In *Proceedings of the First International Conference on AI Planning Systems* (1992). College Park, MD: Morgan Kaufmann.

Systematic and Nonsystematic Search Strategies

Pat Langley
(LANGLEY@PTOLEMY.ARC.NASA.GOV)
AI Research Branch (M/S 269-2)
NASA Ames Research Center
Moffett Field, CA 94035 USA

Abstract

In this paper we compare the relative costs of a systematic problem-solving method – depth-first search – and a nonsystematic method – iterative sampling. An average-case analysis reveals that, for a well-specified class of domains, depth-first search always requires less effort when there exists a single solution, and is generally superior on tasks with shallow solution paths and few solutions. In contrast, iterative sampling is superior on tasks with deep solution paths and many solutions. Depth-first search scales better to cases with high branching factors if the number of solutions remains the same, but random search scales better when the density of solutions remains constant. The search costs predicted by the analysis closely fit the costs observed in experiments with artificial search tasks. We also relate iterative sampling to other methods, including iterative broadening, which is based on similar intuitions.

1. Introduction

Most AI problem-solving systems employ systematic methods such as depth-first, breadth-first, and best-first search, which retain information about states they have considered to avoid duplication of effort. In contrast, humans are remarkably nonsystematic in their problem solving, using methods that may cover the same ground many times. Typically, researchers have assumed that such behavior resulted from limitations in human short-term memory, and that when such limits are not present, the systematic approach should always be preferred. However, in this paper we argue that this intuition is incorrect, and that there are situations in which nonsystematic methods are superior to systematic ones.

Our analysis will deal with a problem space that has a uniform branching factor b, and in which one or more solutions lie exactly at depth d. We also assume that information about success or failure is not available until one reaches level d. Finally, we assume that the search space is a tree rather than a directed graph, so the node-branching and edge-branching factors (Korf, 1985) are equal. One simple problem of this type involves cracking a safe with b settings that requires d turns. One can also formulate some constraint satisfaction tasks (e.g., scheduling) in these terms.

At each branch point in the search tree, one must select from among b alternative children. We will use e to refer to the expected number of nodes one must try at each branch point before selecting a node that lies on a solution path, provided the path one has traversed so far also leads to a solution. In some domains, heuristics are available to bias this selection and reduce e to a reasonable level. For instance, a skilled safe cracker might occasionally hear a click that suggests a likely setting, significantly reducing her expected amount of search. In other domains, e may be low because many solutions exist. We will focus on this latter issue, although we will return briefly to the effects of heuristics in the final section.

2. Analysis of Two Search Algorithms

We will consider two algorithms that handle the class of problems described above. The first method, *depth-first search*, systematically explores each node and its associated subtree in turn, without duplicating any effort. However, one disadvantage of this strategy is that, having selected a node N, it does not return and consider N's siblings unless it has expanded the entire subtree below N and found no solution. Thus, a selection error high in the tree can lead to considerable unnecessary search.

We will call the second algorithm *iterative sampling*, though we will sometimes refer to *random* or *nonsystematic* search. At each branch point in the search, iterative sampling selects a node at random and then recurses until it reaches the depth limit. Thus, the method is similar to a greedy technique in that it selects a single option without backtracking, though it need not use an evaluation function. If the generated path does not lead to a solution, iterative sampling begins again at the initial state. The method continues in this fashion until it finds a path that solves the problem. Unlike depth-first search, the nonsystematic algorithm retains no memory of the states it has visited, so it can retrace entire paths. However, decisions made high in the tree cannot lock the method into a 'wild goose chase' as in the depth-first scheme, so it seems possible that iterative sampling may outperform the more systematic strategy.

2.1 Overall Cost of the Algorithms

In order to determine the relative cost of these algorithms, we will analyze their average-case behavior as a function of the branching factor b, the solution depth d, and the expected number of nodes e considered at each branch point before selecting the correct one, if it exists in the set of alternatives. Later we will ground e in other factors, but for now we assume it as a separate parameter of the domain and method. We will measure cost as the total number of nodes generated during search, assuming that, at each branch point, one generates all children before any are expanded themselves. We use this metric because it is biased in favor of the systematic strategy, which generates each node only once.

Let us first examine the behavior of depth-first search. Even if the algorithm selects the right node on its first attempt at each branch point, it will generate a total of bd nodes. In addition, recall that depth-first search expands the entire subtree below each node that it selects incorrectly. If the search has reached a node N at level j out of d, then there remain $d-j$ levels to explore, and the total number of nodes below (but not including) N is $\sum_{k=1}^{d-j} b^k$. If we let e_{dfs} be the expected number of nodes required for depth-first search to make the right selection at each branch point, then for each level j, on the average it will consider $(e_{dfs}-1)\sum_{k=1}^{d-j} b^k$ nodes in following fruitless paths before it selects a good child. This recurs at each of d levels, giving

$$t_{dfs} = (e_{dfs}-1)(\sum_{j=1}^{d-1}\sum_{k=1}^{d-j} b^k) + bd$$

as the expected number of nodes that depth-first search will generate before finding a solution. Furthermore, since $\sum_{k=1}^{d-j} b^k = (\frac{b}{b-1})(b^{d-j}-1)$, we can factor $\frac{b}{b-1}$ out of the remaining summation, giving

$$t_{dfs} = (e_{dfs}-1)(\frac{b}{b-1})(\sum_{j=1}^{d-1}(b^{d-j}-1)) + bd \quad .$$

One can rewrite the summation in this expression as

$$\begin{aligned}(\sum_{j=1}^{d-1} b^{d-j}) - (\sum_{j=1}^{d-1} 1) &= (\sum_{j=1}^{d-1} b^j) - (d-1)\\ = \frac{b^d-b}{b-1} - d + 1 &= \frac{b^d-1}{b-1} - d \quad .\end{aligned}$$

Substituting this into the second expression for t_{dfs}, we have

$$\begin{aligned} t_{dfs} &= (e_{dfs}-1)(\frac{b}{b-1})(\frac{b^d-1}{b-1} - d) + bd\\ &= (e_{dfs}-1)\cdot\frac{(b^{d+1}-db^2+db-b)}{(b-1)^2} + bd\end{aligned}$$

as the expected number of nodes the depth-first scheme will examine during its search. Note that e_{dfs} is much less significant than b in this expression.

The analysis for the iterative sampling algorithm is more straightforward. Let p be the probability of selecting a good node (one on a solution path) at a given branch point. We know that, if one is sampling with replacement, the expected number of attempts until one selects a good node is $1/p$. We refer to this expected value as e_{is} to distinguish it from that for depth-first search. The probability that the algorithm will solve the problem on any given pass is p^d, giving $1/p^d = e_{is}^d$ as the expected number of attempts before finding a solution. Finally, since there are exactly $b \cdot d$ nodes generated on each pass, we have $t_{is} = b \cdot d \cdot e_{is}^d$ as the expected number of nodes that iterative sampling will generate.

2.2 The Expected Number of Nodes

The above analysis referred to e, the expected number of nodes one must try at each branch point of the search tree before selecting one on a solution path. This number is simple to compute for problems with a single solution. For a systematic strategy like depth-first search, which samples states without replacement, we have $e_{dfs} = (b+1)/2$. In contrast, for a nonsystematic strategy that randomly samples states with replacement, we simply have $e_{is} = b$.

However, we are also concerned with problems that have multiple solutions, so that $e_{dfs} < (b+1)/2$ for the systematic case and $e_{is} < b$ for the nonsystematic one. Following Ginsberg and Harvey (1990), we will assume there exists some integer s such that, if a node lies on a solution path, exactly s of its b children also lie on a solution path. If a problem with branching factor b and depth d can be solved, then its initial state lies on such a path, and there will be exactly s^d solutions. These solutions will not be evenly distributed in the space, but will be 'clustered' near one another.

Because a nonsystematic strategy samples with replacement, the probability of selecting a good node is s/b on each pass. This gives $e_{is} = b/s$ as the expected number of nodes examined by iterative sampling at each branch point. This term can vary from b, for problems with a single solution, to 1, for problems in which all paths lead to a solution. Substituting for e_{is} in the earlier analysis gives $t_{is} = b \cdot d \cdot (\frac{b}{s})^d$ as the total number of nodes that we expect iterative sampling to generate before finding a solution.

Computing e_{dfs} for a systematic strategy is more complicated. Given a finite set of outcomes with values from 1 to n, the expected value is simply $\sum_{k=1}^{n} kp_k$. Since depth-first search samples without replacement, we know that it will never select more than $b - s + 1$ nodes at any branch point. If we let p_n be the probability of selecting a good node on the n_{th} attempt, then p_1 is simply s/b. However, to compute later terms we must multiply the probability of failure on the previous passes by the probability of success on the current one. Each time the algorithm fails to select a node on a solution path, there is one fewer child in competition. Thus, we have $(1-\frac{s}{b})(\frac{s}{b-1})$ for p_2, $(1-\frac{s}{b})(1-\frac{s}{b-1})(\frac{s}{b-2})$ for p_3, and so forth.

Multiplying these terms by the respective values of k, we get the general expression

$$\begin{aligned} e_{dfs} &= \frac{s}{b} + \sum_{k=2}^{b-s+1} \left[k\frac{s}{b-k+1} \prod_{j=0}^{k-2} (1 - \frac{s}{b-j}) \right] \\ &= \frac{(b-s)}{(s+1)} + 1 \end{aligned}$$

as the expected number of nodes examined by a systematic method at each branch point before finding one on a solution path. This reduces to $(b+1)/2$ in the case where $s = 1$ and there is a unique solution. Substituting for e_{dfs} in the earlier analysis gives

$$t_{dfs} = \frac{(b-s)}{(s+1)} \cdot \frac{(b^{d+1} - db^2 + db - b)}{(b-1)^2} + bd$$

as the total number of nodes that we expect depth-first search to generate before finding a solution.

3. Behavior of the Search Algorithms

The above analysis makes specific predictions about the amount of search required by the random and depth-first algorithms under various conditions. In particular, it predicts the effects of the branching factor, the solution depth, and the number of solutions. For any search task within the space analyzed, comparing t_{dfs} to t_{is} will let us predict the more efficient method. However, to gain better intuitions about the relative costs of the algorithms, we will examine their behavior across a range of settings for the three domain parameters.

Figure 1 (a) shows the predicted behavior of the two search algorithms for ten settings of the depth d when only one solution exists and when the branching factor b is two. The dependent measure is the number of nodes generated before finding a solution, presented in logarithmic scale to counter the exponential growth inherent in search.

The figure also shows experimental results obtained for the two methods using artificial search problems. We implemented versions of iterative sampling and depth-first search that accepted parameters for the breadth b, depth d, and good children s, and that explored search trees with these characteristics. If a node selected by an algorithm lay along a solution path, then s out of b of its children also lay along solution paths. The nonsystematic method selected one child at random, whereas the depth-first technique ordered them randomly and considered each in turn.

Each experimental point in Figure 1 (a) represents an average of 1000 runs for one of the algorithms on a specific setting of the parameters b, d, and s. Note that the theoretical values typically fall within the 99% error bars that bound the observed means. The close fit between the predictions and observations lends confidence that the overall analysis is correct, though computational constraints limited our experimental runs to problems that involved under 300,000 nodes.

The basic results are somewhat discouraging. In the absence of multiple solutions, depth-first search always outperforms iterative sampling. The systematic nature of the former algorithm, which reduces the expected number of nodes one must consider at each branch point, more than compensates for the extra search it carries out when it strays down a useless path. At first glance, this suggests that our intuitions were incorrect, and that one should prefer systematic methods to nonsystematic ones.

However, the curves in Figure 1 (b) reveal a quite different story when multiple solutions are present. When the branching factor is four and two children at each branch point lead to solutions, we again find that iterative sampling requires more search than the depth-first scheme on problems in which the depth d is low. But as d increases, the difference between the two methods decreases, until their curves cross over each other near $d = 7$. Beyond this point, random search outperforms depth-first search, and its relative advantage increases with the depth. Recall that the curves present cost on a logarithmic scale; thus, for $d = 12$, the analysis predicts that depth-first search will take 20 times as long as iterative sampling. In the presence of sufficient numbers of solutions, nonsystematic search scales to longer solution paths much better than the systematic technique. The experimental points in the figure are consistent with these predictions.

The results change again when we hold the depth and solutions constant and vary the branching factor b. Figure 2 (a) shows that the systematic method initially fares slightly worse than the nonsystematic one, but as b increases, depth-first search pulls slightly ahead. When the depth is five and each good node has three good children, the crossover occurs when the branching factor is around ten. After this point, the depth-first algorithm requires less search than iterative sampling, and its advantage increases with the setting for b. Thus, the systematic method scales better to problems with higher branching factors than does it-

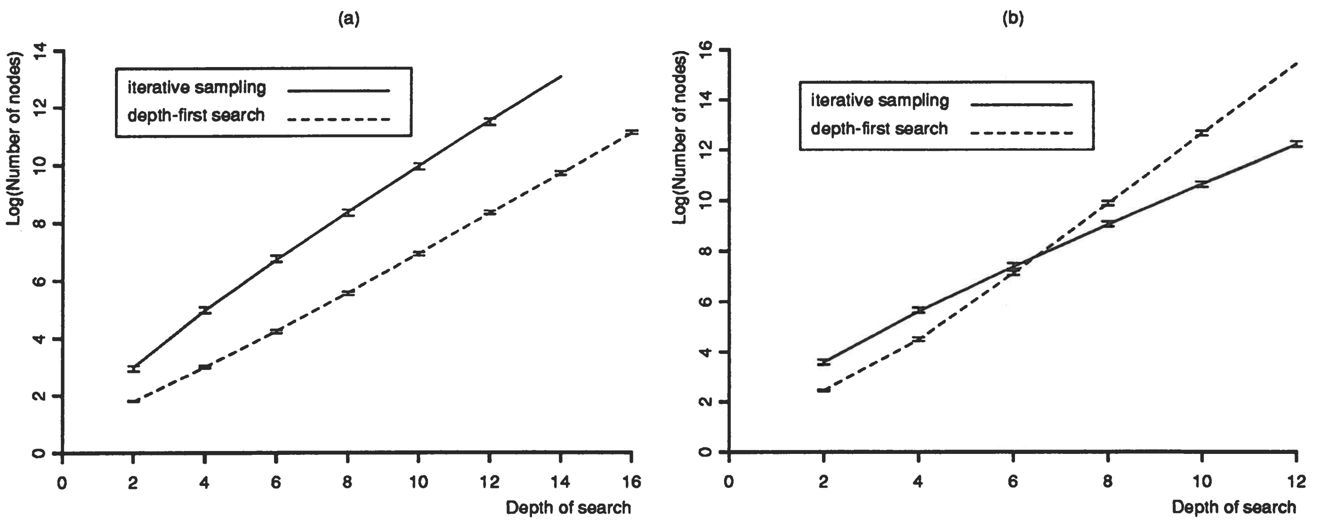

Figure 1: Nodes generated by depth-first search and iterative sampling for different depths d (a) when there is a unique solution and the branching factor b is two, and (b) when the number of good nodes s is two and the branching factor b is four. The lines represent theoretical behavior; the error bars indicate experimental results.

erative sampling, but the difference is small compared to other factors we have examined. In fact, the experimental points in the graph are so close that the difference is barely detectable. Interestingly, nearly indistinguishable behavior also occurred for a number of other settings for d and s that we examined.

Both crossovers occur in the presence of multiple solutions, but not when it is absent, suggesting that we examine the role of this factor more closely. Figure 2 (b) reveals the effect of varying s, the number of children on a solution path at each branch point, given a branching factor of six and a solution depth of five. For low settings of s, iterative sampling carries out substantially more search than the depth-first approach. However, this difference decreases as the number of solutions increases, and around $s = 3$ the curves intersect. Past this point, the nonsystematic method outperforms the systematic one, except in the degenerate case where $s = b$, where neither requires any search. Thus, iterative sampling benefits more from multiple solutions than does depth-first search, making the former more desirable in domains where they are present.

This result suggests an alternative treatment of the effect of the branching factor b. In some classes of domains, as one increases the branching factor, the number of solutions increases as well. To model this situation, we examined the effect when we varied b but held the depth d at four and held the ratio b/s at the constant two. Figure 3 (a) shows the curves that result. Under these conditions, we again find that iterative sampling starts off requiring more search than the depth-first organization, but that a crossover occurs near $b = 8$ or $s = 4$, and that after this point, the nonsystematic method outperforms the systematic one. Thus, iterative sampling scales to higher branching factors better than does depth-first search, provided that the number of good nodes at each branch point keeps pace, and thus the density of solutions remains constant. Whether there exist real-world tasks with this characteristic remains an open question.

In comparing depth-first search to their iterative broadening method, Ginsberg and Harvey (1990) examine behavior as the depth d becomes very large. In a similar manner, we can compare systematic and nonsystematic search in the 'large depth limit'. To determine the conditions under which iterative sampling will outperform depth-first search, we set $t_{is} < t_{dfs}$ and solve for s. As Ginsberg and Harvey note, one can simplify t_{dfs} for large d, giving us the inequality

$$b \cdot d \cdot \left(\frac{b}{s}\right)^d < \frac{(b-s)}{(s+1)} \cdot \frac{b^{d+1}}{(b-1)^2} \quad .$$

Dividing both sides of this expression by b^{d+1} produces

$$\frac{d}{s^d} < \frac{(b-s)}{(s+1)(b-1)^2} \quad ,$$

and taking both sides to the limit as d approaches infinity gives the inequality

$$0 < \frac{(b-s)}{(s+1)(b-1)^2} \quad ,$$

provided $s > 1$, since in this case the denominator s^d dominates the numerator d. Finally, adding to both sides and then dividing gives the relation

$$s < b \quad .$$

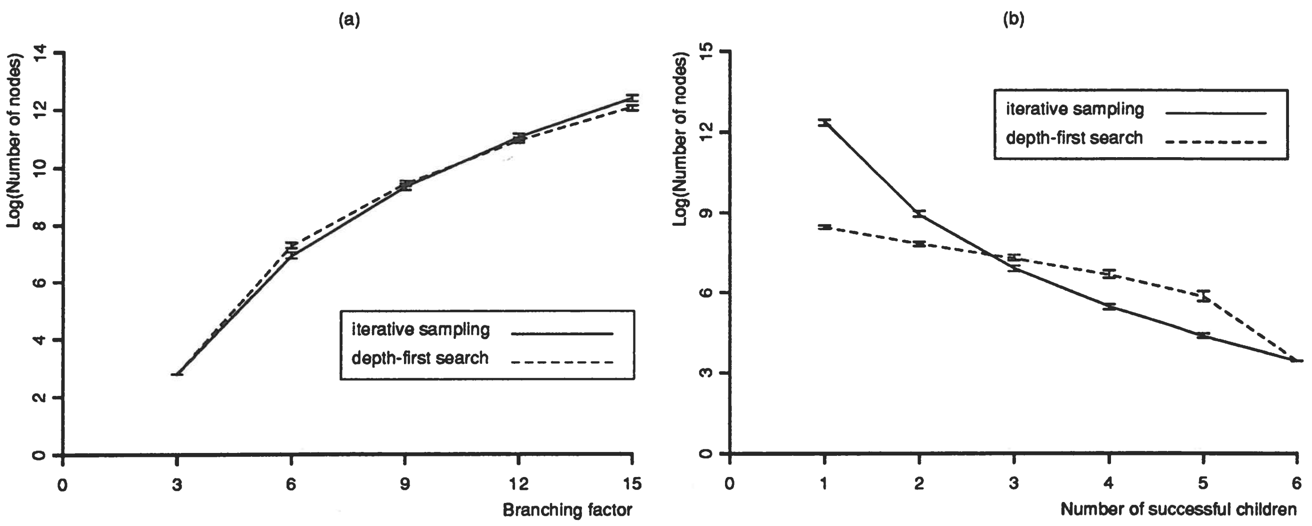

Figure 2: Predicted and observed search for the depth-first and iterative sampling algorithms (a) on different branching factors b when the number of good nodes s is three and the depth of the solution d is five, and (b) for different numbers of good nodes s when the branching factor b is six and the depth of the solution d is five.

In other words, in the large depth limit, iterative sampling requires less effort to find a solution than depth-first search whenever the number of good children s is greater than one and less than the branching factor b.

This result is similar to the one that Ginsberg and Harvey report for their iterative broadening method, which outperforms depth-first search for large depths whenever $s^d > 2(b-1)/(b-2)$. One conclusion they draw is that when $b > 3$, their method will be more efficient (for large d) even when there exist only three terminal goal nodes. However, this result follows from an extension of their analysis to nonintegral values of s between one and two. This range of values is crucial, since it seems unlikely that many real-world domains have 2^d or more solutions.

Our equations do not cover the nonintegral case for iterative sampling, but we have carried out experimental studies of this condition. Figure 3 (b) presents curves for systematic and nonsystematic search when $b = 2$ and the expected value of $s = 1.5$. In this experiment, half of the nodes along solution paths had one child that also led toward a solution ($s = 1$) and half had two such children ($s = 2$). In general, nonintegral s values can follow other distributions, but this one is analogous to that in Ginsberg and Harvey's analysis. As for integral values, the random method starts out worse than depth-first search, but eventually a crossover occurs, in this situation around $d = 10$. Additional experiments show that this effect recurs for other nonintegral s values.

Although our analysis has focused on a specific class of domains, we hoped that iterative sampling would have advantages in other search tasks as well. To this end, we compared the behavior of iterative sampling with that of depth-first search on the eight puzzle. One can reach nodes in the eight puzzle through many paths, which clearly violates our assumption about tree structure. In one experiment, we ran both algorithms on problems that could be solved in a minimum of four steps, but we varied the depth limit used to control backtracking and iteration. The idea here was that, lacking knowledge of minimal solution length, one might overestimate this factor. Under these conditions, we found that depth-first search carried out progressively more search as one increases the depth limit, whereas random search was nearly unaffected by the parameter. In this study, crossover occurred when the depth limit exceeded ten. However, in another experiment with the eight puzzle, we varied the length of the minimal solution path while setting the depth cutoff equal to this minimum. In this case, depth-first search consistently outperformed iterative sampling even for problems with 20 steps in their solutions.

In summary, depth-first search is clearly superior to iterative sampling in the absence of multiple solutions, but the case is much less clear cut when there exists more than one successful path. For tasks that involve shallow solution paths and few solutions, depth-first search remains the method of choice. However, the random algorithm scales better to domains that involve deep searches, and it receives greater benefits from multiple solutions. The situation with respect to the branching factor is even more complex. The systematic technique scales better to problems with high branching factors when the number of solutions remains constant, but the nonsystematic method scales better when the density of solutions remains the same.

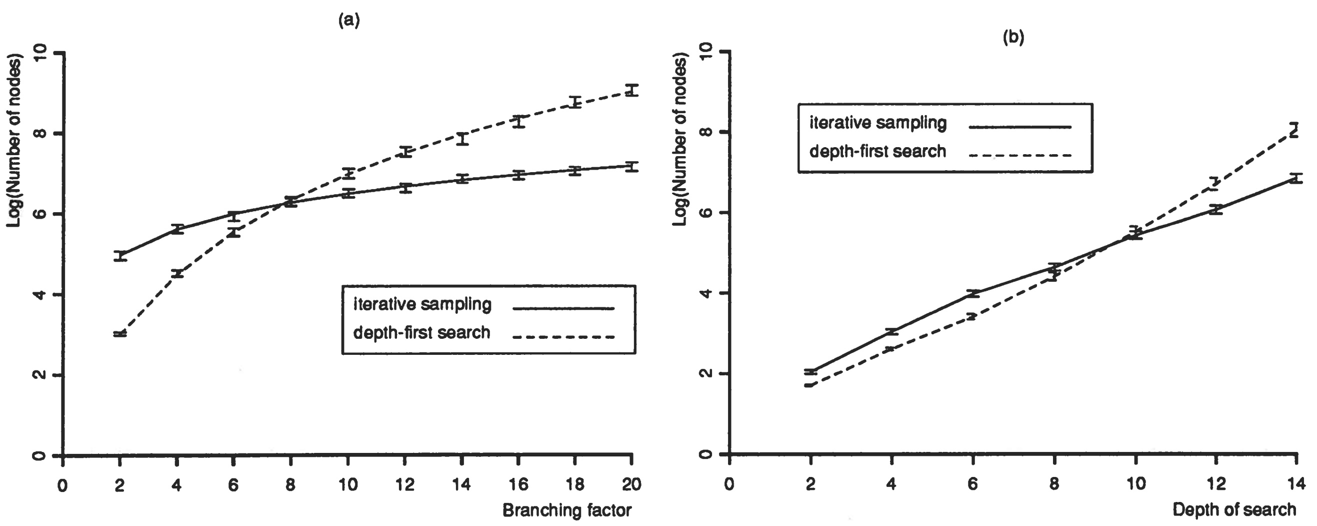

Figure 3: (a) Predicted and observed search for the depth-first and iterative sampling algorithms for different branching factors b when the depth of the solution d is four and the ratio b/s is two; and (b) the observed cost for depth-first search and iterative sampling when b is two and s is 1.5.

Thus, the preferred method depends on the characteristics of the problem-solving domain, but there exist clear cases where random search is more desirable.

4. Discussion

Our work is not the first on nonsystematic search, and we hope it will not be the last to explore the potential of this intriguing class of methods. Here we briefly review some related work and discuss open issues for future research.

4.1 Relation to Iterative Broadening

Our intuitions about the superiority of nonsystematic search over depth-first search are similar to Ginsberg and Harvey's (1990) arguments for the advantages of their *iterative broadening* technique. This method first carries out a depth-first search with a breadth limit of two. If this fails, it repeats the search with a breadth limit of three, and so on, until it finds a solution. As in iterative sampling, the iterative broadening method can repeat previous effort, but it does not get locked into expanding entire subtrees because of errors made early in its search. It differs from the iterative sampling algorithm in its systematic nature, but it also appears to benefit from situations in which multiple solutions are present.

Ginsberg and Harvey's analysis of iterative broadening lets one calculate its expected performance on particular search tasks. The results one obtains in this manner suggest that, like depth-first search, iterative broadening outperforms random search initially, but that as the depth increases, the nonsystematic approach fares better. The theoretical crossover point for iterative broadening and random search (around $d = 12$ for $b = 4$ and $s = 2$) occurs later than that for depth-first and random search, but otherwise the effects of solution depth are very similar, presumably because Ginsberg and Harvey's method sometimes expands entire breadth-limited subtrees.

Experiments with artificial search tasks are consistent with these predictions. Figure 4 (a) presents the results for $b = 4$ and $s = 2$, which suggest that the crossover does indeed occur near $d = 12$. Figure 4 (b) shows that similar curves occur in the nonintegral case when $b = 2$ and $s = 1.5$. However, other experiments indicate that iterative broadening is much less affected by higher branching factors than iterative sampling, and that it benefits from higher values of s as well as the nonsystematic approach. In summary, iterative broadening does not scale to large depths as well as does random search, but it outperforms iterative sampling on other dimensions and in general is much harder to beat than depth-first search.

4.2 Related Studies of Random Search

A number of researchers have reported methods that incorporate some notion of random search. For instance, Brassard and Bratley (1988) described a set of nonsystematic techniques that they called *Las Vegas* algorithms. One variant, which they applied to the eight queens puzzle, is almost identical to the iterative sampling algorithm. They reported experiments showing that their method outperforms depth-first search on the queens puzzle, but they did not systematically explore the effect of the branching factor, search depth, or number of solutions.

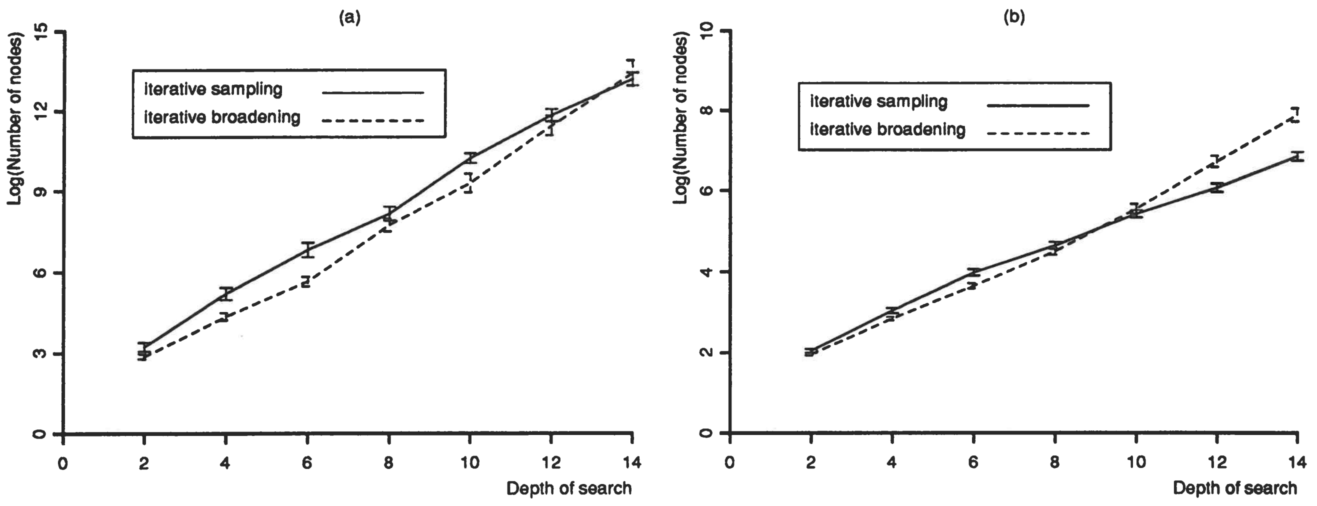

Figure 4: (a) Observed search for iterative broadening and iterative sampling for different depths d when the branching factor b is four and s is two; and (b) the observed search for the two algorithms when b is two and s is 1.5. These curves do not include theoretical costs.

On the other hand, unlike our analysis, Brassard and Bratley's treatment included separate costs for success and failure. This takes into account the chance that one will abandon a path before reaching the depth d, which seems quite possible in constraint satisfaction domains where one can detect illegal partial states. They also showed that a combination of systematic and nonsystematic search can outperform either method in isolation, at least in the queens domain. In closely related work, Minton, Johnston, Philips, and Laird (1990) have also collected experimental evidence that iterative sampling outperforms the depth-first approach on the queens task. Their explanation of this result hinged on the presence of multiple solutions that were clustered in the search space; this is analogous to our assumption that s is greater than one.

Janakiram, Agrawal, and Mehrotra (1988) have applied similar ideas to parallel search methods. They retained the idea of systematic depth-first search, but to minimize redundancy on parallel processors, they ordered children randomly at each choice point. For some classes of problems, they obtained speedup that was linear in the number of processors. More important from our perspective, they analyzed and experimenally studied a number of different problem types, including ones in which solutions were clustered and others in which they followed a uniform distribution. However, they did not examine nonsystematic methods like iterative sampling, despite their suitability for parallel processing.

4.3 Nonsystematic Search in Planning

Findings in the psychology of human planning and problem solving are also relevant to our work. For example, De Groot (1965) has reported that chess players rely on *progressive deepening*, a search method in which they repeatedly return to the initial state. From this base position they explore a single path at ever greater depths, considering side branches only briefly before returning to the main path. Newell and Simon (1972) observed similar behavior in their subjects, and thus incorporated progressive deepening into their theory of human problem solving. There is no clear evidence that chess players use random selection, but they do favor an iterative greedy approach.

More recently, Jones (1989) has described EUREKA, a model of human problem solving that used a variant of iterative sampling to control a means-ends planner. The system probabilistically selected one state to expand at each branch point in its search, but if this path did not produce a solution by a prespecified depth, it returned to the initial state and began again from there. This continued until the model found a solution or gave up after a number of passes. EUREKA also used a heuristic to bias selection in favor of some states and incorporated a simple form of reinforcement learning that let it improve this selection process with experience.

Minton, Drummond, Bresina, and Philips (1992) have also used iterative sampling to generate plans. They examined interactions between the search strategy and the nature of the search space, finding the construction of partial order plans was more efficient than generation of total order plans when using depth-first search, but that this difference disappeared when using iterative sampling. From these experimental results, they concluded that solutions for their planning tasks (the blocks world) were clustered, lending plausibility to this assumption in our analysis. Although depth-first search outperformed iterative sampling for the problems that Minton et al. studied, they also found that a simple heuristic – preferring the partial plan with the fewest unmatched preconditions – completely reversed this result.

4.4 Directions for Future Work

Our analysis of nonsystematic search also makes a variety of assumptions and omissions that we should address in further research. In particular, our model assumes that the branching factor, depth of search, and number of good nodes at each branch point are constant. We should run experiments to examine the behavior of the algorithms when b, d, and s vary across nodes and branches. Hopefully, the analysis will remain a good predictor when using the average values of these parameters rather than constant ones. Our experimental study of nonintegral s values provides a reasonable start, but we must examine other forms of variation as well. We should also extend the analysis beyond trees to include search through graphs.

As we have noted, heuristics feature prominantly in the work by Jones and by Minton et al., and we should examine their role more carefully in the future. Preliminary analyses suggest that, for certain definitions, iterative sampling benefits more from heuristic knowledge than does depth-first search. This hypothesis is consistent with Minton et al.'s results on planning tasks. Moreover, it makes intuitive sense in that heuristics reduce e, the expected number of nodes examined at each branch point, which has a greater effect on iterative sampling than on depth-first search. However, one can imagine different formulations of heuristic information, and an extended analysis should use one that reflects characteristics of actual heuristics.

Iterative sampling bears a strong similarity to reactive methods for interacting with the external environment, in that the latter select one of many actions and cannot backtrack. This connection suggests that nonsystematic planning methods may be easier to integrate with reactive executors, as Drummond (personal communication, 1992) has noted that the former generate the same probability distribution of behaviors as many reactive systems. The similarity of iterative sampling to reactive methods should also let one directly apply techniques for reinforcement learning to acquire heuristic knowledge from successful and unsuccessful iterations.

Despite the research issues that remain, the results to date demonstrate that, in some circumstances, nonsystematic methods like iterative sampling can solve problems much more efficiently than systematic techniques, including depth-first search and iterative broadening. Moreover, the nonsystematic approach appears to scale better on some dimensions of difficulty and seems better able to take advantage of multiple solutions. These results cast a different light on the nonsystematic nature of human problem solving, suggesting that this behavior is more adaptive than it appears. Future work should further clarify the conditions under which random search is desirable, examine the effect and acquisition of heuristic knowledge, and evaluate the approach on real-world problem-solving and planning tasks.

Acknowledgements

Thanks to S. Minton, G. Iba, P. Andreae, P. Laird, A. Philips, W. Iba, J. Allen, S. Ohlsson, R. Jones, and M. Drummond for discussions that refined many of the ideas in this paper. Thanks also to J. Allen, M. Ginsberg, R. Korf, and the reviewers for insightful comments on earlier drafts.

References

Brassard, G., & Bratley, P. (1988). *Algorithmics: Theory and practice.* Englewood Cliffs, NJ: Prentice Hall.

de Groot, A. D. (1965). *Thought and choice in chess.* The Hague: Mouton.

Ginsberg, M. L., & Harvey, W. D. (1990). Iterative broadening. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 216–220). Boston, MA: MIT Press.

Janakiram, V. K., Agrawal, D. P., & Mehrotra, R. (1988). A randomized parallel backtracking algorithm. *IEEE Transactions on Computers*, *37*, 1665–1676.

Jones, R. (1989). *A model of retrieval in problem solving.* Dissertation, Department of Information & Computer Science, University of California, Irvine.

Korf, R. E. (1985). Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, *27*, 97–109.

Minton, S., Johnston, M. D., Philips, A. B., & Laird, P. (1991). Solving large-scale constraint satisfaction and scheduling problems using a heuristic repair method. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 17–24). Boston, MA: MIT Press.

Minton, S. N., Drummond, M. E., Bresina, J. L., & Philips, A. B. (1992). *Total order vs. partial order planning: Factors influencing performance.* Unpublished manuscript, Artificial Intelligence Research Branch, NASA Ames Research Center, Moffett Field, CA.

Newell, A., & Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice Hall.