Using Deep Structure to Locate Hard Problems

Colin P. Williams and Tad Hogg

Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304, U.S.A.
CWilliams@parc.xerox.com, Hogg@parc.xerox.com

Abstract

One usually writes A.I. programs to be used on a range of examples which, although similar in kind, differ in detail. This paper shows how to predict where, in a space of problem instances, the hardest problems are to be found and where the fluctuations in difficulty are greatest. Our key insight is to shift emphasis from modelling sophisticated algorithms directly to modelling a search space which captures their principal effects. This allows us to analyze complex A.I. problems in a simple and intuitive way. We present a sample analysis, compare our model's quantitative predictions with data obtained independently and describe how to exploit the results to estimate the value of preprocessing. Finally, we circumscribe the kind problems to which the methodology is suited.

Introduction

The qualitative existence of abrupt changes in computational cost has been predicted theoretically in (Purdom 1983, Franco & Paull 1983, Huberman & Hogg 1987) and observed empirically in (Papadimitriou 1991, Cheeseman, Kanefsky & Taylor 1991). Indeed the latter results sparked fervent discussion at IJCAI-91. But the quantitative connection between theory and experiment was never made. In part, this can be attributed to the theoretical work modelling naive methods whilst the experimental work used more sophisticated algorithms, essential for success in larger cases. With the demonstrable existence of phase transition phenomena in the context of real problems the need for a better theoretical understanding is all the more urgent.

In this paper we present an approach to analyzing sophisticated A.I. problems in such a way that we can predict where (in a space of possible problems instances) problems become hard, where the fluctuations in performance are worst, whether it is worth preprocessing, how our estimates would change if we considered a bigger version of the problem and how reliable these estimates are.

Our key insight over previous complexity analyses is to shift emphasis from analyzing the algorithm directly to analyzing the deep structure of the problem. This allows us to finesse handling the minutiae of real algorithms and still determine quantitative estimates of critical values.

The Problem

We will take constraint satisfaction problems (CSPs) as representative examples of A.I. problems. In general, a CSP involves a set of μ variables, each having an associated set of domain values, together with a set of ν constraints specifying which assignments of values to variables are consistent ("good") and inconsistent ("nogood"). For simplicity we suppose all variables have the same number, b, of domain values. If we call the variable/value pairings "assumptions", a solution to the CSP can be defined as a set of assumptions such that every variable has some value, no variable is assigned conflicting values and all the constraints are simultaneously satisfied.

Even with a fixed μ and b different choices of constraints can result in problems of vastly different difficulty. Our goal is to predict, in a space of problem instances,

1. where the hardest problems lie, and
2. where the fluctuations in difficulty are greatest.

The other concerns mentioned above, e.g., what technique is probably the best one to use, whether it is worth preprocessing the problem, and how the number of solutions and problem difficulty change as larger problem instances are considered, can all be tackled using the answers to these questions.

Our Approach

Usually, an estimate of problem difficulty would be addressed by specifying an algorithm and conducting a complexity analysis. This can be extraordinarily difficult especially for "smart" programs. Instead we advocate analyzing a representative search space rather than the exact one induced by any particular algorithm. Nevertheless, we should choose a space that mimics the *effects* of a clever search algorithm which we can summarize as avoiding redundant and irrelevant search. A space offering the potential for such economies is the directed-lattice of sets of assumptions that an assumption-based truth maintenance system (ATMS) creates (de Kleer 1986).

As assumptions are variable/value pairs the nogoods split into two camps: *necessary nogoods* (because at least

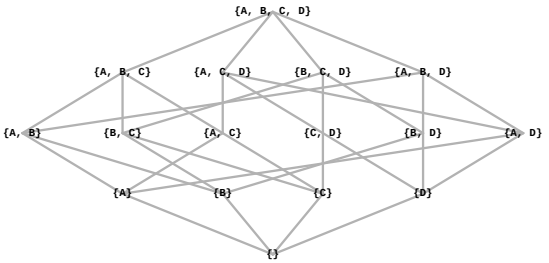

Fig. 1. The lattice of 4 assumptions, $\{\mathtt{A},\mathtt{B},\mathtt{C},\mathtt{D}\}$ showing levels 0 (bottom) through 4 (top).

one variable is assigned multiple values) and *problem dependent nogoods* (in which no variable appears more than once, so its status as a nogood is determined by domain constraints). We will assume our sophisticated A.I. program can make this distinction, allowing it to avoid considering the necessary nogoods explicitly. Thus we shall model a "reduced" lattice containing only the problem dependent nogoods[1]. Moreover, in the lattice model, the solutions are precisely the "goods" at level μ because nodes at higher levels must include at least one necessary nogood and those at lower levels do not have a full complement of variables.

How is difficulty defined? Finding a solution for the CSP involves exploring states in this lattice; the more states examined before a solution is found, the higher the cost. So we shall identify problem difficulty with C_s, the cost to find the first solution or prove there are none. Its precise value depends on the distribution of solutions in the lattice and the search method, but an approximation is given by using the cost to search for all solutions and, when there are solutions, dividing it by the number of solutions to estimate the cost to find just one of them. A simple direct measure of the cost to find all solutions counts the number of goods in the lattice since they represent partial solutions which could be considered during a search. This then gives

$$C_s \approx \begin{cases} \sum_{j=0}^{\mu} G(j)/N_{soln} & \text{if } N_{soln} > 0 \\ \sum_{j=0}^{\mu} G(j) & \text{if } N_{soln} = 0 \end{cases} \quad (1)$$

where $G(j)$ is the number of "goods" at level j in the lattice and $N_{soln} = G(\mu)$ is the number of solutions. We should note that this cost measure directly corresponds to search methods that examine all the goods. Often this will examine more states than is really necessary and more sophisticated search methods could eliminate some or all of the redundant search. Provided such methods just remove a fixed fraction of the redundant states, the sharpness of the maximum in the cost described below means that our results on the existence and location of the transition apply to these sophisticated methods as well.

Other cost measures, appropriate for different solution methods, could be defined in a similar way. For example, one could pick an order in which to instantiate the variables and restrict consideration to those sets of assumptions that contain variables in this order. This restriction results in a tree, considerably smaller than the lattice, whose cost can be readily related to that of the lattice as follows. Each node at level j in the lattice specifies values for j variables. In a tree search procedure these could be instantiated in $j!$ different orders and likewise the remaining variables could be instantiated in $(\mu - j)!$ orders. So each node at level j in the lattice therefore participates in $j!(\mu - j)!$ trees. As there are μ variables in total, there are $\mu!$ orders in which all the variables could be instantiated. So the lattice cost measure can be transformed into the tree cost measure by replacing $G(j)$ with $j!(\mu - j)!G(j)/\mu! = G(j)/\binom{\mu}{j}$. This transformation gives a cost measure appropriate for simple backtracking search in which successive variables are instantiated until no compatible extension exists at which point the method backtracks to the last choice. Since our interest lies in the *relative* cost of solving different CSP instances rather than the *absolute* cost and cost measures for the tree and full lattice attain maxima around the same problems (as confirmed empirically below), and the lattice cost is conceptually easier to work with (because we can ignore variable ordering), we use the lattice version.

Although we characterize global parameters of the CSP in terms of the number of states that must be examined in the lattice we certainly do not want to have to construct the lattice explicitly because it can be enormous. Luckily, the lattice can be characterized in every detail by the set of minimal inconsistent partial assignments (which we'll call the "minimized nogoods") which can be obtained directly from the CSP by listing all inconsistent constraint instances and retaining only those that are not a superset of any other. These form a Sperner system (Bollobas 1986) because they satisfy the Sperner requirement that no set is a subset of any other. Note that minimized nogoods and minimal nogoods are slightly different: if there were no solutions there might be many minimized nogoods but there would be a single minimal nogood, namely the empty set. As we shall see, by mapping a CSP into its equivalent set of minimized nogoods we can predict the cost and number of solutions and hence the problem difficulty. This two stage process can be summarized as:

1. CSP → Sperner System, i.e., minimized nogoods
2. Sperner System → Estimate of Difficulty, i.e., C_s

The details of the first step vary depending on the CSP but those of the second do not.

[1] In (Williams & Hogg 1992) we compare this model with one containing all the nogoods and find no qualitative difference although the reduced lattice yields better quantitative results.

An exact specification of the minimized nogoods permits an exact calculation of N_{soln} and C_s. Thus an approximate specification of the minimized nogoods, e.g., involving just their cardinality, m and average size, k, can be expected to permit an approximate calculation of the global parameters. There are both scientific and engineering reasons for wanting to deal with approximate specifications:

1. we want to identify which aspects of the minimized nogoods have the most significant impact upon cost and number of solutions.
2. as a matter of pragmatics, it may be simpler to estimate characteristics of the minimized nogoods, e.g., by sampling, than to compute the set exactly.
3. the specification of real-world CSPs may be inexact or incomplete, so we couldn't, even in principle, obtain a set minimized nogoods. Nevertheless we might still be able to estimate local characteristics of the minimized nogoods and hence make global predictions.

Thus we should prepend "approximate or exact" before the aforementioned steps in our analysis process.

In the remainder of this paper, we walk through an analysis of a familiar CSP to see how these steps pan out in practice, compare our model's predictions against independent data and then discuss how our methodology can be applied to other cases.

CSP → Sperner System

The graph coloring problem consists of a graph, with μ nodes (i.e., variables), b colors (i.e., values) and an average connectivity (i.e., the average number of edges incident on a node) γ. Thus the total number of edges in the graph is $\frac{1}{2}\gamma\mu$ since each edge is incident on two nodes. The task is to find an assignment of colors to nodes such that no two adjacent nodes (i.e., those at either end of an edge) have the same color. Note that this specification is rather coarse; different instances having exactly the same μ and γ can differ markedly in their global performance metrics. We therefore consider an ensemble of graphs and ask what are the values $\langle N_{soln}\rangle$ and $\langle C_s\rangle$ averaged over a family of problems specified by (μ, b, γ).

The condition that no two adjacent nodes may have the same color means that the problem has a set of binary constraints (implying the minimized nogoods are all of size $k = 2$), one for each edge in the graph, each of which induces $b^2 - b$ goods and b distinct nogoods. Hence, the total number of minimized nogoods arising from assigning the same color to linked nodes is $m = \frac{1}{2}\gamma\mu b$. To emphasize the connection between the number of minimized nogoods and the number of variables we write $m = \beta\mu$ with $\beta = \frac{1}{2}\gamma b$.

This completes the mapping between a specific CSP and its minimized nogoods. Next we turn to the question of relating local measures of the minimized nogoods e.g., m and k, to global measures of the lattice e.g., N_{soln} and C_s.

Sperner System → Estimate of Difficulty

$\langle N_{soln}\rangle$ and $\langle C_s\rangle$ depend on $\langle G(j)\rangle$, the expected number of goods at level j in the lattice for $0 \le j \le \mu$. Therefore, we must relate $\langle G(j)\rangle$ to the number, m, and average size, k, of the minimized (problem dependent) nogoods. Note that for graph coloring the minimized nogoods are all at the single level $k = 2$.

Discarding the necessary nogoods, there are $\binom{\mu}{j} b^j$ possible nodes at level j. A node at this level is good if and only if it is not a superset of any of the m minimized nogoods at level k with $0 \le m \le \binom{\mu}{k} b^k$. As each node at level j is above exactly $\binom{j}{k}$ nodes at level k, the number of ways of selecting m minimized nogoods such that a given node at level j is good is just $\binom{\binom{\mu}{k}b^k - \binom{j}{k}}{m}$. Similarly, the total number of ways of selecting m minimized nogoods is $\binom{\binom{\mu}{k}b^k}{m}$. Therefore, assuming the minimized nogoods are independent, the probability, p_j, that a node at level j is a good is given by

$$p_j = \frac{\binom{\binom{\mu}{k}b^k - \binom{j}{k}}{m}}{\binom{\binom{\mu}{k}b^k}{m}} \tag{2}$$

Hence, the expected number of goods at level j is just

$$\langle G(j)\rangle = \binom{\mu}{j} b^j p_j \tag{3}$$

and the expected number of solutions is then $\langle N_{soln}\rangle = \langle G(\mu)\rangle$. For $\langle C_s\rangle$ we use the observation (Williams & Hogg 1992) that the probability of having solutions, $P_{soln} \equiv \Pr(N_{soln} > 0)$, as a function of β, approaches a sharp step as $\mu \to \infty$. This allows us to replace the conditions in Eq. 1 that $N_{soln} > 0$ with $P_{soln} = 1$ and $N_{soln} = 0$ with $P_{soln} = 0$. Applying a mean field approximation[2], the expected cost to first solution or to failure is then

$$\langle C_s\rangle \approx \begin{cases} \sum_{j=0}^{\mu} \langle G(j)\rangle / \langle N_{soln}\rangle & \text{if } P_{soln} = 1 \\ \sum_{j=0}^{\mu} \langle G(j)\rangle & \text{if } P_{soln} = 0 \end{cases} \tag{4}$$

[2]This approximation, widely used in statistical physics, has the form $\langle f(x)\rangle \approx f(\langle x\rangle)$. Ultimately its accuracy must be tested by comparing model predictions to real data.

Probability to Have a Solution

$\langle C_s \rangle$ can now be calculated approximately once the functional form for P_{soln} is known. Unfortunately, an exact derivation of P_{soln} is complex. However, since it approaches a step function, its behavior is determined by the critical value $\beta = \beta_{crit}^{step}$ at which it changes from being near 1 to near 0, i.e., the location of the step. An approximation to the exact position of the step can be obtained by assuming that the existence of each solution is independent of the others. Under this assumption, the probability to have no solutions is just the product of the probabilities that each state at the solution level is not a solution. Thus we have $P_{soln} \approx 1-(1-p_\mu)^{b^\mu}$ where p_μ, given in Eq. 2, is the probability that a state at the solution level is in fact a solution. Taking logarithms and recognizing that at the transition between having solutions and not having solutions p_μ must be small gives us $\ln(1-P_{soln}) \sim -b^\mu p_\mu$ which can be written as $P_{soln} \sim 1-e^{-\langle N_{soln} \rangle}$ because the expected number of solutions is just $\langle N_{soln} \rangle = \langle G(\mu) \rangle$. We see immediately that if $\langle N_{soln} \rangle \to \infty$, $P_{soln} \to 1$ and if $\langle N_{soln} \rangle \to 0$, $P_{soln} \to 0$.

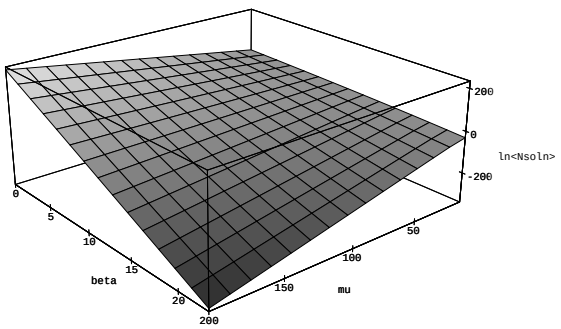

Fig. 2. Behavior of $\ln \langle N_{soln} \rangle$ as a function of β and μ for $k = 2$; $b = 3$.

The behavior of $\ln \langle N_{soln} \rangle$ is shown in Fig. 2. This shows there is a transition from having exponentially many to having exponentially few solutions at some critical value of $\beta = \beta_{crit}$. This value can be determined by using Stirling's approximation to simplify the binomial coefficients from Eq. 3 appearing in $\ln \langle N_{soln} \rangle = \ln \langle G(\mu) \rangle$ to give (Williams & Hogg 1992) $\ln \langle N_{soln} \rangle \sim \mu \left[\ln(b) + \beta \ln\left(1-b^{-k}\right) \right]$. The transition from exponentially many to exponentially few solutions occurs at the point when this leading order term vanishes. Hence,

$$\beta_{crit} = -\frac{\ln b}{\ln\left(1-b^{-k}\right)} \tag{5}$$

Thus, in this approximation, the probability profile, $P_{soln}(\beta)$ does indeed approach a sharp step as $\mu \to \infty$, with the step located at $\beta_{crit}^{step} \approx \beta_{crit}$. Substituting for $\langle N_{soln} \rangle$ in the approximation for P_{soln} yields $P_{soln} \approx 1-\exp\left(-\exp\left(\mu\left[\ln(b) + \beta \ln\left(1-b^{-k}\right)\right]\right)\right)$, a sketch of which is shown in Fig. 3. Notice how the sharpness of the step increases with increasing μ suggesting the above approximations are accurate for large CSPs.

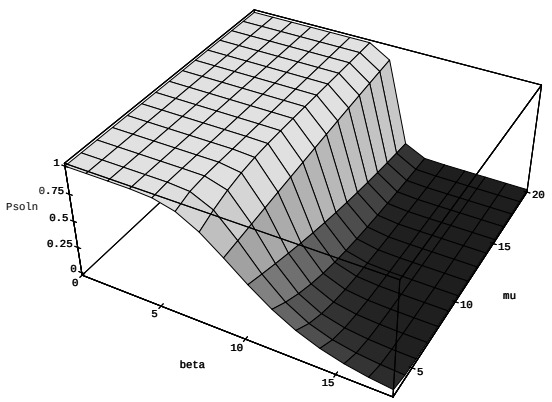

Fig. 3. Approximation to P_{soln} as a function of β for $k = 2$ and $b = 3$.

Cost to First Solution or to Failure (i.e., Difficulty)

At this point we can calculate the average number of goods at any level (including the solution level) and the critical point at which there is a switch from having solutions to not having any. Thus we can estimate $\langle C_s \rangle$ by:

$$\langle C_s \rangle \approx \begin{cases} \sum_{j=0}^{\mu} \langle G(j) \rangle / \langle N_{soln} \rangle & \beta < \beta_{crit} \\ \sum_{j=0}^{\mu} \langle G(j) \rangle & \beta \geq \beta_{crit} \end{cases} \tag{6}$$

The behavior of $\langle C_s \rangle$ versus β (a rescaling of the number of minimized nogoods) is shown in Fig. 4. It shows that the cost attains a maximum at β_{crit} and that this peak become increasingly sharp as $\mu \to \infty$.

Thus a slice at constant μ shows what happens to the computational cost in solving a CSP as we sweep across a range of CSPs having progressively more minimized nogoods. Notice that well below the critical point there are so few minimized nogoods that most states at level μ are solutions so the cost is low. Conversely, well above the critical point there are so many minimized nogoods that most futile paths are truncated early and so the cost is low in this regime too.

Another property of this transition is that the variance in cost, $\langle C_s^2 \rangle - \langle C_s \rangle^2$, is a maximum in the vicinity of the phase transition region, a property that is quite typical of phase transition phenomena (Cheeseman, Kanefsky & Taylor 1991, Williams & Hogg 1991, Williams & Hogg 1992).

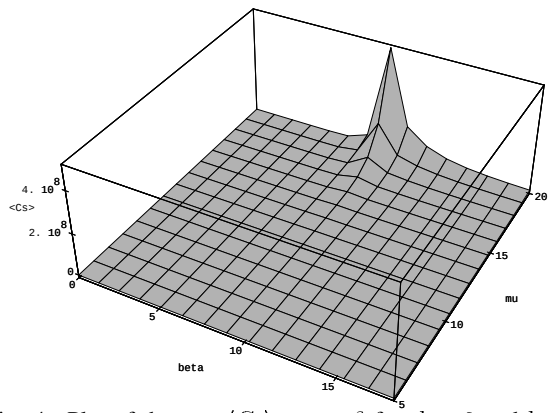

Fig. 4. Plot of the cost $\langle C_s \rangle$ versus β for $k = 2$ and $b = 3$.

#Nodes	#Colors	γ_{crit}^{theory}	γ_{crit}^{expt}
81	3	6.2	5.4 ± 0.2
144	3	6.2	5.4 ± 0.2
144	4	10.7	9.2 ± 0.7

Table 1. Comparison of theory and experiment. The experimental values were obtained from Fig. 3 of (Cheeseman, Kanefsky & Taylor).

We therefore have a major result of this paper: as the number of minimized nogoods increases the cost per solution or to failure displays a sudden and dramatic rise at a critical number of minimized nogoods, $m_{crit} = \beta_{crit}\mu$. Problems in this region are therefore significantly harder than others. This high cost is fundamentally due to increasing pruning ability of minimized nogoods with level in the lattice: near the transition point the nogoods have greatly pruned states at the solution level (resulting in few solutions) but still leave many goods at lower levels near the bulge (resulting in many partial solutions and a relatively high cost).

Experimental Confirmation

So much for the theory. But how good is it? To evaluate the accuracy of the above prediction of where problems become hard we compare it with the behavior of actual constraint satisfaction problems. Fortunately, Cheeseman *et al.* have collected data on a range of constraint satisfaction problems (Cheeseman, Kanefsky & Taylor 1991), so we take the pragmatic step of comparing our theory against their data.

For the graph coloring problem we found $\beta = \frac{1}{2}\gamma b$. By substituting β_{crit} for β we can predict the critical connectivity at which the phase transition takes place, γ_{crit}^{theory}, and compare it against those values Cheeseman *et al.* actually measured.

Our model both predicts the qualitative existence of a phase transition at a critical connectivity (via β_{crit}) and estimates the quantitative value of the transition point to within about 15%. Scaling is even better: as b changes from 3 to 4 this model predicts the transition point increases by a factor of 1.73, compared to 1.70 for the experimental data, a 2% difference.

The outstanding discrepancy is due to a mixture of the mathematical approximations made in the derivation, the absence of explicit correlations among the choices of nogoods in our model, the fact that Cheeseman *et al.* used "reduced" graphs rather than random ones to eliminate trivial cases, the fact that their search algorithm was heuristic whereas ours assumed a complete search strategy, and statistical error in the samples they obtained.

Discussion

We have proposed an analysis technique for predicting where, in a space of problem instances, the hardest problems lie and where the fluctuations in difficulty are greatest. The key insight that made this feasible was to shift emphasis from modelling sophisticated algorithms directly to modelling a search space which captures their principal effects. Such a strategy can be generalized to any problem whose search space can be mapped onto a lattice, e.g., satisficing searches (Mackworth 1987) and version spaces (Mitchell 1982), provided one re-interprets the nodes and links appropriately. In general, the minimized nogoods may not all be confined to a single level or they may overlap more or less than the amount induced by our assuming their independence. In (Williams and Hogg 1992) we explore the consequences of such embellishments and find no significant qualitative changes. However, for a fixed average, but increasing deviation in, the size of the minimized nogoods, the phase transition point is pushed to lower β and the average cost at transition is decreased. Similarly, allowing minimized nogoods to overlap more or less than random moves the phase transition point to higher or lower β respectively with a concomitant decrease or increase in the cost at transition.

A further application is to exploit our cost formula to estimate the value of constraint preprocessing (Mackworth 1987). Although the number of solutions is kept fixed, preprocessing has the effect of decreasing the domain size from $b \rightarrow b'$ causing a corresponding change in $\beta \rightarrow \beta'$ given by the solution to $N_{soln}(\mu, k; b, \beta) = N_{soln}(\mu, k; b', \beta')$ which in turn allows the change in cost to be computed, $C_s(\mu, k, b, \beta) \rightarrow C_s(\mu, k, b', \beta')$. We find that the rate of the decrease in cost is highest for problems right at the phase transition suggesting preprocessing has the most dramatic effect in this regime.

In addition, related work suggests cooperative methods excel in regions of high variance. As the variance is

highest at the phase transition, the proximity of a problem instance to the phase transition can suggest whether or not cooperative methods should be used (Clearwater, Huberman & Hogg 1991).

The closest reported work is that of Provan (Provan 1987a, Provan 1987b). His model is different from ours in that it assumed a detailed specification of the minimized nogoods which included the number of sets by size together with their intra-level and inter-level overlaps. We contend that as A.I. systems scale up such details become progressively less important, in terms of understanding global behavior, and perhaps harder to obtain anyway. In the limit of large problems, the order parameters we have discussed (cardinality m and average size k) appear to be adequate for graph coloring. For other types of CSP it might be necessary to calculate other local characteristics of the minimized nogoods e.g. the average pairwise overlap or higher moments of their size, in order to make sufficiently accurate predictions. One can never know this until the analysis is complete and the predictions compared against real data. But then, one should not accept any theory until it has passed a few experimental tests. Experience with modelling other kinds of A.I. systems (Huberman & Hogg 1987, Williams & Hogg 1991) leads us to believe this phenomenon is quite common; the analysis of relatively small A.I. systems, for which the details most definitely do matter, do not always supply the right intuitions about what happens in the limit of large problems. Moreover, the existence of the phase transition in the vicinity of the hardest problems would not have been apparent for small systems as the transition region is somewhat smeared out.

Conclusions

The results reported are very encouraging considering how pared down our model is, involving only two order parameters, the number and size of the minimized nogoods. This demonstrates that, at least in some cases, a complete specification of the Sperner system is not a prerequisite to predicting performance.

However, a more surprising result was that search spaces as different as the assumption lattice described here and the tree structure used in backtracking yield such close predictions as to where problems become hard. This suggests that the phase transition phenomenon is quite generic across many types of search space and predicting where the cost measure blows up in one space can perhaps suggest where it will blow up in another. This allows us to finesse the need to do *algorithmic* complexity analyses by essentially doing a *problem* complexity analysis on the lattice. This raises the intriguing possibility that we have stumbled onto a new kind of complexity analysis; one that is generic regardless of the clever search algorithm used.

These observations bode well for the future utility of this kind of analysis applied to complex A.I. systems.

References

Bollobas, B. 1986. *Combinatorics*, Cambridge University Press.

Cheeseman, P.; Kanefsky, B.; and Taylor, W. M. 1991. Where the Really Hard Problems Are. In Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, 331–337, Morgan Kaufmann.

Clearwater, S.; Huberman, B. A.; and Hogg, T. 1991. Cooperative Solution of Constraint Satisfaction Problems. *Science*, 254:1181–1183.

de Kleer, J. 1986. An Assumption Based TMS, *Artificial Intelligence*, 28:127–162.

Franco and Paull 1983. Probabilistic Analysis of the Davis Putman Procedure for Solving Satisfiability Problems, Discrete Applied Mathematics 5:77–87.

Huberman, B.A. and Hogg, T. 1987. Phase Transitions in Artificial Intelligence Systems, *Artificial Intelligence*, 33:155–171.

Mackworth, A. K. 1987. Constraint Satisfaction. In S. Shapiro and D. Eckroth (eds.) *Encyclopedia of Artificial Intelligence*, 205–211. John Wiley and Sons.

Mitchell, T. M. 1982. Generalization as Search. *Artificial Intelligence*, 18:203–226.

Papadimitriou, C. H. 1991. On Selecting a Satisfying Truth Assignment. In Proceedings of the 32nd Annual Symposium of Computer Science, IEEE Computer Society.

Provan, G. 1987a. Efficiency Analysis of Multiple Context TMSs in Scene Recognition. Tech. Rep. OU-RRG-87–9, Robotics Research Group, Dept. of Engineering Science, University of Oxford.

Provan, G. 1987b. Efficiency Analysis of Multiple Context TMSs in Scene Representation. In Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI87), 173–177: Morgan Kaufmann.

Purdom, P. W. 1983. Search Rearrangement Backtracking and Polynomial Average Time, *Artificial Intelligence*, 21:117–133

Williams, C. P. and Hogg, T. 1991. Universality of Phase Transition Phenomena in A.I. Systems. Tech. Rep. SSL-91–04, Xerox Palo Alto Research Center, Palo Alto, California.

Williams, C. P. and Hogg, T. 1992. Exploiting the Deep Structure of Constraint Problems. Tech. Rep. SSL-92–24, Xerox Palo Alto Research Center, Palo Alto, California.