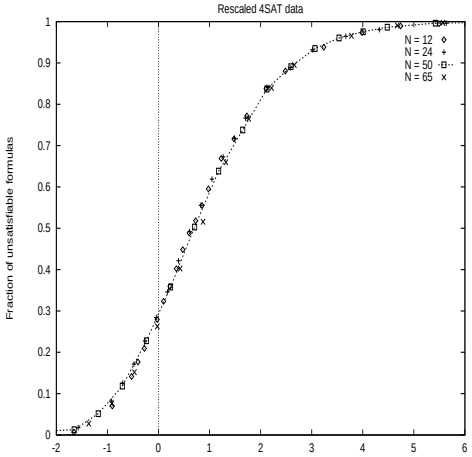

Fig. 4b. Rescaled 4SAT data.

Fig. 6 illustrates a typical search, plotting the number of unsatisfied clauses as a function of the number of flips performed. From the figure it is clear that the search begins with a rapid greedy descent followed by a long sequences of sideways moves. We refer to each sequence of sideways moves as a *plateau*. Note that actual uphill moves almost never occur [20;

```
Procedure GSAT
for i := 1 to MAX-TRIES
    T := a randomly generated truth assignment
    for j := 1 to MAX-FLIPS
        if T satisfies α then return T
        Flip any variable in T that results in greatest
              decrease (can be 0 or negative)
              in the number of unsatisfied clauses
    end for
end for
return "No satisfying assignment found"
```

Fig. 5. The GSAT procedure.

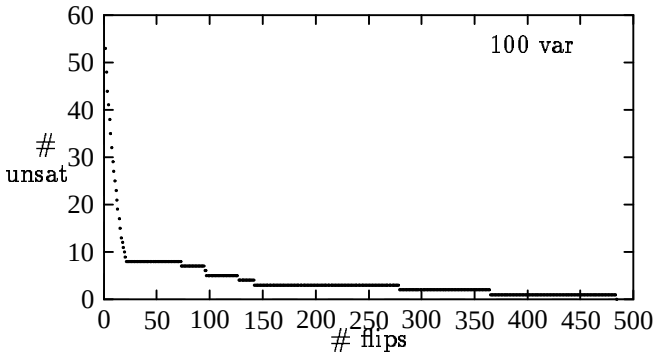

Fig. 6 An example of GSAT's search.

28; 60].

It is clear GSAT spends most of its time moving from plateau to plateau. We have studied various modifications of GSAT in order to speed up this process [61; 62]. One of the most successful strategies is to introduce some noise into the search in the form of uphill moves:

Biased Random Walk
With probability p, follow the standard GSAT scheme, *i.e.*, make the best possible flip.
With probability $1 - p$, pick a variable occuring in some unsatisfied clause and flip its truth assignment. (Note: a possible uphill move.)

The parameter p controls the amount of noise. In Table 1, from [62], we compare the random walk strategy on hard random 3CNF formulas with basic GSAT, simulated annealing [39], and a random noise strategy. We see that the biased walk strategy significantly outperforms the other methods. It enables us to solve hard random instances with over 2000 variables. In contrast, the current best systematic search methods can only handle instances with up to around 400 variables. In [62; 33], we show how stochastic search can also be effective on highly structured instances, such as encodings of circuit design problems, Steiner tree problems, and problems in finite algebra. Some of our problem encodings contain over 20,000 variables and around 500,000 clauses. This is a good indication of the significant progress that has been made in recent years: up to around 1990, experimental work on satisfiability testing was limited to instances with less than a hundred variables and at most one or two hundred clauses. For further experiments with, and comparisons to GSAT and GSAT-style procedures, see also [3; 10; 13; 24; 41; 49; 56; 67; 68].

5 Applications of stochastic methods in reasoning and search

Since GSAT does not systematically explore the space of all truth assignments, if no satisfying assignment is found, it does not mean that no such assignment exists. In other words, GSAT, like other stochastic methods, is inherently incomplete [18]. As a consequence, such methods are suitable for *model* finding but not for theorem proving, which requires showing inconsistency.[4] The success of stochastic search methods suggests, therefore, that it may be useful to encode AI tasks in terms of model-finding, rather than the more usual formulations based on theorem proving. In [37], we show how planning can be encoded as a model-finding task. Certain forms of abduction and default reasoning can also be formulated in terms of model-finding [55; 66]. When no direct model-based encoding exists, one can consider approximate encodings by adapting methods developed for theory compilation [63]. Finally, model-finding procedures can also be used to find representative (or characteristic) models of a knowledge base. Such models can be used to answer certain classes of queries in a highly efficient manner [38; 45].

6 Conclusions

We have discussed the significant progress that has been made recently in our understanding of the nature of computationally hard problems. The hardest problem instances occur at certain critical ratios of constraints to variables. At such ratios, we observe dramatic changes in the problem distributions. This phenomenon can be analyzed with tools from statistical physics. We also discussed new stochastic methods for solving hard problem instances. These methods can substantially outperform systematic methods. Finally, we described how such stochastic methods suggest new ways of dealing with computational challenges in reasoning and search in AI.

References (extended)

[1] Adorf, H.M. and Johnston, M.D. A discrete stochastic neural network algorithm for constraint satisfaction problems. *Int. Joint Conf. on Neural Networks* (1990).

[2] Baker, A. Intelligent backtracking on the hardest constraint problems. *J. of Artificial Intelligence Res.*. (to appear)

[3] Beringer, A., Aschemann, G., Hoos, H., Metzger, M., and Weiss, A. GSAT versus simulated annealing. *Proc. ECAI-94*, Amsterdam, The Netherlands.

[4] Bollobas, B. *Random Graphs.* Academic Press, London (1985).

[5] Broder, A., Frieze, A., and Upfal, E. On the satisfiability and maximum satisfiability of random 3-CNF formulas. *4th ACM Symp. Discr. Alg.* (1993) 322–330.

[6] Buro, M. and Kleine-Büning, H. Report on a SAT competition. Techn. Rep. # 110, Dept. of Math. and Inform., University of Paderborn, Germany (1992).

[7] Cheeseman, P., Kanefsky, B., Taylor, and William M., Where the really hard problems are. *Proc. IJCAI-91* (1991) 331–336.

[8] Clearwater, S.H., Huberman, B.A., Hogg, T. Cooperative solution of constraint satisfaction problems. *Science*, 254 (1991).

[9] Crawford, J.M. and Auton, L.D. Experimental results on the cross-over point in satisfiability problems. *AAAI-93* (1993) 21–27. (Ext. version in Artif. Intel.)

[10] Davenport, A.J., Tsang, E.P.K, Wang, C.J., and Zhu, K. GENET: A connectionist architecture for solving constraint satisfaction problems by iterative improvement. *AAAI-94* (1994) 325–330.

[11] Davis, M. and Putnam, H. A computing procedure for quantification theory. *J. of the ACM*, 7 (1960) 201–215.

[12] Dechter, R. Constraint networks. *Encyclopedia of Artificial Intelligence* John Wiley, New York (1991) 276-285.

[4] An interesting possibility to consider is whether one could devise randomized local search methods that directly explore the space of proofs.

formula		GSAT						Simul. Ann.	
		basic		walk		noise			
vars	clauses	time	flips	time	flips	time	flips	time	flips
100	430	.4	7554	.2	2385	.6	9975	.6	4748
200	860	22	284693	4	27654	47	396534	21	106643
400	1700	122	2.6×10^6	7	59744	95	892048	75	552433
600	2550	1471	30×10^6	35	241651	929	7.8×10^6	427	2.7×10^6
800	3400	*	*	286	1.8×10^6	*	*	*	*
1000	4250	*	*	1095	5.8×10^6	*	*	*	*
2000	8480	*	*	3255	23×10^6	*	*	*	*

Table 1: Comparing noise strategies on hard random 3CNF formulas. (Time in seconds on an SGI Challenge.)

[13] Dechter, R. and Kask, K. GSAT and local consistency. *Proc. IJCAI-95*, Montreal, Canada (1995).

[14] Dubois, O., Andre, P., Boufkhad, Y., and Carlier, J. Can a very simple algorithm be efficient for solving the SAT problem? DIMACS Chall. on SAT Testing, 1993.

[15] Franco, J. and Paull, M. Probabilistic analysis of the Davis Putnam procedure for solving the satisfiability problem. *Discrete Applied Math.* 5 (1983) 77–87.

[16] Franco, J. Elimination of infrequent variables improves average case performance of satisfiability. *SIAM J. Comput.*, 20(6) (1991) 1119–1127.

[17] Freeman, J.W. Hard random 3-SAT problems and the Davis-Putnam procedure. *Artificial Intelligence*. (to appear)

[18] Freuder, F., Dechter, R., Ginsberg, M., Selman, B., and Tsang, E. Systematic versus stochastic constraint satisfaction. *Proc. IJCAI-95*, Montreal, Canada (1995).

[19] Freuder, E. and Mackworth, A. (Eds.). *Constraint-based reasoning.* MIT Press, Cambridge, MA, USA, 1994.

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

[21] Gent, I.P. and Walsh, T. Easy problems are sometimes hard. *Artificial Intelligence*, 70 (1994) 335–345.

[22] Gent, I.P. and Walsh, T. Computational phase transitions in real problems. Research Report, Dept. of AI, Edinburgh (1995).

[23] Ginsberg, M.L. Dynamic backtracking. *J. of Artif. Intel. Res.*, 1 (1993) 25–46.

[24] Ginsberg, M.L. and McAllester, D.A. GSAT and dynamic backtracking. In *Proc. KR-94*, Bonn, Germany, 1994.

[25] Goldberg, A. On the complexity of the satisfiability problem. Courant Comp. Sci. Rep., No. 16, New York University, NY, 1979.

[26] Goldberg, A., Purdom, Jr. P.W., and Brown, C.A. Average time analysis of simplified Davis-Putnam procedures. *Information Process. Lett.* 15 (1982) 72–75.

[27] Gu, J. Efficient local search for very large-scale satisfiability problems. *Sigart Bulletin*, vol. 3, no. 1 (1992) 8–12.

[28] Hampson, S. and Kibler, D. Plateaus and plateau search in Boolean satisfiability problems. DIMACS Chall. on Satisfiability Testing, 1993.

[29] Hansen J. and Jaumard, B. Algorithms for the maximum satisfiability problem. *Computing*, 44, 1990, 279–303.

[30] Hogg, T., Huberman, B.A., and Williams, C.P. (Eds.). Phase Transitions in Problem Solving. *Artificial Intelligence*. (Spec. Issue; to appear)

[31] Hogg, T. and Williams, C.P. Expected gains from parallelizing constraint solving for hard problems. *AAAI-94* (1994) 331–336.

[32] Hogg, T. and Williams, C.P. The hardest constraint problems: a double phase transition. *Artif. Intell.*, 69 (1994) 359–377.

[33] Jiang, Y., Kautz, K., and Selman, B. Solving problems with hard and soft constraints using a stochastic algorithm for MAX-SAT. AI/OR Wrksh., OR (1995).

[34] Johnson, D.S., Aragon, C.R., McGeoch, L.A., and Schevon, C. Optimization by simulated annealing: an experimental evaluation; part II. *Oper. Res.*, 39 (1991).

[35] Kautz, H.A. Hard problems in soft AI. Comp. and Thought Award Lect., *IJCAI-89*.

[36] Kautz, H. and Selman, B. Hard problems for simple default theories. *Artif. Intel.*, 49 (1991) 243–279.

[37] Kautz, H.A. and Selman, B. Planning as satisfiability. *ECAI-92*, Vienna, Austria.

[38] Kautz, H.A., Kearns, M., and Selman, B. Reasoning with characteristic models. *Proc. AAAI-93*, Washington, DC. Ext. version in *Artif. Intel.*.

[39] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. Optimization by simulated annealing. *Science*, 220 (1983) 671–680.

[40] Kirkpatrick, S. and Selman, B. Critical Behavior in the Satisfiability of Random Boolean Expressions. *Science*, 264 (May 1994) 1297–1301. Also p. 1249: "Math. Logic: Pinning Down a Treacherous Border in Logical Statements" by B. Cipra.

[41] Konolige, K. Easy to be hard: Difficult problems for greedy algorithms. *KR-94*, Bonn, Germany (1994) 374–378.

[42] Langley, P. Systematic and nonsystematic search strategies. *AI Planning Systems: Proc. 1st Intl. Conf.* (1992) 145–152.

[43] Langton, C.G. Computation at the edge of chaos: Phase transitions and emergent computation. *Physica D*, 42 (1990) 12–37.

[44] Larrabee, T. and Tsuji, Y. Evidence for a satisfiability threshold for random 3CNF formulas, *Proc. AAAI Symp. on AI and NP-hard problems*, Palto Alto, CA (1993).

[45] Levesque, H.J. Making believers out of computers. *Artificial Intelligence*, 30 (1986) 81–108.

[46] Lin, S. and Kernighan, B.W. An efficient heuristic algorithm for the traveling-salesman problem. *Oper. Res.* 21 (1973) 498–516.

[47] Minton, S., Johnston, M.D., Philips, A.B., and Laird, P., Solving large-scale constraint satisfaction an scheduling problems using a heuristic repair method. *Proc. AAAI-90*, 17–24. Full version in *Artificial Intelligence*, 58 (1992) 161–205.

[48] Mitchell, D., Selman, B., and Levesque, H.J. Hard and easy distributions of SAT problems. *Proc. AAAI-92*, San Jose, CA (1992) 459–465. Full version to appear in *Artificial Intelligence*.

[49] Morris, P. Breakout method for escaping from local minima. *Proc. AAAI-93*.

[50] Nemhauser, G., Barnhart, C., Hane, C., Johnson, E., Marsten, R., and Sigismondi, G. The fleet assignment problem. AI/OR Org. Workshop, Plymounth, VT (1994).

[51] Papadimitriou, C.H. and Steiglitz, K. *Combinatorial optimization.* Englewood Cliffs, NJ: Prentice-Hall, Inc. (1982).

[52] J. Pearl. *Heuristics: intelligent search strategies for computer problem solving.* Addison Wesley (1984).

[53] Pinkas, G. and Dechter, R. On improving connectionist energy minimization. *J. of Artif. Intel. Res.* (to appear)

[54] Purdom, Jr. P.W., and Brown, C.A. Polynomial average-time satisfiability problems. *Inform. Sci.*, 41 (1987) 23–42.

[55] Reiter, R. A logic for default reasoning. *Artificial Intelligence*, 13 (1980) 81–132.

[56] Resende, M.G.C. and Feo T.A. A GRASP for MAX-SAT. *Proc. DIMACS Chall. on Satisfiability Testing*, Piscataway, NJ, 1993.

[57] Schrag, R. and Crawford, J. Implicates and prime implicates. *Artif. Intel.* (to appear)

[58] Selman, B. Domain-specific complexity tradeoffs. *ECAI-94*, The Netherlands.

[59] Selman, B. Near-Optimal Plans, Tractability, and Reactivity. *KR-94*, Bonn, Germany (1994) 521–529.

[60] Selman, B. and Kautz, H.A. An empirical study of greedy local search strategies for satisfiability testing. *Proc. AAAI-93*, Wash., DC (1993) 46–51.

[61] Selman, B. and Kautz, H.A. Domain-independent extensions to GSAT: Solving large structured satisfiability problems. *IJCAI-93*, France (1993) 290–295.

[62] Selman, B., Kautz, H.A., and Cohen, B. Noise strategies for improving local search. *AAAI-94*, Seattle, WA (1994) 337–343.

[63] Selman, B. and Kautz, H.A. Knowledge compilation and theory approximation. *J. of the ACM.* (to appear)

[64] Selman, B. and Kirkpatrick, S. Critical behavior in the computational cost of satisfiability testing. *Artificial Intelligence.* (to appear)

[65] Selman, B., Levesque, H.J., and Mitchell, D. A new method for solving hard satisfiability problems. *Proc. AAAI-92*, San Jose, CA (1992) 440–446.

[66] Selman, B. and Levesque, H. Support set selection for abductive and default reasoning. *Artificial Intelligence*. (to appear)

[67] Spears, W.M. Simulated annealing for hard satisfiability problems. *Proc. DIMACS Chall. on Satisfiability Testing*, Piscataway, NJ, 1993.

[68] Trick, M. and Johnson, D. (Eds.) *Proc. DIMACS Challenge on Satisfiability Testing.* Piscataway, NJ, 1993. (DIMACS Series on Discr. Math.)

[69] Tsang, E.P.K. *Foundations of constraint satisfaction.* Academic Press, 1993.

[70] Williams, C.P. and Hogg, T. Using deep structure to locate hard problems. *Proc. AAAI-92*, San Jose, CA (1992) 472–277.

[71] Wolfram, S. Universality and complexity in cellular automata. *Physica D*, 10 (1984) 1–35.