A New Look at the Easy-Hard-Easy Pattern of Combinatorial Search Difficulty*

Dorothy L. Mammen[1] and Tad Hogg[2]

[1]Department of Computer Science
University of Massachusetts
Amherst, MA 01003, U.S.A.
mammen@cs.umass.edu

[2]Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304, U.S.A.
hogg@parc.xerox.com

University of Massachusetts Technical Report 96-22

March 1996

Abstract

The easy-hard-easy pattern in the difficulty of combinatorial search problems as constraints are added has been explained as due to a competition between the decrease in number of solutions and increased pruning. We test the generality of this explanation by examining one of its predictions: if the number of solutions is held fixed by the choice of problems, then increased pruning should lead to a monotonic decrease in search cost. Instead, we find the easy-hard-easy pattern in median search cost even when the number of solutions is held constant, for some search methods. This generalizes previous observations of this pattern and shows that the existing theory does not explain the full range of the peak in search cost. In these cases the pattern appears to be due to changes in the size of the minimal unsolvable subproblems, an aspect of the transition not previously reported.

*Much of this research was carried out while the first author was a summer intern at Xerox Palo Alto Research Center. This research was also partially supported by the National Science Foundation under Grant No. IRI-9321324 to Victor R. Lesser. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

1 Introduction

Recently, many authors have shown that the solution cost for various kinds of combinatorial search problems follows a pattern of easy-hard-easy as a function of how tightly constrained the problems are. For example, this pattern appears for graph coloring as a function of the average graph connectivity [Cheeseman *et al.*, 1991, Hogg and Williams, 1994], for propositional satisfiability (SAT) as a function of the ratio of number of clauses to number of variables [Cheeseman *et al.*, 1991, Mitchell *et al.*, 1992, Crawford and Auton, 1993, Gent and Walsh, 1994b], and for constraint satisfaction problems (CSPs) as a function of the number of nogoods [Williams and Hogg, 1994] and constraint tightness [Smith, 1994, Prosser, 1996].

This regularity raises the possibility of determining, prior to search, the likely difficulty of problems. Unfortunately, this is not yet possible because of the high variance associated with the observations. This is compounded by the fact that a single problem can be viewed as belonging to a variety of problem classes, each with somewhat different transition points. Thus one important direction for improvement is to investigate whether there are simple additional parameters that can reduce this variance and allow predictions with higher confidence.

One approach to this question is based on the explanation of the easy-hard-easy pattern as a competition between changes in the number of solutions and pruning of unproductive search paths as a function of some measure of the degree to which the problems are constrained. In particular this predicts that problems with many solutions tend to be easier, on average, than those with fewer for a given number of constraints. Thus, at least one aspect of the high variance in search cost appears to be due to the variance in number of solutions in the problems of a fixed degree of constraint. This observation has motivated the introduction of additional parameters describing problem structure based on a more precise specification of the number of solutions [Hogg, 1996].

In this paper we investigate the generality of this explanation by examining problems for which the number of solutions is restricted, including cases where the number is specified exactly to be either zero or one. If the peak in search cost in fact arises generally from a competition between changes in the number of solutions and pruning, cases with a fixed number of solutions should not show a peak. In fact, we find that a peak continues to appear in these cases for some sophisticated search algorithms while it fails to appear in other cases. This calls into question the generality of the explanation based on number of solutions, and also suggests that a search for additional problem structure parameters based solely on reducing the variance in the number of solutions is not likely to be sufficient to accurately predict search cost.

In the next section we describe some classes of search problems. We then review the pattern of search behavior and the current theoretical explanation for it. In the following section we uncover some limitations of this explanation by examining problems with some specification on their number of solutions. This shows the easy-hard-easy pattern is a more general phenomenon than suggested by current explanations. We then discuss some of the implications of these observations and suggestions for obtaining a better understanding and predictability for hard search problems.

2 Some Classes of Search Problems

In common with many previous studies of the transition phenomenon, we use random CSPs and graph coloring as example classes of search problems. This section describes how the problems were generated and searched.

2.1 Random CSPs

The constraint satisfaction problems we used in most of our experiments consist of 10 variables with 3 possible values for each one. The constraints are specified by a number of binary nogoods, i.e., assignments to a pair of variables that are considered to be inconsistent. The search problem is then to find a consistent complete assignment, i.e., a value for each of the 10 variables that does not include any of the inconsistent pairs.

We generated problems in a number of ways to fully sample the range of behaviors. In the first method ("generate-select") we generate CSPs by randomly selecting the specified number of binary nogoods. From the problems generated in this way, we consider only those that satisfy particular requirements on the number of solutions. In the general case, all such problems are included. For solvable problems, only those with a solution are included. Finally, for problems with a fixed number of solutions, only those problems with a specified number of solutions are included.

This random generation method gives a simple, uniform selection from the various problem classes. However, it can also be very inefficient in generating problems. For instance, with few nogoods, most randomly generated problems are solvable, hence requiring a large number of random trials to obtain even a few unsolvable cases.

To address this problem, we also used more efficient ("hill-climbing") methods. Specifically, for generating solvable problems with many nogoods, starting with a randomly generated unsolvable problem, we removed constraints at random until the problem became solvable, then restored the number of constraints removed with constraints chosen randomly, but with the requirement that the problem not become unsolvable again.

For generating unsolvable problems with few nogoods, the hill-climbing method started with a randomly generated solvable problem, removed the constraint that constrained the problem the least (the one whose removal increased the number of solutions the least), and added a randomly chosen constraint that resulted in a problem with fewer solutions than the problem had before the constraint removal. If, having removed one constraint, no other constraint could decrease the number of solutions, the constraint that increased the number of solutions the least was chosen – a slightly backwards step. To speed this process up, we checked only one third of the possible constraints before giving up, choosing the one that increased the number of solutions the least, and starting another iteration.

Other methods for generating problems with specified requirements on the number of solutions have also been studied. One popular method for solvable problems is to randomly select an assignment to all of the variables (a pre-specified solution) and then, during the random selection of nogoods, avoid any that are inconsistent with this pre-specified solution. This tends to emphasize problems with many solutions and results in instances that are somewhat easier than uniform random selection. [Cha and Iwama, 1995] have also used the approach of generating problems with specific attributes, for SAT problems, using the AIM generators developed by [Asahiro *et al.*, 1993].

We solved these problems using dynamic backtracking [Ginsberg, 1993] in most cases. For comparison, we also did some searches with simple chronological backtrack instead. The search cost is measured as the number of nodes explored.

2.2 Graph Coloring

We also experimented with the 3-coloring problem. This constraint satisfaction problem consists of a graph and the requirement to assign each node one of three colors so that no pair of nodes linked by an edge have the same color. Each edge in the graph defines some binary nogoods for the problem, namely all pairs of assignments giving the same color to the two nodes connected by the edge. Thus each edge in the graph gives three binary nogoods. A convenient measure of the number of constraints is γ, the connectivity or average degree of the nodes in the graph. This is equal to twice the number of edges in the graph divided by the number of nodes, because each edge is incident on two nodes. For the 100-node graphs we studied, the number of binary nogoods is given by 150γ.

In this case, we used a simple chronological backtrack search in combination with the Brelaz heuristic for variable and value ordering [Johnson *et al.*, 1991]. This heuristic assigns the most constrained nodes first (i.e., those with the most distinctly colored neighbors), breaking ties by choosing nodes with the most uncolored neighbors, and with any remaining ties broken randomly. The colors are considered in a fixed ordering for all of the nodes in the search. As a simple optimization, the search never changes the colors selected for the first two nodes. Any such changes would amount to unnecessarily repeating the search with a permutation of the colors for unsolvable cases. Search cost is measured by the number of nodes explored.

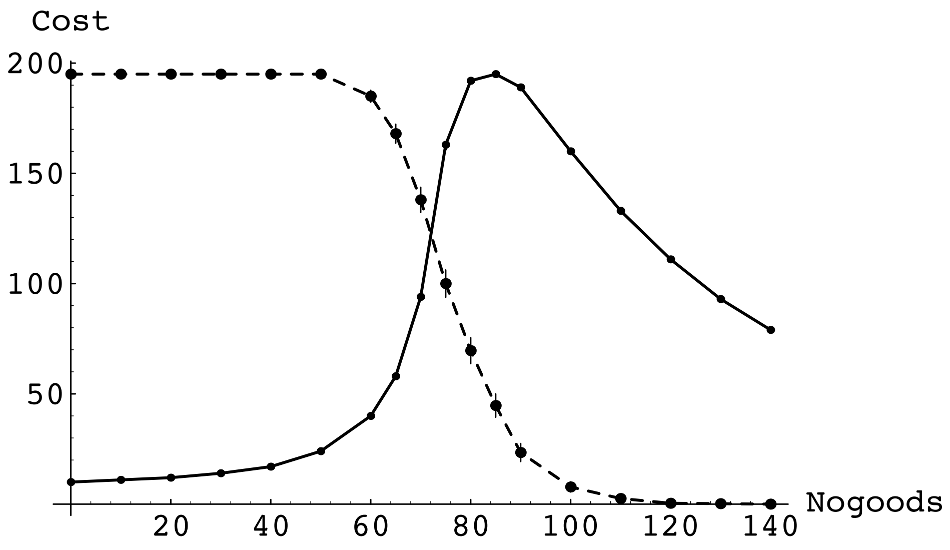

Figure 1: Typical transition pattern. Median solution cost for dynamic backtracking (solid line) and probability of a solution (dotted line) as a function of number of nogoods. Each point represents 1000 problems, each solved 100 times. Error bars showing 95% confidence intervals are included, but are in some cases smaller than the size of the plotted points.

3 The Easy-Hard-Easy Pattern

In this section, we present an example of the how search cost varies with the tightness of constraints for a class of problems, and describe how this behavior can be understood in terms of changes in the structure of the problems, independent of particular search algorithms. This review and summary of previous studies of the transition then forms a basis for comparison with the new results presented in subsequent sections.

3.1 An Example

Figure 1 shows a typical example of the easy-hard-easy pattern as a function of the constrainedness of the problem. Problems with few or many constraints tend to be easy to solve while those with an intermediate number are more difficult. The fraction of solvable problems is also shown in Figure 1, scaled from 1.0 on the left to 0.0 on the right. This illustrates that the hard problems are concentrated in the so-called "mushy region" [Smith and Dyer, 1996] where the probability of a solution is changing from 1.0 to 0.0. In particular, the peak in search cost is near the "crossover point," the point at which half the problems are solvable and half unsolvable. For this problem class, the crossover point occurs at just over 75 binary nogoods, and the peak in dynamic backtracking solution cost occurs at about 85 binary nogoods.

In all of our results in this paper, we include 95% confidence intervals. The 95% confidence intervals for the estimate of the median obtained from our samples is given approximately [Snedecor and Cochran, 1967, p. 124] by the percentiles $50 \pm 100/\sqrt{N}$ of the data, where N is the number of samples. The 95% confidence intervals for the estimate of fractions is given approximately [Snedecor and Cochran, 1967, p. 210] by $f \pm 2\sqrt{f(1-f)/N}$, where f is the estimated value of the fraction. In many cases in this paper, there are sufficient samples to make this interval smaller than the size of the plotted points.

A key point from examples such as this is that the difficult instances within a class of search problems tend to be concentrated near a particular value of the constraint tightness (here measured by the number of binary nogoods). Because this behavior is seen for a variety of search methods, it indicates this concentration does not depend much on the details of the search algorithm. Instead, it appears to be associated with a change in the properties of the problems themselves, namely their solvability.

3.2 An Explanation

These observations raise a number of questions, such as why a peak in search cost exists, why the peak occurs near the transition from mostly solvable to mostly unsolvable problems and is thus independent of the particular search algorithm, and why this behavior is seen for a large variety of constraint satisfaction problems.

The existing explanation for the concentration of hard problems relies on a competition between changes in the number of solutions and the amount of pruning provided by the problem constraints[Williams and Hogg, 1994]. With few constraints, there are many solutions so the search is usually easy. As constraints are added the number of solutions drops rapidly, making problems harder. But the new constraints also increase the pruning of unproductive search choices, tending to make search easier. When there are few constraints, the decrease in the number of solutions overwhelms the increase in pruning, giving harder problems on average. Eventually the last solution is eliminated and all that remains is the increased pruning from additional constraints, leading to easier problems. Thus the phase transition, the point at which there is a precipitous change from solvability to unsolvability, more or less coincides with the peak in solution cost. All these effects become more pronounced as larger problems are considered, leading to sharper peaks and more abrupt transitions. This qualitative description explains many features of the observed behavior. This pruning explanation was also offered by [Cheeseman *et al.*, 1991] with respect to finding Hamiltonian circuits in highly constrained problems.

This explanation can also be used to obtain a quantitative understanding of the behavior. For instance, the location of the transition region can be understood by an approximate theory predicting that the cost peak occurs when the expected number of solutions equals one [Smith and Dyer, 1996, Williams and Hogg, 1994]. In our example there are 3^{10} possible assignments to the 10 variables in the problem. There are $\binom{10}{2} 3^2 = 405$ possible binary nogoods for the problem, which counts the number of ways to select a pair of variables and the different assignments for that pair. A given complete assignment for the 10 variables will be a solution provided each of the selected binary nogoods does not use the same assignment for its pair of variables as in the given complete assignment. This leaves $\binom{10}{2}(3^2 - 1) = 360$ possible choices for the binary nogoods. Thus the expected number of solutions is given by

$$3^{10} \times \frac{\binom{360}{m}}{\binom{405}{m}}$$

for problems with m randomly selected binary nogoods. This expression equals one at $m = 82.9$, fairly close to the location of the observed cost peak. Furthermore, because the expected number of solutions grows exponentially with the number of variables when m is smaller than this threshold value and decreases exponentially to zero when m is larger, the range of m values over which the expected number of solutions is near one rapidly decreases as variables are added. This accounts for the observed sharpening of the transition for larger problems.

A further quantitative success of relating the search cost peak to transition phenomena is the evaluation of scaling behavior of the transition and search cost peak [Kirkpatrick and Selman, 1994, Gent *et al.*, 1995].

4 Search Difficulty and Solvability

In this section we take a closer look at the behavior of the search cost, specifically, by examining how the behavior depends on whether the problem has a solution and, if so, the number of solutions.

4.1 Search Behavior

Figure 2 shows the median dynamic backtracking solution cost for solvable and unsolvable random CSPs generated as described above. We generated 1000 solvable and 1000 unsolvable problems with 10 variables and 3 values. Solvable problems up to 130 binary

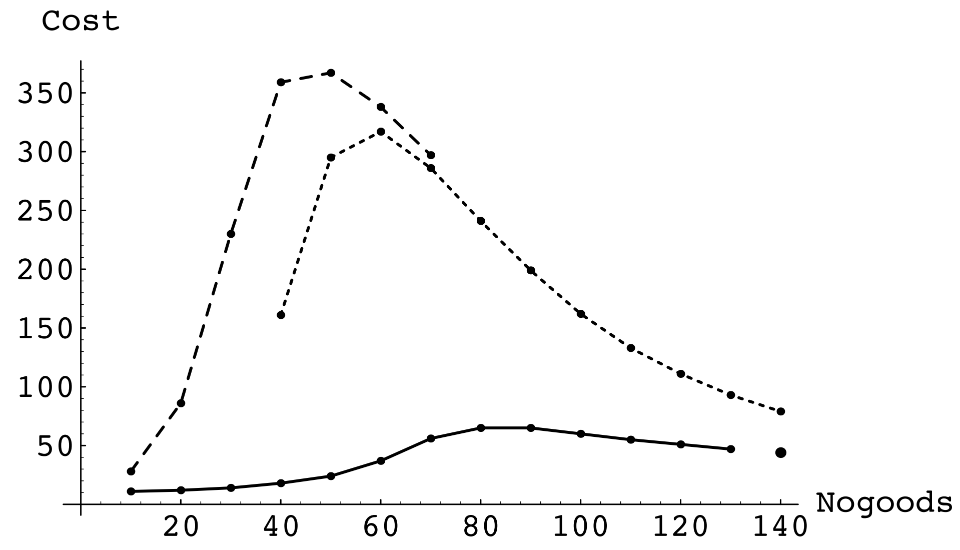

Figure 2: Solution cost for solvable problems (solid line), and unsolvable problems generated with the "generate-select" (dotted line) and "hill-climbing" (dashed line) methods, solved using dynamic backtracking. The isolated point at 140 nogoods is for solvable problems generated with the "hill-climbing" method. Each point is the median of 1000 problems each solved 100 times, except for the unsolvable problems generated by "generate-select" at 40 nogoods, which is based on 248 problems. Error bars showing 95% confidence intervals are included, but are smaller than the size of the plotted points.

nogoods were generated by the generate-select method. At 140 nogoods, generating solvable problems this way became quite difficult, so we used the hill-climbing method. We generated unsolvable problems with 40 or more nogoods by the generate-select method. For fewer nogoods, however, this method became infeasible, so we used the hill-climbing method. To show how the different generation methods affect search cost, a few points were generated both ways, resulting in the overlap of curves shown in the plot.

This figure clearly shows the easy-hard-easy pattern of solution cost for both solvable and unsolvable problems. The two methods of generating unsolvable problems give distinct curves: the unsolvable problems generated by the "hill-climbing" method are harder than those generated by the "generate-select" method.

Another example with the same behavior is shown in Figure 3 for the median search cost for instances of 3-coloring of random graphs. In contrast to Figure 2, the solvable and unsolvable cases have similar median search costs near the peaks. This is because, as described above, the graph coloring searches for unsolvable cases used the symmetry with respect to permutations of the colors to avoid unnecessary search. Without this

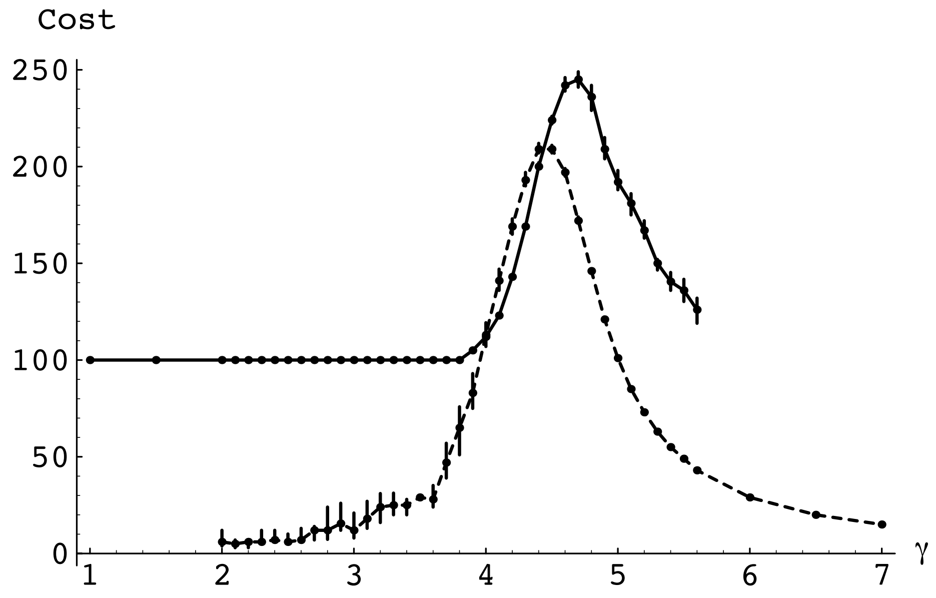

Figure 3: Median solution cost for 3-coloring random graphs with 100 nodes as a function of connectivity γ using backtrack search with the Brelaz heuristic. The solid and dashed curves correspond to solvable and unsolvable cases respectively. These results started with 100,000 random graphs at each value of γ, and additional samples were generated at the extremes to produce at least 100 samples for each point. For random graphs, the crossover from mostly solvable to mostly unsolvable occurs around a connectivity of 4.5. Error bars showing 95% confidence intervals are included.

optimization, the costs for unsolvable cases would be six times greater than the values shown in the figure. Similar peaks are seen for other classes of graphs, such as connected ones, although at somewhat different values of γ.

These data show that both random CSPs and graph coloring problems exhibit an easy-hard-easy pattern for solvable and unsolvable problems considered separately.

4.2 Solvable Problems

How does the existence of a peak for solvable problems fit with the explanation given above? Certainly an explanation based on a transition from solvable to unsolvable problems cannot apply directly to the class of solvable problems. However, the competition between increased pruning and decreased number of solutions still applies. As shown in Figure 4, the number of solutions for solvable random CSPs at first decreases rapidly as constraints are added but then nears its minimum value of one, giving a slower decrease. Except for the change in minimum value from 0 to 1 solution, this behavior for the number of solutions is qualitatively similar to that for the general case including both solvable

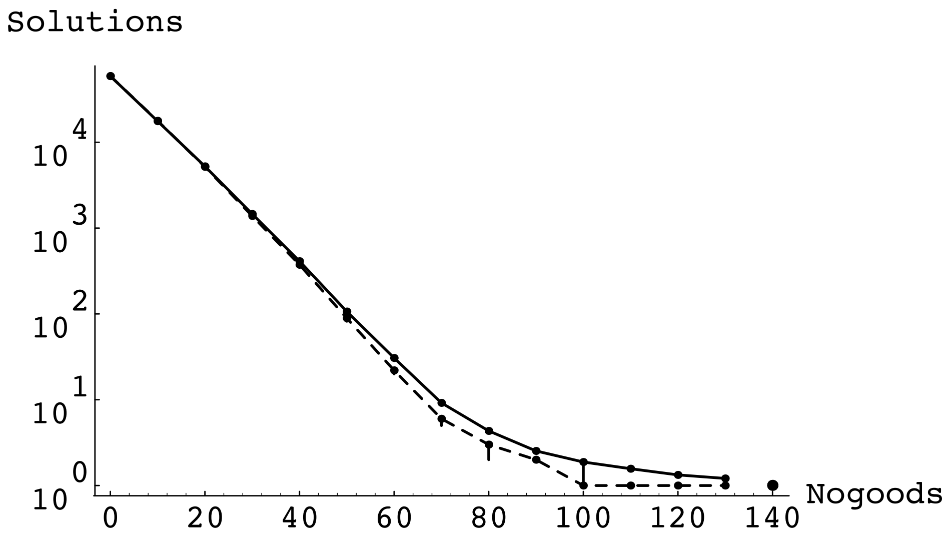

Figure 4: Mean (solid) and median (dashed) number of solutions on a log scale as a function of the number of binary nogoods, for solvable problems with 10 variables, 3 values each, based on 1000 problems at each multiple of 10 binary nogoods. At 0 nogoods there are $3^{10} = 59049$ solutions. These were generated by the "generate-select" method, except for 140 nogoods where the "hill-climbing" method was used (mean and median on top of each other). Error bars showing 95% confidence intervals are included.

and unsolvable problems. The additional constraints continue to increase the pruning of unproductive search paths. Thus the explanation given above continues to apply but now predicts the peak will be at the point where solutions can drop no further (i.e., one solution) rather than becoming unsolvable (i.e., zero solutions).

That this is in fact correct is shown in Figure 5 for random CSPs. The second to last solution disappears, on average, between 90 and 100 nogoods: the median number of solutions has dropped to 2 by 90 nogoods, and to 1 by 100 nogoods (Figure 4). The peak in solution cost for solvable problems is slightly lower than this, at between 80 and 90 nogoods, close to the crossover point of Figure 5 where half the solvable problems have only one solution. However, comparing with Figure 1 shows the disappearance of the second to last solution is more gradual than the change in solvability for general problems. An open question is whether this becomes more abrupt for larger solvable problems as happens with the transition in solvability for general problems.

Thus the location of the solution cost peak is consistent with the explanation based on the competition between the number of solutions and pruning. The association with the disappearance of the second to last solution for solvable problems has not been previously

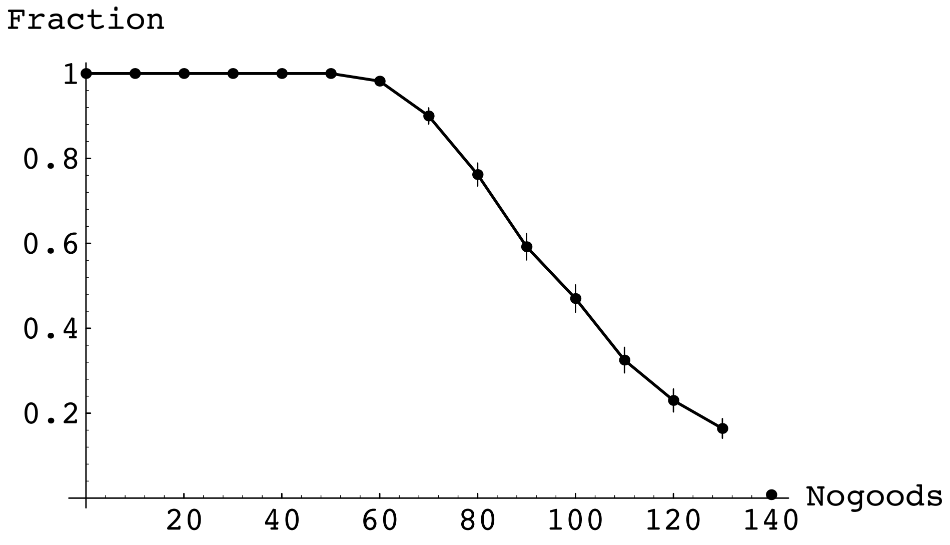

Figure 5: Fraction of problems with at least two solutions as a function of number of nogoods based on 1000 solvable problems at each point. All points except at 140 nogoods were created by the "generate-select" method, while the point at 140 is from problems generated by the "hill-climbing" method. Error bars showing 95% confidence intervals are included.

noted, which left open the question as to why a class of solvable problems should have a peak in search cost at the point where a different class, i.e., containing both solvable and unsolvable problems, has a transition in solvability. This is especially so since other changes in the definition of the class of problems generally result in some shift in the location of the transition point.

An additional generalization of these comments generates a prediction for the case of graph coloring. For this problem, a permutation of any consistent coloring is also a solution to the problem. Thus, for instance, a 3-colorable graph will usually have a minimum of six solutions and we would expect the peak for colorable graphs to be associated with the point at which any solutions beyond those required by this symmetry disappear.

The peak in search cost for solvable problems has also been seen extensively in studies of local-repair search methods and for problems generated with a pre-specified solution [Yugami *et al.*, 1994, Kask and Dechter, 1995, Williams and Hogg, 1994]. These search methods start with some assignment to all of the variables in the problem and then attempt to adjust them until a solution is found. Generally, such methods are not systematic searches: they can never determine that a problem has no solution. Thus empirical studies of these methods are restricted to consider solvable problems and incidentally

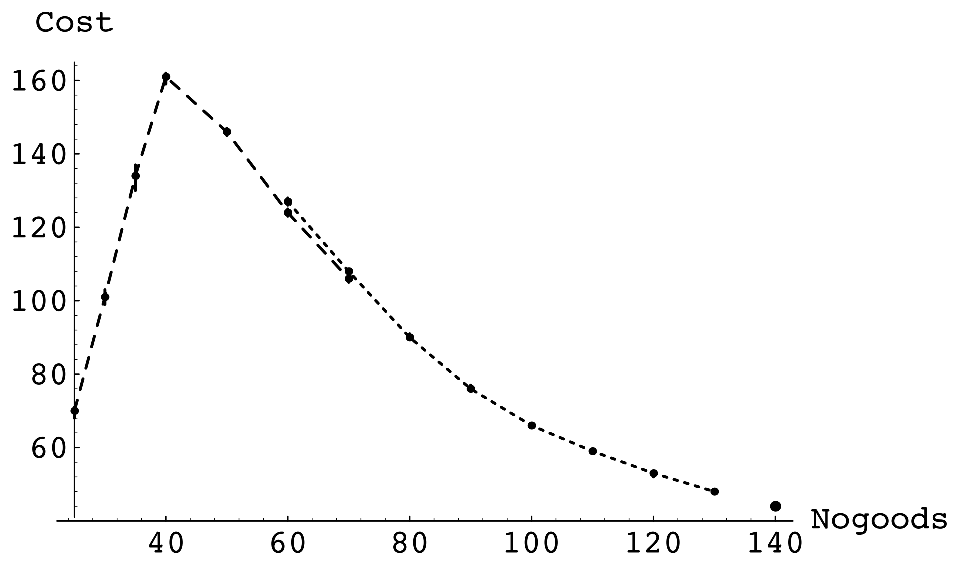

Figure 6: Solution cost as a function of number of nogoods for problems of 10 variables, 3 values each, with exactly one solution, generated using the "generate-select" method (dotted line), and by hill-climbing down to one solution starting from solvable problems with many solutions produced using "generate-select" (dashed line), solved using dynamic backtracking. Each point is the median of 1000 problems each solved 100 times, except for hill-climbing generated problems at 25, 30 and 35 nogoods, of which there are 100. Error bars showing 95% confidence intervals are included.

provide a useful examination of the properties of solvable problems.

Furthermore, a study of satisfiability problems with backtracking search is consistent with a peak in cost for solvable problems [Mitchell *et al.*, 1992], but there were insufficient highly constrained solvable problems to make a definite conclusion for the behavior with many constraints.

4.3 Problems With a Fixed Number of Solutions

A more interesting case is the behavior of the problems with no solutions shown in Figures 2 and 3. As a further example, Figure 6 shows the solution cost for problems with exactly one solution. This also shows a peak. These observations on problems with zero or one solution show that even with the number of solutions held constant, problems exhibit an easy-hard-easy pattern of solution cost.

According to the explanation of the transition, if the number of solutions is held constant then the increase in pruning will be the only factor, giving rise to a monotonic decrease in search cost as constraints are added. Instead, we see in Figures 2 and 6, that

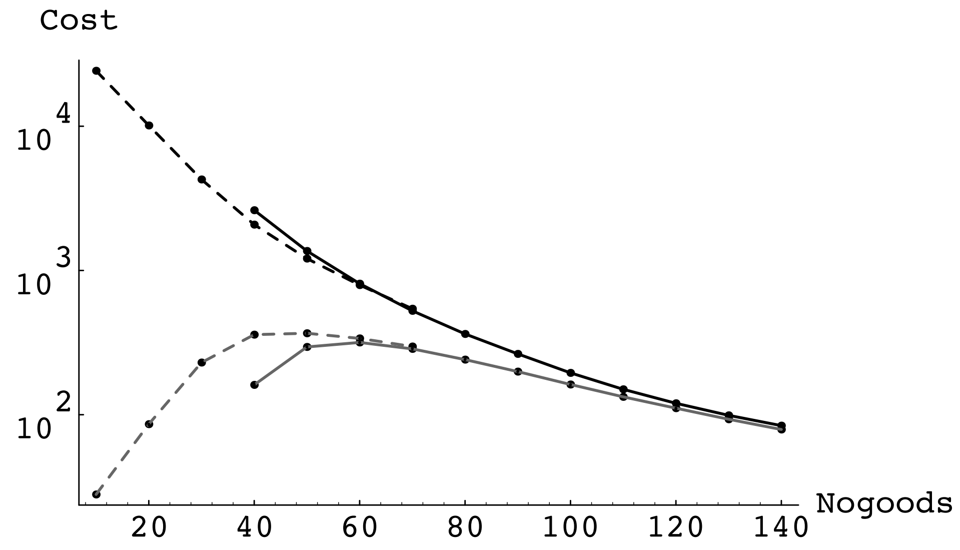

Figure 7: Comparison of median solution cost on a log scale using the same sets of unsolvable problems for chronological backtracking (black) and dynamic backtracking (gray). Dashed lines are for problems generated using the "hill-climbing" method, solid lines for the "generate-select" method. Each point is the median of 1000 problems each solved 100 times, except for the "generate-select" method at 40 nogoods, which is based on 248 problems. Error bars showing 95% confidence intervals are included, but are smaller than the size of the plotted points.

even when the number of solutions is held fixed at zero or one, there is still a peak in solution cost, and at a smaller number of nogoods. Thus the existing explanation does not capture the full range of behaviors. Instead, it appears that there are other factors at work in producing hard problems. By focusing more closely on these factors we can hope to gain a better understanding of the structure of hard problems, which may lead to more precise predictions of search cost.

In contrast to our observations, a monotonic decrease in cost has been reported for unsolvable binary constraint problems [Smith and Dyer, 1996] and for unsolvable 3SAT problems [Mitchell *et al.*, 1992].

What might account for the different pattern we see here? Smith's plots are based on one trial per generated problem, while we solve each generated problem 100 times to avoid characterizing any given problem by what may actually be merely a "bad luck" search path. In addition, Smith used chronological backtracking, but we use dynamic backtracking. To determine the relative importance of these differences, we repeated the search of random CSPs using chronological backtracking. A comparison of this with our previous dynamic backtrack search for unsolvable problems is shown in Figure 7. In this figure, the curves for dynamic backtracking are the same as those for the unsolvable problems shown in Figure 2, except here the cost curves are shown on a logarithmic scale. Because we do not see a peak in unsolvable problems using the less sophisticated method of chronological backtrack, we conclude that it is the choice of algorithm in Smith's work that accounts for the difference in the observed patterns.

This observation raises an important point: the easy-hard-easy pattern is not a universal feature of search algorithms for problems restricted to a fixed number of solutions. This suggests that the competition between number of solutions and pruning, when it occurs, is sufficiently powerful to affect most search algorithms (very simple methods, such as generate-and-test, do not make use of pruning and show a monotonic increase in search cost as the number of solutions decreases). However, only some algorithms are able to exploit the features of weakly constrained problems with a fixed number of solutions that make them easy.

5 Discussion

Our observations on classes of problems with restrictions on the number of solutions they may have shows that the common identification of the peak in solution cost with the algorithm-independent transition in solvability seen in general problem classes does not capture the full generality of the easy-hard-easy pattern.

In our observation for solvable problems, this explanation can be readily modified to use a transition in the existence of solutions beyond those specified by the construction of the class of problems and symmetries those problems might have that constrain the allowable range of solutions. This modification is a simple generalization of the existing explanation based on the competition between the number of solutions and pruning.

When the number of solutions is held constant, however, competition between increased pruning and decreasing number of solutions cannot possibly be responsible for a peak in solution cost. The decrease in search cost for highly constrained problems (to the right of the peak) is adequately explained by the prevailing explanation, based on the increase in pruning with additional constraints. But this does not explain why weakly constrained problems are also found to be easy, at least for some search methods. The low cost of unsolvable problems in the underconstrained region is a new and unexpected observation in light of previous studies of the easy-hard-easy pattern and its explanation. This raises the question of whether there is a different aspect of problem structure that can account for the peak in search cost for problems with a fixed number of solutions.

One possibility that is often mentioned in this context is the notion of critically constrained problems. These are problems just on the boundary between solvable and unsolvable problems, i.e., neither underconstrained (with many solutions) nor overconstrained (with none). This notion forms the basis for another common interpretation of the cost

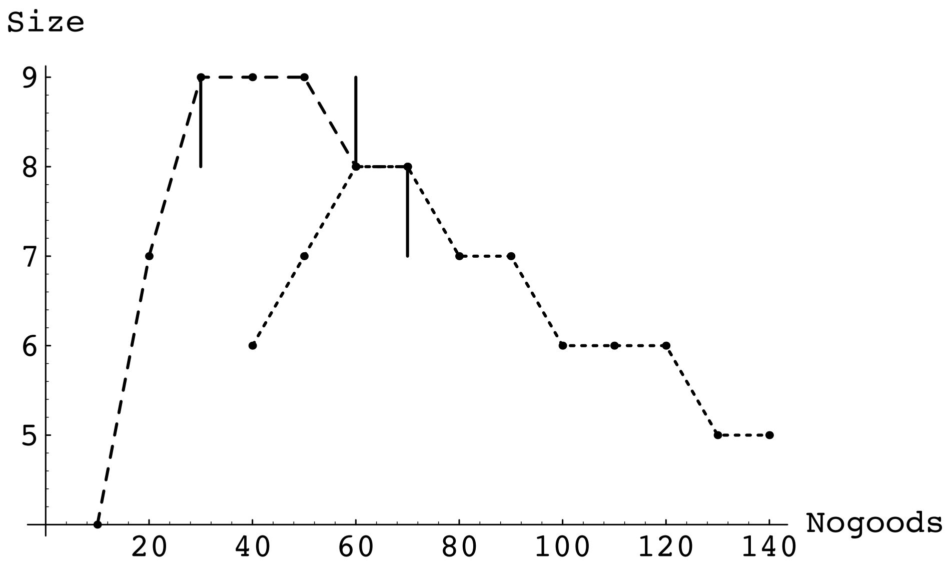

Figure 8: Median minimal unsolvable subproblem size as a function of number of nogoods, for unsolvable problems generated using the "hill-climbing" (dashed line) and "generate-select" (dotted line) methods. Each point is based on the median, for 1000 problems, of the median unsolvable subproblem size for the problem (some problems have multiple minimal unsolvable subproblems). Error bars showing 95% confidence intervals are included.

peak. That is, these critically constrained problems will typically be hard to search (because most of the constraints must be instantiated before any unproductive search paths can be identified) and, since they are concentrated at the transition [Smith and Dyer, 1996], give rise to the search peak. This explanation does not include any discussion of the changes in pruning capability as constraints are added. Taken at face value, this explanation would predict no peak at all for solvable problems because such a class has no transition from solvable to unsolvable problems. Moreover, this description of critically constrained problems is not simply a characteristic of an individual problem (as, say, the number of solutions is) but rather is partly dependent on the class of problems under consideration because the exact location of the transition depends on the method by which problems are generated. Thus, as currently described, the notion of critically constrained problems does not explain our observations nor does it give an explicit way to characterize individual problems.

A more precisely defined alternative characteristic is the size of minimal unsolvable subproblems, i.e., a subproblem that is unsolvable, but for which any subset is solvable. Some problems have more than one minimal unsolvable subproblem. For example, they might have one such subproblem of 5 variables, and another, different one, of say, 6.

We found that problems with 140 nogoods usually have around 40 minimal unsolvable subproblems, those with 70 nogoods have about 5, and problems with 50 or fewer nogoods, just 1 or 2.

The behavior of the size of these minimal unsolvable subproblems is shown in Figure 8. Comparing with Figure 2, we see that the peak in the size of minimal unsolvable subproblems matches the location of the search cost peak for unsolvable problems. This result is independent of whether we plot medians, as shown in Figure 8, or means or mimina, which we have not shown here; regardless of the particular choice of how to plot minimal unsolvable subproblem size, the shape of the plot is the same. Moreover, the location of the peaks in minimal unsolvable subproblem size for the different generation methods correspond to the location of their respective search cost peaks. The peak in both search cost and minimal unsolvable subproblem size occurs at around 40 nogoods for problems generated using the "hill-climbing" method, and significantly higher, around 60 nogoods, for problems generated using the "generate-select" method. The strong correspondence between minimal unsolvable subproblem size and search cost is very suggestive that minimal unsolvable subproblem size is a structural characteristic of problems that plays an important role in search cost.

The behavior of the minimal unsolvable subproblem size as a function of the number of constraints has a simple explanation. In order for weakly constrained problems to be unsolvable, they will generally need to concentrate most of the available constraints on a few variables in order to make all assignments inconsistent. This will tend to give small minimal unsolvable subproblems. As more constraints are added, this concentration is no longer required and, since problems where most of the randomly selected constraints happen to be concentrated on a few variables are rare, we can expect larger minimal unsolvable subproblems. Finally, as more and more constraints are added, the increased pruning is equivalent to the notion that instantiating only a few variables is all that is required to find an inconsistency. This means we can expect a large number of unsolvable subproblems. This qualitative description corresponds to what we observe in Figure 8.

Our observations of weakly constrained problems suggest that some search algorithms, such as dynamic backtracking, are able to rapidly focus in on one of the unsolvable subproblems and hence avoid the extensive thrashing, and high search cost, seen in other methods. In such cases, one would expect that the smaller the unsolvable subproblem is, the easier it will be for the search to determine there are no solutions.

This discussion is also relevant to solvable problems: once a series of choices is made during the search that precludes a solution, the remaining subproblem is now an unsolvable one. For example, in a 10 variable CSP, suppose values are given to the first two variables that are incompatible with all solutions to the problem. This means that in the context of these two assignments, the remaining 8 variables constitute an unsolvable subproblem. The number of search steps required to determine this subproblem is in fact unsolvable will be the cost added to the search before backtracking to the original two variables and trying a new assignment for one of them. Thus, the cost of identifying unproductive search choices for solvable problems is determined by how rapidly the associated unsolvable subproblem can be searched. As described above, when there are few constraints we can expect that such unsolvable subproblems will themselves have small minimal unsolvable subproblems and hence be easy to search with methods that are able to focus on such subproblems. While the unsolvable subproblems associated with incorrect variable choices in solvable problems may have a different structure, this argument suggests that the changes in minimal unsolvable subproblems explain the behavior of solvable problems with a fixed number of solutions as well. Thus this could also explain observations of thrashing behavior for rare but extremely costly searches seen in the underconstrained region [Gent and Walsh, 1994a, Hogg and Williams, 1994]. It would also be interesting to study the behavior of local repair search methods for problems with a single solution to see if they also are affected by the change in minimal subproblem size.

6 Conclusions

We have presented evidence that the explanation of the easy-hard-easy pattern in solution cost based on a competition between changes in the number of solutions and pruning is insufficient to explain the phenomenon completely for sophisticated search methods. It does explain the overall pattern for problems not restricted by solvability or number of solutions. However, the explanation fails when the number of solutions is held constant and sophisticated search methods are used. In these cases the solution cost peak does not disappear as would be predicted. Alternatively, we can view this explanation as adequate for less sophisticated methods that are not able to readily focus in on unsolvable subproblems encountered during the search.

By considering relatively small search problems, we are able to exhaustively examine the properties of the search space. This allowed us to identify a new aspect of problem structure that is important for search behavior: the size of minimal unsolvable subproblems. This contrasts with much work in this area that involves solving problems as large as feasible within reasonable time bounds. While this approach gives a better indication of the asymptotic behavior of the transition, it is not suitable for exhaustive evaluation of the nature of the search spaces encountered.

We believe that detailed examination of the structure of combinatorial problems can yield information about why certain types of problems are difficult or easy. As a class graph coloring or random CSPs are NP-complete, yet in practice many such problems are actually very easy. In addition, while theoretical work in this area has produced predictions that are asymptotically correct on average, the variance among individual problems in a predicted class is enormous. Increased understanding of the relationships between problem structure, problem solving algorithm, and solution cost is important to determining whether, and if so, how, we can determine prior to problem solving which problems are easy versus infeasibly hard. In contrast to previous theoretical studies that focus on the number of solutions, this suggests the size of minimal unsolvable subproblems is an alternate characteristic to study with the potential for producing a more precise characterization of the transition behavior and the nature of hard search problems.

References

[Asahiro *et al.*, 1993] Y. Asahiro, K. Iwama, and E. Miyano. Random generation of test instances with controlled attributes. In *Second DIMACS Challenge Workshop*, 1993.

[Cha and Iwama, 1995] B. Cha and K. Iwama. Performance test of local search algorithms using new types of random CNF formulas. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 304–310, Montréal, Québec, Canada, 1995.

[Cheeseman *et al.*, 1991] P. Cheeseman, B. Kanefsky, and W. Taylor. Where the *really* hard problems are. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 331–337, Sydney, Australia, 1991.

[Crawford and Auton, 1993] J. M. Crawford and L. D. Auton. Experimental results on the cross-over point in satisfiability problems. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 21–27, Washington, DC, USA, 1993.

[Gent and Walsh, 1994a] Ian P. Gent and Toby Walsh. Easy problems are sometimes hard. *Artificial Intelligence*, 70:335–345, 1994.

[Gent and Walsh, 1994b] I.P. Gent and T. Walsh. The SAT phase transition. In A.G. Cohn, editor, *Proceedings of the ECAI-94*, pages 105–109. John Wiley and Sons, 1994.

[Gent *et al.*, 1995] Ian P. Gent, Ewan MacIntyer, Patrick Prosser, and Toby Walsh. Scaling effects in the CSP phase transition. In U. Montanari and F. Rossi, editors, *Proc. of Principles and Practices of Constraint Programming PPCP95*, pages 70–87. Springer-Verlag, 1995.

[Ginsberg, 1993] Matthew L. Ginsberg. Dynamic backtracking. *Journal of Artificial Intelligence Research*, 1:25–46, 1993.

[Hogg and Williams, 1994] Tad Hogg and Colin P. Williams. The hardest constraint problems: A double phase transition. *Artificial Intelligence*, 69:359–377, 1994.

[Hogg, 1996] Tad Hogg. Refining the phase transitions in combinatorial search. *Artificial Intelligence*, 1996. to appear.

[Johnson *et al.*, 1991] David S. Johnson, Cecilia R. Aragon, Lyle A. McGeoch, and Catherine Schevon. Optimization by simulated annealing: An experimental evaluation; part ii, graph coloring and number partitioning. *Operations Research*, 39(3):378–406, 1991.

[Kask and Dechter, 1995] Kalev Kask and Rina Dechter. GSAT and local consistency. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 616–622, Montréal, Québec, Canada, 1995.

[Kirkpatrick and Selman, 1994] Scott Kirkpatrick and Bart Selman. Critical behavior in the satisfiability of random boolean expressions. *Science*, 264:1297–1301, 1994.

[Mitchell *et al.*, 1992] D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of SAT problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 459–465, San Jose, CA, USA, 1992.

[Prosser, 1996] Patrick Prosser. An empirical study of phase transitions in binary constraint satisfaction problems. *Artificial Intelligence*, 1996. to appear.

[Smith and Dyer, 1996] Barbara M. Smith and Martin E. Dyer. Locating the phase transition in binary constraint satisfaction problems. *Artificial Intelligence*, 1996. to appear.

[Smith, 1994] Barbara M. Smith. Phase transition and the mushy region in constraint satisfaction problems. In A.G. Cohn, editor, *Proceedings of the ECAI-94*, pages 100–104. John Wiley and Sons, 1994.

[Snedecor and Cochran, 1967] George W. Snedecor and William G. Cochran. *Statistical Methods.* Iowa State Univ. Press, Ames, Iowa, 6th edition, 1967.

[Williams and Hogg, 1994] C. P. Williams and T. Hogg. Exploiting the deep structure of constraint problems. *Artificial Intelligence*, 70:73–117, 1994.

[Yugami *et al.*, 1994] Nobuhiro Yugami, Yuiko Ohta, and Hirotaka Hara. Improving repair-based constraint satisfaction methods by value propagation. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 344–349, Seattle, WA, USA, 1994.