Exploiting phase transitions for fusion optimization problems

Pontus Svenson

Department of Data and Information Fusion
Division of Command and Control Systems,
Swedish Defence Research Agency
SE 164 90 Stockholm, Sweden

http://www.foi.se/fusion

ABSTRACT

Many optimization problems that arise in multi-target tracking and fusion applications are known to be NP-complete, *i.e.*, believed to have worst-case complexities that are exponential in problem size. Recently, many such NP-complete problems have been shown to display threshold phenomena: it is possible to define a parameter such that the probability of a random problem instance having a solution jumps from 1 to 0 at a specific value of the parameter. It is also found that the amount of resources needed to solve the problem instance peaks at the transition point.

Among the problems found to display this behavior are graph coloring (aka clustering, relevant for multi-target tracking), satisfiability (which occurs in resource allocation and planning problem), and the travelling salesperson problem.

Physicists studying these problems have found intriguing similarities to phase transitions in spin models of statistical mechanics. Many methods previously used to analyze spin glasses have been used to explain some of the properties of the behavior at the transition point. It turns out that the transition happens because the fitness landscape of the problem changes as the parameter is varied. Some algorithms have been introduced that exploit this knowledge of the structure of the fitness landscape.

In this paper, we review some of the experimental and theoretical work on threshold phenomena in optimization problems and indicate how optimization problems from tracking and sensor resource allocation could be analyzed using these results.

1. INTRODUCTION

Optimization problems occur frequently in fusion research. Perhaps the simplest example is the association problem when observations from multiple sensors are to be assigned to various tracks in a multi-target tracker. System performance depends on solving this problem fast and accurately. Similar problems arise in the weapon-to-target matching problem, were we are asked to assign the most appropriate system to deal with different enemy objects.

In this paper, we review recent work on *phase transitions* that occur for many important optimization problems and discuss some possible ways of exploiting the phase transitions to solve important optimization problems more quickly. The subject was touched upon in our paper here last year,[1] were we used the extremal optimization algorithm to solve problems that were first transformed to involve fewer constraints. This paper expands considerably on both the background needed to understand the transformation and the description of how to transform the problem.

Further author information:
E-mail: ponsve@foi.se, Telephone: +46 8 5550 37 32

Signal Processing, Sensor Fusion, and Target Recognition XIV, edited by Ivan Kadar,
Proc. of SPIE Vol. 5809 (SPIE, Bellingham, WA, 2005) 0277-786X/05/$15 · doi: 10.1117/12.602741

The paper is outlined as follows. Section 2 gives a brief introduction to computation complexity and some model problems that display the phase transition phenomenon and that can be mapped to a variety of fusion optimization problems. Section 3 explains the phenomenology behind the phase transitions in detail, giving details on an approximate theory for locating the threshold value and describing some work on analyzing the dynamics of algorithms. Section 4, finally, outlines some possible ways of taking advantage of the phase transitions in order to solve fusion optimization problems more quickly.

2. BACKGROUND ON COMPUTATIONAL COMPLEXITY

Computer scientists classify problems according to the maximal amount of resources needed for their solution. The most important resource is time, but it is also possible to distinguish between problems that require qualitatively different amounts of memory. For example, a list of N elements can always be sorted in time less than $kN \log N$, where k is some constant.[2] The problems whose running time on a *universal Turing machine* (e.g.,[3]) is bounded by a polynomial in their size are said to be in the class P. The important class NP (for non-deterministic polynomial) consists of those problems where it can be checked in polynomial time whether a proposed solution actually solves the problem. (A *non-deterministic* Turing machine would be able to solve NP problems in polynomial time.) It is obvious that $P \subseteq NP$, but there is no proof that $P \neq NP$. However, most people believe that there are NP problems whose worst-case instances take exponential time to solve on a universal Turing machine.

The class NP-complete (or NPC) are the most important problems in NP. A problem of size N is in NPC if all other NP problems can be transformed into it in time at most polynomial in N. A method to solve an NPC problem efficiently can thus be used to solve any NP problem efficiently. It is known that if $P \neq NP$ then there are problems in NP that are in neither P nor NPC. A problem is called NP-hard if it is at least as difficult as the most difficult NP problems; NPC is the intersection of NP and NP-hard. A modern reference on complexity theory and NP problems is,[4] while[5] has an extensive list of NPC problems.

Two important problems in NPC are graph coloring (K-COL) and satisfiability testing (**SAT**). Graph coloring is the problem of coloring a graph with N vertices and M edges using K colors so that no two adjacent vertices have the same color.

The most natural application of graph coloring is in scheduling or association problems. For example, a school where each teacher and student can be involved in several different classes must schedule the classes so that no collisions occur. If there are K different time slots available, this is K-COL. Similarly, a multi-target tracker that should assign N reports to K targets solves the K-COL problem.

Satisfiability was the first problem shown to be in NPC.[6] It is the problem of finding an assignment of true or false to N variables so that a boolean formula in them is satisfied. In K-SAT, this formula is written in *conjunctive normal form* (CNF), that is, it consists of the logical **AND** of M clauses, each clause being the **OR** of K (possibly negated) variables, where the same clause may appear more than once in a formula. For example, $(x \vee y) \wedge (y \vee \neg z)$ is an instance of **2-SAT** with two clauses and three variables. Applications of K-SAT include theorem proving, VLSI design, and learning. Important fusion applications of K-SAT include resource allocation and planning.[7] Resource allocation problems that can be mapped to K-SAT occur, for example, in problems where there are several active sensors and several targets and where each sensor can view several of the targets. In order to minimize the number of sensors that are active (to protect them, or to conserve energy), we can form a boolean expression that includes constraints so that all targets are viewed and seek a solution where the minimum possible number of sensors are used.

In K-SAT, each clause forbids one of the 2^K possible assignments for its variables. In the same way, an edge in a graph forbids K of the K^2 different colorings of its vertices. For both problems, there are M *constraints* on the solutions. The energy ϵ of a problem instance is defined as the number of unsatisfied constraints per variable. Both K-SAT and K-COL are in P for $K = 2$ and in NPC for $K \geq 3$.[5] The related problem (**MAX-K-SAT**) of trying to minimize the number of unsatisfied clauses in K-SAT is in NPC even for $K = 2$.

Many important problems are such that the number of constraints is of the same order as the number of variables, $M = \alpha N$. The scheduling problem described above fulfills this condition, for example. For graph

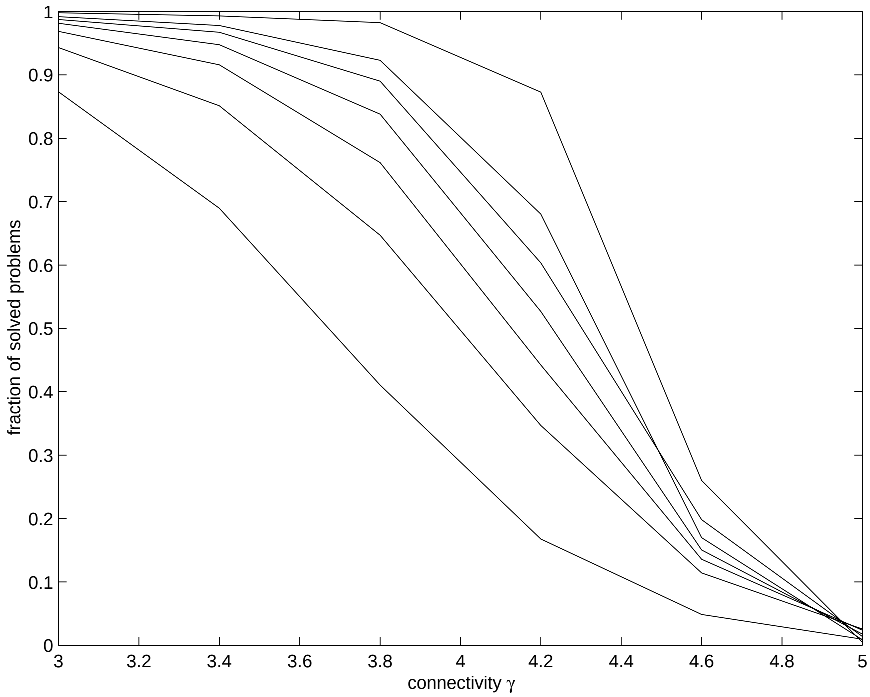

Figure 1. This figure illustrates the phase transition in solvability that occurs at $\alpha \approx 4.3$ for the 3-COL problem. The plot shows the fraction of colorable graphs for system sizes ranging from 10 (leftmost curve) to 100. The Brelaz algorithm was run on up to 50000 different graphs for each value of γ. The gradual sharpening of the transition as system size increases is indicative of finite-size scaling. Note the restricted range on the x-axis.

coloring, $\alpha = \gamma/2$, where γ is the *connectivity* of the graph. The connectivity (or average degree) is defined as the mean number of edges exiting each node. For a graph with n vertices and e edges, it is $2e/n$. Below, we will use α for K-SAT and γ when we talk of K-COL. We will concentrate on K-COL; most results for K-SAT are similar.

3. THE TRANSITION

For some optimization problem, it is possible to define a constrainedness parameter so that a randomly chosen problem instance that has a low degree of constrainedness is always solvable, while one that is highly constrained never has a solution. This is of course in a sense trivial, but it is surprising that the boundary between the two cases is sharp. Kirkpatrick and Selman[8] have shown that this transition sharpens as problem size is increased and that finite size scaling can be used to describe it. Friedgut and Achlioptas[9,10] have shown rigorously that there is a sharp transition for all problem sizes. Note that some of the methods commonly used to generate more complicated random constraint satisfaction problems have been shown not to have a transition in the thermodynamic limit.[11] The transition can be seen in figure 1, which shows how the fraction of solvable problems changes from 1 to 0 for several different problem sizes.

Related to this phase transition in problem solvability, there is a transition in how difficult it is to solve a problem or show that no solutions exist.[12–14] This transition is sometimes referred to as the "easy-hard-easy" transition. Is it very easy to find a solution for underconstrained problems — since most variable assignments do not lead to conflicts with others, not much backtracking will be needed. For overconstrained problems, on the other hand, the increased number of constraints makes the search methods quickly run into inconsistencies and not many nodes of the search tree will have to be examined. For problems in the region between over and underconstrained (termed critically constrained), the search method will have to spend a long time searching through dead ends that it can avoid in the other phases.

By treating all the constraints in the problem as independent, it is possible to make an approximation for the number of solutions of a problem with $M = \alpha N$ constraints and N variables (see, e.g.,[15–17]). The approximation is exact for graphs without loops and for satisfiability problems where no variable is contained in more than one clause.

For simplicity, consider K-SAT. Each constraint here involves k variables and forbids one of the 2^k possible combinations of assignments to these variables. Approximate the probability that a constraint is violated in an assignment by $p_v = \frac{1}{2^k}$. Similarly for K-COL, $p_v = K/K^2 = 1/K$

Assuming that the constraints are independent now gives $(1 - p_v)^M$ as the probability of a formula with M clauses having no clause that is violated. This approximation ignores all correlations between constraints, such as loops in a graph. Multiplying with the number of possible assignments, 2^N, then gives an approximation to the number of solutions for K-SAT

$$N_{\mathrm{sol}} = 2^N (1 - \frac{1}{2^k})^{\alpha N}, \tag{1}$$

while the appropriate expression for K-COL is

$$N_{\mathrm{sol}} = K^N (1 - \frac{1}{K})^{\gamma N/2}. \tag{2}$$

Using the inclusion-exclusion principle it is possible to write an exact expression for N_{sol}.[15] The inclusion-exclusion principle is the generalization of the simple formula

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

from mathematical statistics. If we let A_i be the event that constraint i is violated, it expresses the probability that any (*i.e.*, at least one) constraint is violated in terms of the probabilities of one, two, three or more constraints being violated simultaneously

$$P(\cup_i A_i) = \sum_{r=1}^{M} (-1)^{r+1} S_r, \tag{3}$$

where S_r is the probability of exactly r constraints being violated simultaneously. The number of solutions can now be found as

$$N_{\mathrm{sol}} = N_{\mathrm{tot}}(1 - P(\cup_i A_i)), \tag{4}$$

where N_{tot} is the number of possible assignments of the variables, $N_{\mathrm{tot}} = K^N$ for K-COL and $N_{\mathrm{tot}} = 2^N$ for K-SAT. For K-COL, $S_1 = MK^{-1}$, since there are M edges and each of them eliminates K^{N-1} (of the K^N) solutions. For S_2, we need to express the number of states that are eliminated by each of two edges. This is given by $\binom{M}{2} K^{-2}$, while the expression for

$$S_3 = \binom{M}{3} K^{-3} + (K^{-2} - K^{-3})t, \tag{5}$$

requires knowledge of the number of triangles, t, in the graph. Expression (5) can be understood by noting that if two edges in a triangle are frustrated, the third is always frustrated too. It can be shown that t is Poisson-distributed with mean $\gamma^3/6$. To calculate S_i for $i \geq 4$, we also need to know the distribution of more complex sub-graphs.

The critical value of the parameter can now be approximated as that γ which gives $N_{\mathrm{sol}} = 1$ in (2), giving

$$\gamma_c = -2 \frac{\log K}{\log (1 - \frac{1}{K})}. \tag{6}$$

For $K = 3$, equation (6) gives $\gamma_c = 5.4$ for K-COL and $\alpha_c = 5.2$ for K-SAT. These values are larger than the experimental values of $\gamma_c = 4.6$ and $\alpha_c = 4.21$. For K-SAT, this approximation has been independently introduced several times.[18]

This calculation of the critical value of γ ignores all correlations between different constraints in the problem. It gives an upper bound for γ_c and is analogous to studying a *forest*, a graph without cycles, in which all edges are violated with a probability p. Taking correlations into account reduces the number of solutions.[15]

This annealed approximation gives qualitative explanations for both the solvable-unsolvable transition and the easy-hard-easy pattern of the amount of resources necessary to solve the problem. Mammen and Hogg[19] have found that the size of the smallest minimal unsolvable subproblems shows a behavior that coincides roughly with that of the search cost, and also that search cost appears to be a strictly increasing function of this size. A minimum unsolvable subproblem is a subset of the problem that is unsolvable but becomes solvable if any variable and the constraints in which it appears are deleted from the problem. Obtaining the minimal unsolvable subproblem would thus be a very good heuristic for search algorithm. However, this problem is in general as difficult as finding the optimum solution itself. In recent years, there has been quite a lot work on determining better exact bounds for the phase-transition, see, e.g.,.[20–24]

The K-SAT problem has been studied in detail by Monasson and Zecchina[25–29] and others[30–32] who have found interesting analogies between it and various models from theoretical physics. Among other things, they have found that the entropy stays finite at the transition. This means that the number of solutions of the problem has a discontinuous jump at the transition point, *i.e.*, there are several approximate solutions to the problem at the transition.

It has been established that the occurrence of the sat-unsat-phase transition is due to a finite fraction of the variables in the problem becoming over-constrained, that is they must have the same value in all solutions of the problem.[33] The set of all fully constrained variables is called the backbone and has been compared to percolation. The backbone vanishes in the sat-phase — the presence of any finite number of fully constrained variables could otherwise be used to add an infinitesimal number of clauses that would cause the problem to become unsat. The fraction of sites in the backbone is the proper order parameter for SAT and has been shown to have different behavior for 2-SAT and 3-SAT. For 2-SAT, the fraction smoothly increase above the threshold $\alpha_c = 1$, while for 3-SAT there is a discontinuous or first-order transition. The $2 + p$- SAT problem (in which a fraction p of the clauses have three literals while the rest have two) has a continuous transition for $p \geq 0.4$. It has been shown that even though this problem is in NPC for all $p > 0$, problem instances do not become exponentially hard until $p > 0.4$.[33, 34] This led to some early speculation that there could be a relation between the order of the phase transition in a problem and its worst-case computation properties. Recent results however indicate that this is not the case. Achlioptas et al[35] are the first to calculate the exact position of the threshold for an NPC problem. The problem they analyze is the 1-in- K-SAT problem, which is normal satisfiability but with the added constraint that each clause should have exactly one true literal. They also show that the transition here is second order, thus showing that the order of the transition is in general not related to the problem complexity. This is a very important result, since it means that the hopes of physicist to connect the P = NP question with the order of the transition have been shown to be futile.

Another problem in NPC that also shows a transition[36] is the traveling salesperson problem (**TSP**), where the objective is to find a tour of minimum length visiting N given distinct cities. A difficulty in studying this problem is that there is no natural parameter (like α and γ) that distinguishes between under- and overconstrained problems. To get one, the **TSP** must be reformulated as a decision problem: is there a Hamiltonian path of length less than l? The parameter l plays the same rôle as α — for a given distribution of problems there is an l_c such that if $l \gg l_c$, almost all instances have a tour with length $< l$, but if $l \ll l_c$ practically no such tours exist. Traditionally, most NPC problems are formulated as decision rather than optimization problems.

There are also many NPC problems that contain no obvious parameter which makes it difficult to say if the solvability phase transition (and also the relaxation transition found here) exists in all NPC-problems or in just a few. There have been attempts to formulate a more general parameter (e.g.,[37]), with the drawback that it requires us to approximate the number of solutions.

Phase transitions have also been found in problems beyond NPC, e.g., in **QSAT**,[38] a harder version of satisfiability where the boolean variables are quantified by either ∀ or ∃ (in ordinary **SAT**, all variables are existentially quantified). This problem is known to be PSPACE-complete,[4] meaning that it is at least as hard as all problems that can be solved by a universal Turing machine without time limits but using memory at most polynomial in problem size.

Walsh[39] has made an interesting comparison between search methods for constraint satisfaction problems and renormalisation group flows from the theory of critical phenomena. Walsh has studied how the constrainedness changes during search using a variant of the Davis-Putnam algorithm. this algorithm changes the clauses as it traverses the search tree. This means in particular that the ratio α between clauses and variables will change as the solution is approached.

By plotting the constrainedness as a function of search depth and for different initial values of α, an interesting picture is found. For problems that are critically constrained, the constrainedness does not vary much as search progresses. For overconstrained problems, the constrainedness increases rapidly, while for underconstrained problems it decreases just as rapidly. That is, the constrainedness parameter α shows much the same behavior as the coupling constant of a critical system. Here, starting at the critical coupling temperature means that the coupling constant is constant, while starting above or below the critical temperature will cause the coupling constant to be drawn towards either the high or low temperature fixed point representing the disordered and ordered phase, respectively. The comparison is of course to be expected, but is nonetheless interesting, since it provides a qualitative comparison between search procedures and renormalisation group flows.

A similar — but much more complete — analysis of the phase space of a search method has been performed by Cocco and Monasson.[40–42] In these papers, the authors study the phase diagram of 3-SAT, also using the Davis-Putnam algorithm. Since some variables are also removed from the problem, some clauses might change character from involving three variables to just containing 2. This, too, can be captured using the terminology of the $2+p$- SAT problem — instead of describing a problem instance using just α, we add p and hence get a two-dimensional phase-diagram. The evolution of α and p can now be tracked as the search-algorithm progresses and we thus get a dynamical trajectory in (α, p)-space. Cocco and Monasson find that the DP-algorithm finds a solution very quickly for all $\alpha < 3.03$.

Trajectories in this phase diagram follow three different behaviors, depending on their starting points. Those that start at large α quickly find their path to the unsat fix-point, while those that start at small α find the sat fix-point. For intermediate α, the DP-algorithm need to spend time backtracking before it can decide whether a problem instance is satisfiable or not; thus the trajectories for $\alpha > 3$ go back-and-forth a lot before finally reaching either a solution or exhausting the search-tree. The hardness of the problem instance is in part determined by the number of times that the DP method must cross the sat-unsat-phase border. Cocco and Monasson also manage to get quantitative results for the time needed to solve the problem based on where the trajectory first crosses the border.

In[43] a similar analysis is made for analog computation for a linear programming problem. Their model of analog computation in this case consists of the solution of a differential equation. Majumdar and Krapivsky[44] have performed an extensive analysis of the binary search problem. While this problem is not NP-hard, the analysis is nevertheless interesting since it allows calculation of the height of the tree for an arbitrary dis tribution of elements to sort.

The approximate values where the transition for K-COL happens are shown in table 1 for $2 \leq K \leq 5$. The values were obtained using a non-optimized backtrack-search program with the Brelaz heuristic.[45]

4. EXPLOITING PHASE TRANSITION THEORY

There are several possible way of actually exploiting the phase transition phenomenon in optimization problems. Last year, we briefly mentioned one way of solving the report-to-track association problem more

$K =$	3	4	5
γ_c^{exp}	4.6	8.7	13.1

Table 1. The approximate values of γ_c determined by using the Brelaz algorithm.

quickly. Using the notation $x_i = a$ when report i is placed in cluster a, the problem can be written as

$$\min_{\{x_i\}} C(\{x_i\}), \tag{7}$$

where C denotes the cost of a configuration. The cost includes terms that give the cost of placing reports together, and also the cost of not placing reports together. Most often, all but the pair-wise costs are ignored and the object to minimize is instead written as

$$C(x_i) \approx \sum_{i<j} C(i,j)\delta_{x_i}^{x_j}. \tag{8}$$

If the N^2 items in equation 8 are hard to compute, it would be beneficial to only need to compute some of them.

This goal can be obtained by approximating the problem described in equation 8 by another, which we construct so that it has the same solution. We do this by assuming that the problem has a solution. (This is an approximation which might not be valid in cases where there is a large amount of clutter.) If the problem has a solution, can we find another problem instance, belonging to a different ensemble, that has the same solution but a different $C(i,j)$ matrix, with a smaller number of non-zero entries? We assume that by calculating only $\frac{\gamma N}{2}$ randomly chosen entries from the matrix in equation 8, we get such a problem. By choosing γ as close to γ_c as possible, we get the benefit of using the most constrained possible version of the original problem.

The phase transitions have also inspired new algorithms. In particular, the analogy between the so called "cavity method" and belief propagation has led to new message passing algorithms (see, *e.g.*,[46] and references therein). An amusing example of how to use problem structure when developing an algorithm is,[47] while[48] presents results for choosing appropriate parameters for some different approximation algorithms.

5. CONCLUSIONS

We presented a fairly detailed overview of the phase transition that has been observed and (partially) explained in a number of NP-complete problems. The area is important for fusion research for a number of reasons:

- Approximate solutions to fusion optimization problems could be obtained by mapping them to more simple versions, using the phase transitions as guide to determine how much simpler these could be while still retaining the important properties of the original problem;
- Analyzing algorithms in the manner described in section 3 could provide important clues to improving them;
- Algorithms such as the improved message passing method could be used.

REFERENCES

1. P. Svenson, "Extremal optimization for sensor report pre-processing," in *Signal Processing, Sensor Fusion, and Target Recognition XIII*, I. Kadar, ed., *Proceedings of SPIE* **5429**, p. 162, 2004.
2. D. E. Knuth, *Sorting and Searching*, vol. 3 of *The Art of Computer Programming*, Addison-Wesley, Reading, Ma, 2 ed., 1998.

3. J. E. Hopcroft and J. D. Ulman, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Mass., 1979.
4. C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley, Reading, MA, 1994.
5. M. R. Garey and D. S. Johnson, *Computers and Intractability. A guide to the Theory of NP-Completeness*, W H Freeman, New York, 1979.
6. S. A. Cook, "The complexity of theorem-proving procedures," in *Proc. 3rd Ann. ACM Symp. on Theory of Computing*, pp. 151–158, ACM, (New York), 1971.
7. H. Kautz and B. Selman, "Planning as Satisfiability," in *Proc. 10th European Conference on Artificial Intelligence*, 1992.
8. S. Kirkpatrick and B. Selman, "Critical Behavior in the Satisfiability of Random Boolean Expression," *Science* **264**, pp. 1297–1301, May 1994.
9. E. Friedgut, "Sharp Thresholds of Graph Properties, and the k-SAT Problem." To appear in J. Am. Math. Soc.
10. D. Achlioptas and E. Friedgut, "A Sharp Threshold for k-Colorability," *Random Structures and Algorithms* **14**, p. 63, 1999.
11. D. Achlioptas, L. M. Kirousis, E. Kranakis, D. Krizanc, M. S. O. Molloy, and Y. C. Stamatiou, "Random Constraint Satisfaction: A More Accurate Picture," in *Proceedings of Third International Conference on Principles and Practice of Constraint Programming*, **1330**, pp. 107–120, Springer, (Berlin), 1997.
12. P. Cheeseman, B. Kanefsky, and W. M. Taylor, "Where the Really Hard Problems Are," in *Proceedings of IJCAI-91*, pp. 331–337, Morgan Kauffman, (San Mateo, CA), 1991.
13. T. Hogg and C. P. Williams, "The Hardest Constraint Problems: A Double Phase Transition," *Artificial Intelligence* **69**, pp. 359–377, 1994.
14. D. Mitchell, B. Selman, and H. Levesque, "Hard and Easy Distributions of SAT Problems," in *Proceedings of AAAI-92*, pp. 459–465, AAAI Press, (San José, CA), 1992.
15. T. Hogg, "Applications of Statistical Mechanics to Combinatorial Search Problems," in *Annual Reviews of Computational Physics*, **2**, pp. 357–406, World Scientific, (Singapore), 1995.
16. C. P. Williams and T. Hogg, "Exploiting the Deep Structure of Constraint Problems," *Artificial Intelligence* **70**, pp. 73–117, 1994.
17. C. P. Williams and T. Hogg, "Extending Deep Structure," in *Proc. of AAAI93*, pp. 152–157, AAAI Press, (Menlo Park, CA), 1993.
18. L. M. Kirousis, E. Kranakis, D. Krizanc, and Y. C. Stamatiou, "Approximating the Unsatisfiability Threshold of Random Formulas," *Random Structures and Algorithms* **12**(3), pp. 253–269, 1998.
19. D. L. Mammen and T. Hogg, "A New Look at the Easy-Hard-Easy Pattern of Combinatorial Search Difficulty," *J. of Artificial Intelligence Research* **7**, pp. 47–66, 1997.
20. D. Achlioptas and C. Moore. Manuscript, 2001.
21. L. M. Kirousis, E. Kranakis, and D. Krizanc, "A better upper bound for the unsatisfiability threshold."
22. D. Achlioptas, *Threshold Phenomena in Random Graph Colouring and Satisfiability*. PhD thesis, University of Toronto, 1999. Available at http://www.research.microsoft.com/∼optas/.
23. D. Achlioptas, P. Beame, and M. Molloy, "A Sharp Threshold in Proof Complexity." Manuscript 2001.
24. D. Achlioptas, L. M. Kirousis, E. Kranakis, and D. Krizanc, "Rigorous Results for Random (2+p)-SAT," *Theoretical Computer Science* **265**(1-2), p. 109, 2001.
25. R. Monasson, "Some remarks on hierarchical replica symmetry breaking in finite-connectivity systems," *Phil. Mag. B* **77**(5), pp. 1515–1521, 1998.
26. R. Monasson, "Optimisation problems and replica symmetry breaking in finite connectivity spin-glasses," *J. Phys. A* **31**(2), pp. 513–29, 1998.
27. R. Monasson and R. Zecchina, "Tricritical Points in Random Combinatorics: the (2+p)-SAT case." eprint cond-mat/9810008. To appear in J Phys A.
28. R. Monasson and R. Zecchina, "Statistical Mechanics of the Random K-satisfiability Model," *Phys. Rev. E* **56**(2), pp. 1357–1370, 1997.
29. R. Monasson and R. Zecchina, "Entropy of the K-Satisfiability Problem," *Phys. Rev. Lett.* **76**(21), pp. 3881–3885, 1996.

30. A. Crisanti, L. Leuzzi, and G. Parisi, "The 3-SAT problem with large number of clauses in the ∞-replica symmetry breaking scheme." eprint cond-mat/0108433.
31. M. Leone, F. Ricci-Tersenghi, and R. Zecchina, "Phase coexistence and finite-size scaling in random combinatorial problems." eprint cond-mat/0103200.
32. W. Barthel, A. Hartmann, M. Leone, F. Ricci-Tersenghi, M. Weigt, and R. Zecchina, "Generating hard and solvable satisfiability problems: A statistical mechanics approach." eprint cond-mat/0111153.
33. R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky, "2+p-SAT: Relation of Typical-Case Complexity to the Nature of the Phase Transition." eprint cond-mat/9910080. To appear in Random Structures and Algorithms.
34. R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky, "Computational complexity from 'characteristic' phase transition," *Nature* **400**, p. 133, 1999.
35. D. Achlioptas, A. Chtcherba, G. Istrate, and C. Moore, "The Phase Transition in NAESAT and 1-in-k SAT," in *Symposium on Discrete Algorithms*, p. 721, 2001.
36. I. P. Gent and T. Walsh, "The TSP phase transition," *Artificial Intelligence* **88**, pp. 349–358, 1996.
37. I. P. Gent, E. MacIntyre, P. Prosser, and T. Walsh, "The Constrainedness of Search," in *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, **1**, pp. 246–52, MIT Press, (Boston), 1996.
38. I. P. Gent and T. Walsh, "Beyond NP: the QSAT phase transition." Report APES 05-1998. Available at **http://www.cs.strath.ac.uk/∼apes/apereports.html**.
39. T. Walsh, "The Constrainedness Knife-Edge." Available at http://www.cs.strath.ac.uk/∼apes/apepapers.html.
40. S. Cocco and R. Monasson, "Trajectories in phase diagrams, growth processes and computational complexity: how search algorithms solve the 3-Satisfiability problem." eprint cond-mat/0009410.
41. S. Cocco and R. Monasson, "Analysis of the computational complexity of solving random satisfiability problems using branch and bound search algorithms." eprint cond-mat/0012191.
42. S. Cocco and R. Monasson, "Heuristic average-case analysis of the backtrack resolution of random 3-satisfiability instances."
43. A. Ben-Hur, J. Feinberg, S. Fishman, and H. T. Siegelmann, "Probabilistic analysis of the phase space flow for linear programming." eprint cond-mat/0110655.
44. S. N. Majumdar and P. L. Krapivsky, "Extreme Value Statistics and Traveling Fronts: An Application to Computer Science." eprint cond-mat/0109313.
45. D. Brelaz, "New Methods to Color Vertices of a Graph," *Comm. ACM* **22**, pp. 251–256, 1979.
46.
47. T. Hogg, "Single-step quantum search using problem structure." eprint quant-ph/9812049.
48. S. Seitz, M. Alava, and P. Orponen, "Focused local search for random 3-satisfiability." eprint cond-mat/0501707.