Expected Gains from Parallelizing Constraint Solving for Hard Problems

Tad Hogg and Colin P. Williams

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

Abstract

A number of recent studies have examined how the difficulty of various NP-hard problems varies with simple parameters describing their structure. In particular, they have identified parameter values that distinguish regions with many hard problem instances from relatively easier ones. In this paper we continue this work by examining independent parallel search. Specifically, we evaluate the speedup as function of connectivity and search difficulty for the particular case of graph coloring with a standard heuristic search method. This requires examining the full search cost distribution rather than just the more commonly reported mean and variance. We also show similar behavior for a single-agent search strategy in which the search is restarted whenever it fails to complete within a specified cost bound.

Introduction

Several recent studies have related the structure of constraint satisfaction problems to the difficulty of solving them with search [Cheeseman et al., 1991, Mitchell et al., 1992, Williams and Hogg, 1992, Crawford and Auton, 1993, Gent and Walsh, 1993, Williams and Hogg, 1993]. Particular values of simple parameters, describing the problem structure, that lead to hard problem instances, on average, were identified. These values are also associated with high variance in the solution cost for different problem instances, and for a single instance with respect to different search methods or a single nondeterministic method with, e.g., different initial conditions or different tie-breaking choices made when the search heuristic ranks some choices equally. Structurally, these hard problems are characterized by many large partial solutions, which prevent early pruning by many types of heuristics.

Can these observations be exploited in practical search algorithms? One possibility is to run several searches independently in parallel, stopping when any process first finds a solution (or determines there are none) [Fishburn, 1984, Helmbold and McDowell, 1989, Pramanick and Kuhl, 1991, Kornfeld, 1981, Imai et al., 1979, Rao and Kumer, 1992, Mehrotra and Gehringer, 1985, Ertel, 1992]. Since the benefit of this approach relies on variation in the individual methods employed, the high variance seen for the hard problems suggests it should be particularly applicable for them [Cheeseman et al., 1991, Rao and Kumer, 1992]. In some cases, this approach could be useful even if multiplexed on a serial machine [Janakiram et al., 1987]. This method is particularly appealing since it requires no communication between the different processes and is very easy to implement.

However, the precise benefit to be obtained from independent parallel searches is determined by the nature of the full distribution of search cost, not just by the variance. In this paper, we present experimental results on the expected speedup from such parallel searches for a particular well-studied example, graph coloring. By evaluating the speedup obtainable from graphs of different connectivities we investigate consequences of different structural properties for the benefit of parallelization, even when such problems are equally hard for a serial search.

In the remainder of the paper we describe the graph coloring search problem and show the types of cost distributions that arise for a heuristic search method. We then show how the speedup from parallelization is determined from the distribution and present empirically observed speedups for a variety of graphs. We also present a similar analysis and experiments for a single-agent search strategy.

Graph Coloring Problems

The graph coloring problem consists of a graph, a specified number of colors, and the requirement to find a color for each node in the graph such that no pair of adjacent nodes (i.e., nodes linked by an edge in the graph) have the same color. Graph coloring has received considerable attention and a number of search methods have been developed [Minton et al., 1990, Johnson et al., 1991, Selman et al., 1992]. This is a well-known NP-complete problem whose solution cost grows exponentially in the worst case as the number of nodes in the graph increases.

For this problem, the average degree of the graph γ (i.e., the average number of edges coming from a node in the graph) distinguishes relativelv easy from harder problems, on average. In this paper, we focus on the case of 3–coloring (i.e., when 3 different colors are available).

In our experiments we used a complete, depth-first backtracking search based on the Brelaz heuristic [Johnson et al., 1991] which 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 (with any remaining ties broken randomly). For each node, the smallest color consistent with the previous assignments is chosen first, with successive choices made when the search is forced to backtrack. This complete search method is guaranteed to eventually terminate and produce correct results.

The Cost Distribution

For our experiments we used randomly generated graphs with 100 nodes and with a specified average connectivity γ. For these problems, there are two distinct regions with hard problems. The first, near $\gamma = 4.5$ has a fairly high density of hard instances. The second, at somewhat lower connectivities, has mostly easy problems but occasionally instances with extremely high search costs. These extreme cases have such large cost that they dominate the mean cost in this region. These observations are summarized in Fig. 1.

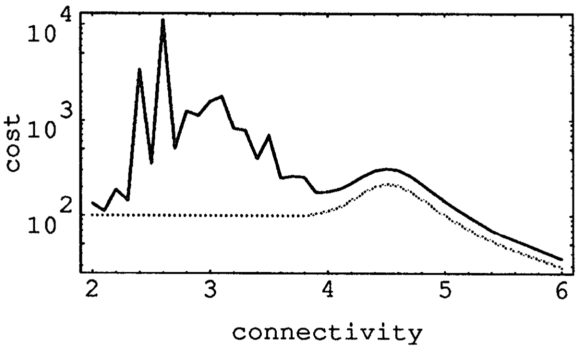

Fig. 1. Search cost as a function of average connectivity γ of the graphs with 100 nodes. The black curve shows the mean cost and the gray one is the median cost. Values are shown in increments of 0.1 for γ and are based on 50,000 samples at each point. The large variance in search cost for the lower connectivities produces the sharp peaks in the mean cost, an indication that many more samples are required to determine it accurately.

For the behavior of parallel search we need to consider the distribution of search costs. For the cases in which a single search solves the problem rapidly there is not much to be gained from parallel methods. Thus we focus on the behavior of problems in the hard regions of intermediate connectivity. Fig. 2 gives examples of the types of cost distribution we encountered. Specifically, there were two distinct shapes. On one hand, we found multimodal distributions with a wide range of individual costs. In these cases, the search often completes rapidly but because of the occasional high-cost instance, the mean search cost is relatively large. On the other hand, we found more tightly clustered distributions in which all the searches required about the same, large cost. As described below, these distinct distribution shapes give rise to very different potential for parallel speedup through multiple searches.

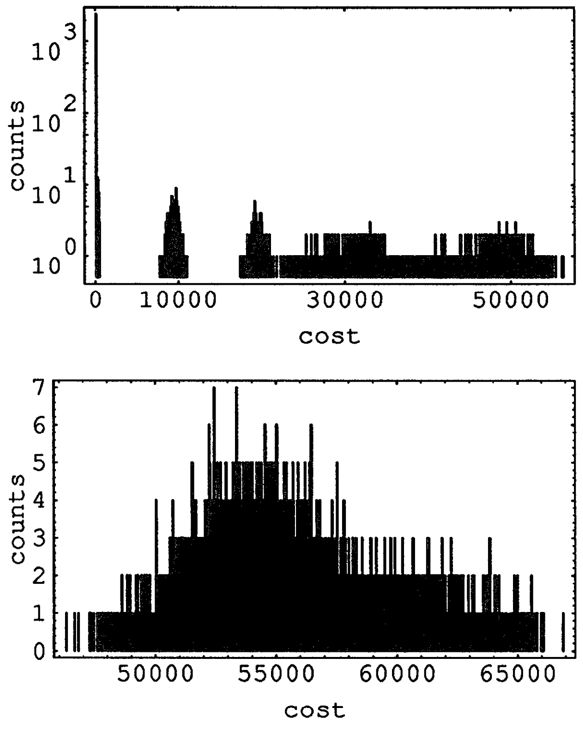

Fig. 2. The search cost distribution for two 100–node graphs with $\gamma = 3.5$. In each case, the graph was searched 10,000 times. The plots give the number of times each cost was found. The first case is a multimodal distribution with average cost $T_1 = 15122$ and a large possible speedup: $S(10) = 76$, $S_{opt} = 37$ with $\tau^* = 412$. The second case is a fairly sharp unimodal distribution (note the expanded cost scale on the horizontal axis). It has average cost $T_1 = 55302$ and very limited speedup: $S(10) = 1.1$ and $S_{opt} = 1$ with no cutoff (i.e., $\tau^* = \infty$). These quantities are defined in Eqs. 1 and 3.

Speedup with Parallel Search

When a heuristic search involves some random choices, repeated runs on the same problem can give different search costs. This is especially true when incorrect choices made early in the search are undone only after a large number of steps. In such cases, one could benefit from running multiple, independent versions of the search, stopping when the first one completes. In this section, we show how the expected speedup for this method is determined by the full cost distribution of the search method.

Specifically, for a given problem instance, let $p(i)$ be the probability that an individual search run terminates after exactly i search steps. We also define the cumulative distribution, i.e., the probability that the search requires *at least* i steps, as $q(i) = \sum_{j \geq i} p(j)$. For a group of k independent searches the search cost for the group is defined as the cost for the first agent that terminates, i.e., the minimum of the individual finishing times. The probability that the group as a whole requires at least i steps, $q_k(i)$, is just the probability that *all* of the individual searches require at least i steps. Because the searches run independently, we have simply $q_k(i) = q(i)^k$. Thus the probability that the group finishes in exactly i steps is $p_k(i) = q(i)^k - q(i+1)^k$.

With these expressions we can now determine the average speedup. Specifically, again for a given problem instance, let T_k be the average cost for a group of k agents to solve the problem. Then we define the speedup as

$$S(k) = \frac{T_1}{T_k} \quad (1)$$

This can be expressed in terms of the cost distribution by noting that $T_k = \sum_i i p_k(i)$. Whilst this measure of speedup is a commonly used measure of parallel performance, we should note that for the extended distributions seen in our studies, it can differ from "typical" behavior when the mean search times are greatly influenced by rare, high-cost events. Even in these cases, however, this measure does give some insight into the nature of the distribution and allows for a simple comparison among different classes of graphs.

In our experimental studies, we estimate the cost distribution $p(i)$ by repeating the individual search N times (for most of our experiments we used $N = 100$ to allow for testing many graphs, but saw similar behaviors in a few graphs with more samples, such as $N = 10^4$ for the examples of Fig. 2). Let $n(i)$ be the number of searches that finished in exactly i steps. Then we used $p(i) \approx n(i)/N$ to obtain an estimate for the speedup for a given problem. By repeating this procedure over many different graphs, all having the same connectivity, we can obtain an estimate of the average speedup, $\langle S(k) \rangle$, as a function of average connectivity, γ. Alternatively we can also ask how large k needs to be in order to solve a problem, having a particular connectivity, within a prespecified cost limit.

Experimental Results: Speedup

We searched a number of graphs at various connectivities and recorded the distribution of search times. From this we obtained estimates of the expected search cost for a single search and the speedup when several searches are run independently in parallel. To study the two regions of hard problems we selected graphs with $\gamma = 3.5$ and $\gamma = 4.5$. More limited experiments with $\gamma = 3.0$ were qualitatively similar to the $\gamma = 3.5$ case.

The speedups for $\gamma = 3.5$ are shown in Fig. 3 as a function of single-agent search cost. Whilst we see many samples with fairly limited speedup, this class of graphs generally shows an increased speedup with single search cost. A similar story is seen in Fig. 4 for the average speedup among these samples as a function of number of parallel searches. This shows an increasing speedup, especially for the harder cases. Together with Fig. 1 we conclude that at this connectivity we have mostly easy cases, but many of the harder cases can often benefit increasingly from parallelization.

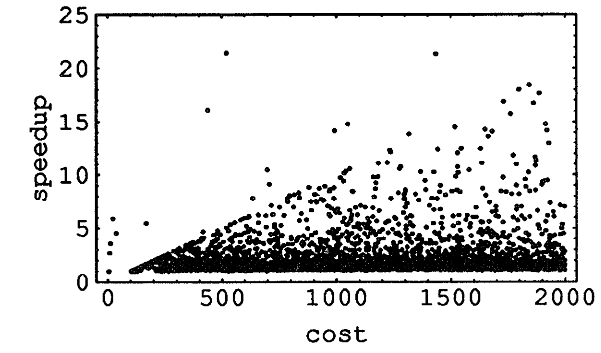

Fig. 3. Speedup $S(10)$ for 10 agents vs. average single-agent cost T_1 for 100–node graphs at $\gamma = 3.5$. Each graph was searched 100 times to estimate the single-agent cost distribution. There were also a few samples with larger speedups than shown in the plot, as well as samples with substantially higher costs which continued the trend toward increasing speedups.

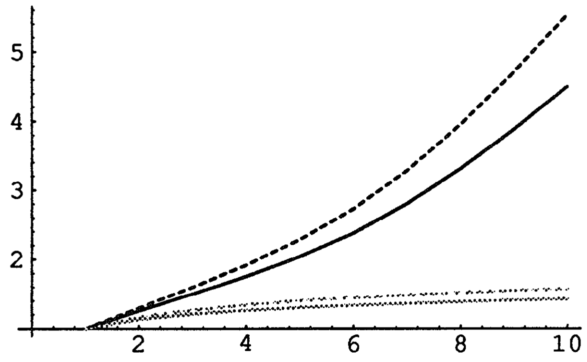

Fig. 4. Average speedup $\langle S(k) \rangle$ (black) and median speedup (gray) vs. number of agents k for 100–node graphs at $\gamma = 3.5$. The solid curves include all samples while the dashed ones include only those whose average single-agent cost was at least 1000.

The behavior is different for $\gamma = 4.5$ as shown in Fig. 5. In this case most samples exhibit limited speedup. In particular, there is no increase in speedup with single search cost. In Fig. 6 we see the limited benefit of additional parallel searches, both for all samples and those with high cost.

Speeding Up a Single Agent

Another way to use the cost distribution of an individual search method for a given problem is to devise a better

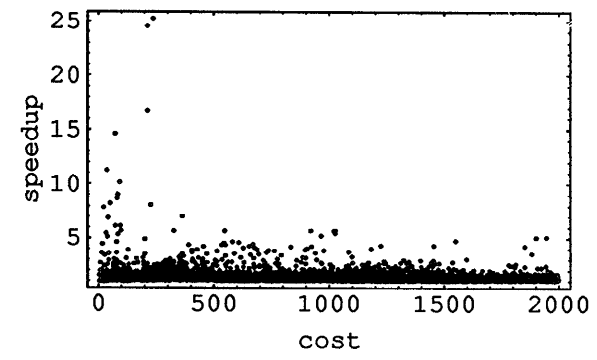

Fig. 5. Speedup $S(10)$ for 10 agents vs. average single-agent cost T_1 for 100–node graphs at $\gamma = 4.5$. Each graph was searched 100 times to estimate the single-agent cost distribution.

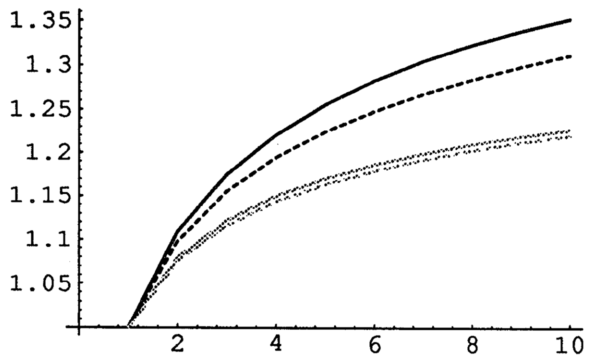

Fig. 6. Average speedup $\langle S(k) \rangle$ (black) and median speedup (gray) vs. number of agents k for 100–node graphs at $\gamma = 4.5$. The solid curves include all samples while the dashed ones include only those whose average single-agent cost was at least 1000.

strategy for use by a single agent. For example, if the agent finds it is taking an inordinately long time in the search, it can abandon the search and start over. With a prudent choice of when to quit, this strategy can lead to improved performance. Specifically, suppose the agent restarts its search after τ steps. Then the time to solve the problem will be given by $T = \tau(m-1) + t$, where m is the number of search repetitions required to find a case that finishes within the time bound and t is the time spent on this last search case. The expected overall search cost for this strategy when applied to a given problem is just $\ell(\tau) = \tau(\overline{m} - 1) + \overline{t}$ with the overbars denoting averages over the individual search distribution for this problem. Let $Q(i) = 1 - q(i+1)$ be the probability the individual search completes in *at most* i steps. Then $Q(\tau)$ is the probability that a search trial will succeed before being terminated by the cost bound. Thus the average number of repetitions required for success is $\overline{m} = \frac{1}{Q(\tau)}$. The expected cost within a repetition that does in fact succeed is given by $\overline{t} = \sum_{i \leq \tau} ip(i|\tau)$ where $p(i|\tau) = \frac{p(i)}{Q(\tau)}$ is the conditional probability the search succeeds after exactly i steps given it succeeds in at most τ steps. Finally, using the identity $\sum_{i \leq \tau} ip(i) = \tau Q(\tau) - \sum_{i<\tau} Q(\tau)$ we obtain the expression for, $\ell(\tau)$, the expected overall search cost for the "restart after τ steps" strategy:

$$\ell(\tau) = \frac{1}{Q(\tau)} \left(\tau - \sum_{i=0}^{\tau-1} Q(i) \right) \tag{2}$$

In particular, when the individual search is never aborted, corresponding to $\tau = \infty$, we recover the average individual search cost, i.e., $\ell(\infty) = T_1$.

Among all possible choices of the cutoff time, even allowing it to vary from one repetition of the search to the next, the optimal strategy [Luby et al., 1993] is obtained by selecting a single termination time τ^* which minimizes $\ell(\tau)$. The speedup of this search method over the original one is then given by

$$S_{opt} = \frac{T_1}{\ell(\tau^*)} \tag{3}$$

Finally we should note that this strategy of restarting searches if they don't finish within a prespecified time can be combined with independent parallel agents to give even greater potential speedups [Luby and Ertel, 1993].

Experimental Results: Optimal Strategy

Unfortunately, in practice one never knows the cost distribution for a new problem before starting the search. However, we can evaluate it for a range of graph coloring problems as an additional indication of the benefit of independent search for graphs of different connectivities. This can also give some indication of the appropriate termination time to use.

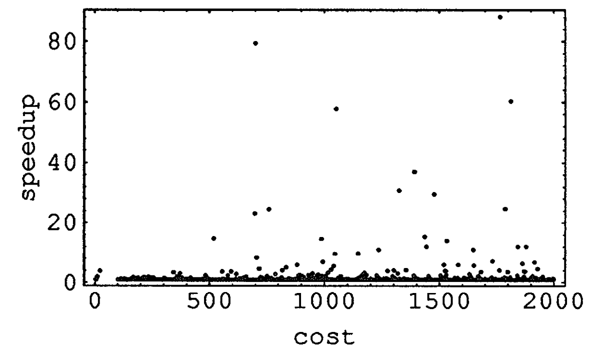

Fig. 7. Speedup S_{opt} for optimal single-agent search vs. average single-agent cost T_1 for 100–node graphs at $\gamma = 3.5$. Each graph was searched 100 times to estimate the single-agent cost distribution. We also found samples with substantially higher costs than shown in the plot, which continued the trend toward increasing speedups.

In Figs. 7 and 8 we show the speedup of this "optimal restart" single search strategy compared to the average "never restart" single search time. We see the same

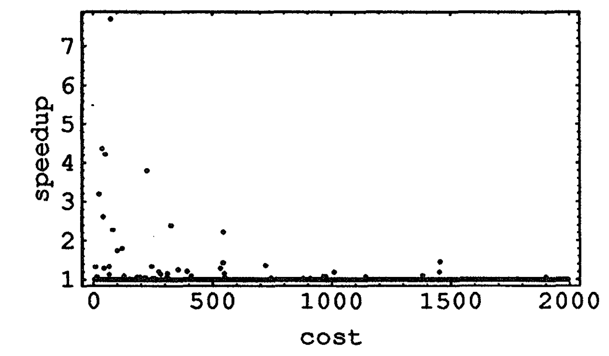

Fig. 8. Speedup S_{opt} for optimal single-agent search vs. average single-agent cost T_1 for 100–node graphs at $\gamma = 4.5$. Each graph was searched 100 times to estimate the single-agent cost distribution.

qualitative behavior as with our previous results on independent parallel search: no significant speedup for many of the harder problems but with somewhat more speedup for the cases at the lower connectivity.

As with the independent parallel search, an extended multimodal distribution allows this strategy to greatly outperform a single search, on average. Thus these results again demonstrate the existence of the two types of cost distributions.

Discussion

We have examined the benefit from independent parallel search for a particular constraint satisfaction problem, as a function of a parameter characterizing the structure of the problem. This extends previous work on the existence of hard and easy regions of problems to the question of what kinds of search methods are suitable for the hard instances. With our samples we saw that, for the most part, independent parallelization gives fairly limited improvement for the hard cases. This is consistent with previous observations that the hard problems persist for a range of common heuristic search algorithms; and the resulting conjecture that these problems are intrinsically hard because of their structure. Specifically, this may be due to their having a large number of partial solutions, few of which can be extended to full solutions [Cheeseman et al., 1991, Williams and Hogg, 1992]. These large partial solutions in turn make it difficult for heuristics, based on a local evaluation of the search space, to rapidly prune unproductive search paths. Our results for the graphs with limited speedup lend support to this conjecture.

There were also a number of cases with significant speedup, especially for the lower connectivity graphs. Since this is due to a very extended, multimodal distribution, an interpretation of these cases is that the overall cost is very sensitive to the early decisions made by the heuristic. While the correct choice leads to relatively rapid search, the structure of the problem does not readily provide an indication of incorrect choices. Thus in the latter case the search continues a long time before finally revising the early choices.

As a caveat, we should point out that our observations are based on a limited sampling of the individual search cost distribution (i.e., 100 trials per graph). Because of the extended nature of these distributions, additional samples may reveal some rare but extremely high cost search runs. These runs could significantly increase the mean individual search time while having only a modest effect on the parallel speeds. In such cases, the estimate of the potential speedup based on our limited sampling would be too low. To partially address this issue we searched a few graphs with significantly more trials (up to 10^4 per graph), finding the same qualitative behaviors.

There are a number of future directions for this work. One important question is the extent to which our results apply to other constraint satisfaction problems which also exhibit this behavior of easy and hard regions; as well as to other types of search methods such as genetic algorithms [Goldberg, 1989] or simulated annealing [Johnson et al., 1991]. For instance, when applied to optimization problems, one would need to consider the quality of solutions obtained as well as the search time required. Another issue concerns how the behaviors reported here may change as larger problems are considered.

Curiously, independent studies of other NP-hard problems, using very different search algorithms, have discovered qualitatively similar kinds of cost distributions [Ertel, 1992]. It is possible, therefore, that such distributions are quite generic across many different problems and algorithms. If so, our observation that the potential gains from independent parallel search vary with some parameter characterizing the problem structure, might be useful in designing an overall better parallel search algorithm. Specifically, as hard problems in the "hard" region do not appear to benefit much from independent parallelization we might prefer instead attempt to solve problems in this region by some more sophisticated parallel search method.

We have experimented with one such method, which we call "cooperative problem solving" in which the different computational agents exchange and reuse information found during the search, rather than executing independently. If the search methods are sufficiently diverse but nevertheless occasionally able to utilize information found in other parts of the search space, greater performance improvements are possible [Hogg and Williams, 1993] including, in some cases, the possibility of superlinear speedups [Clearwater et al., 1991].

References

Cheeseman, P., Kanefsky, B., and Taylor, W. M. (1991). Where the really hard problems are. In Mylopoulos, J. and

Reiter, R., editors, *Proceedings of IJCAI91*, pages 331–337, San Mateo, CA. Morgan Kaufmann.

Clearwater, S. H., Huberman, B. A., and Hogg, T. (1991). Cooperative solution of constraint satisfaction problems. *Science*, 254:1181–1183.

Crawford, J. M. and Auton, L. D. (1993). Experimental results on the cross-over point in satisfiability problems. In *Proc. of the Eleventh Natl. Conf. on AI (AAAI93)*, pages 21–27, Menlo Park, CA. AAAI Press.

Ertel, W. (1992). Random competition: A simple, but efficient method for parallelizing inference systems. In Fronhofer, B. and Wrightson, G., editors, *Parallelization in Inference Systems*, pages 195–209. Springer, Dagstuhl, Germany.

Fishburn, J. P. (1984). *Analysis of Speedup in Distributed Algorithms*. UMI Research Press, Ann Arbor, Michigan.

Gent, I. P. and Walsh, T. (1993). An empirical analysis of search in GSAT. *J. of AI Research*, 1:47–59.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, NY.

Helmbold, D. P. and McDowell, C. E. (1989). Modeling speedup(n) greater than n. In Ris, F. and Kogge, P. M., editors, *Proc. of 1989 Intl. Conf. on Parallel Processing*, volume 3, pages 219–225, University Park, PA. Penn State Press.

Hogg, T. and Williams, C. P. (1993). Solving the really hard problems with cooperative search. In *Proc. of the Eleventh Natl. Conf. on AI (AAAI93)*, pages 231–236, Menlo Park, CA. AAAI Press.

Imai, M., Yoshida, Y., and Fukumura, T. (1979). A parallel searching scheme for multiprocessor systems and its application to combinatorial problems. In *Proc. of IJCAI-79*, pages 416–418.

Janakiram, V. K., Agrawal, D. P., and Mehrotra, R. (1987). Randomized parallel algorithms for prolog programs and backtracking applications. In Sahni, S. K., editor, *Proc. of 1987 Intl. Conf. on Parallel Processing*, pages 278–281, University Park, PA. Penn State Univ. Press.

Johnson, D. S., Aragon, C. R., McGeoch, L. A., and Schevon, C. (1991). Optimization by simulated annealing: An experimental evaluation; part ii, graph coloring and number partitioning. *Operations Research*, 39(3):378–406.

Kornfeld, W. A. (1981). The use of parallelism to implement a heuristic search. In *Proc. of IJCAI-81*, pages 575–580.

Luby, M. and Ertel, W. (1993). Optimal parallelization of las vegas algorithms. Technical report, Intl. Comp. Sci. Inst., Berkeley, CA.

Luby, M., Sinclair, A., and Zuckerman, D. (1993). Optimal speedup of las vagas algorithms. Technical Report TR-93-010, Intl. Comp. Sci. Inst., Berkeley, CA.

Mehrotra, R. and Gehringer, E. F. (1985). Superlinear speedup through randomized algorithms. In Degroot, D., editor, *Proc. of 1985 Intl. Conf. on Parallel Processing*, pages 291–300, Washington, DC. IEEE.

Minton, S., Johnston, M. D., Philips, A. B., and Laird, P. (1990). Solving large-scale constraint satisfaction and scheduling problems using a heursitic repair method. In *Proceedings of AAAI-90*, pages 17–24, Menlo Park, CA. AAAI Press.

Mitchell, D., Selman, B., and Levesque, H. (1992). Hard and easy distributions of SAT problems. In *Proc. of 10th Natl. Conf. on Artificial Intelligence (AAAI92)*, pages 459–465, Menlo Park. AAAI Press.

Pramanick, I. and Kuhl, J. G. (1991). Study of an inherently parallel heuristic technique. In *Proc. of 1991 Intl. Conf. on Parallel Processing*, volume 3, pages 95–99.

Rao, V. N. and Kumer, V. (1992). On the efficiency of parallel backtracking. *IEEE Trans. on Parallel and Distributed Computing*.

Selman, B., Levesque, H., and Mitchell, D. (1992). A new method for solving hard satisfiability problems. In *Proc. of 10th Natl. Conf. on Artificial Intelligence (AAAI92)*, pages 440–446, Menlo Park, CA. AAAI Press.

Williams, C. P. and Hogg, T. (1992). Using deep structure to locate hard problems. In *Proc. of 10th Natl. Conf. on Artificial Intelligence (AAAI92)*, pages 472–477, Menlo Park, CA. AAAI Press.

Williams, C. P. and Hogg, T. (1993). Extending deep structure. In *Proc. of the Eleventh Natl. Conf. on AI (AAAI93)*, pages 152–157, Menlo Park, CA. AAAI Press.