ACCURACY AND SAVINGS IN DEPTH-LIMITED CAPTURE SEARCH

Prakash Bettadapur
T. A. Marsland

Computing Science Department
University of Alberta
Edmonton
Canada T6G 2H1

ABSTRACT

Capture search, an expensive part of any chess program, is conducted at every leaf node of the approximating game tree. Often an exhaustive capture search is not feasible, and yet limiting the search depth compromises the result. Our experiments confirm that for chess a deeper search results in less error, and show that a shallow search does not provide significant savings. It is therefore better to do an arbitrary depth capture search. If a limit is used for search termination, an odd depth is preferable.

Final copy for the Int. J. of Man-Machine Studies

submitted Dec. 1986
accepted July 1987

4 March 2013

ACCURACY AND SAVINGS IN DEPTH-LIMITED CAPTURE SEARCH

Prakash Bettadapur
T. A. Marsland

Computing Science Department
University of Alberta
Edmonton
Canada T6G 2H1

1. INTRODUCTION

The complete tree for chess consists of about 10^{43} positions, so searching it to find an exact solution is not feasible. In practice, game playing programs search to a limiting horizon to find an approximate solution within a specified time. Restricting the look-ahead to a particular depth introduces some error in the search procedure. The purpose of our work here is to study an essential component of game-tree search, namely the subtree of capture moves that is expanded as part of the "terminal" or horizon node evaluation of most computer-chess programs. This topic has not been covered before, not even in the recent comprehensive review (Marsland, 1986) of search and pruning methods used in chess programs.

Pearl (1985) states that the dual notions of *visibility* and *filtering* are usually used to support look-ahead. At deep nodes in the game tree, the game is more evident, evaluations are more exact and the *visibility* is better. His *filtering* argument states that since a deeper search involves a look through more nodes, the backup value is therefore integrated over more features and so is better informed. Nau (1980) and Pearl (1985) have proved that for some games this argument is fallacious, but the type of trees that degrade the solution on deeper search are considered pathological. Despite the attempts by Beal (1980, 1982) and Bratko & Gams (1982) at an error analysis of the minimax search it has not yet been proved that chess is anomaly-free, even though all the experimental evidence supports such a view.

Capture search, an expensive part of any chess program, is conducted at every leaf node of a depth-limited game tree. The value of the subtrees that extend beyond the horizon are estimated by restricting the selection of moves that are considered. Even here an exhaustive search may not be possible. For example, a capture tree with six captures from the initial position, but with reduced captures at each successive level, has about 5×10^5 nodes, provided there are no cut-offs. The minimal-error look-ahead argument for visibility requires that the search be continued without limit. Since capture searches are both frequent and expensive, it is desirable to terminate the search as soon as an acceptable result is known. More complex criteria than depth can be used to limit the search, but to assess their worth one must first quantify the error and savings obtained on searches to different depths.

2. EXPERIMENTAL STUDY

A simple chess program was modified to do depth experiments on chess positions that derive from the basic test suite by Bratko & Kopec (1982), and from another group of twenty-five positions (Kopec, Newborn & Yu, 1985). Two-ply and 3-ply *alpha-beta* searches were done on these two sets, yielding over 35000 horizon nodes from the Bratko-Kopec suite to produce SET1, and about 45000 nodes from the second group to form SET2. These roughly 80000 non-quiescent terminal positions formed the problem set for the experiments reported here. In both theoretical and experimental studies it is normal to compare tree-search algorithms on the basis of how many bottom positions are visited, as Marsland (1986) has done. This measure is used because the cost of bottom positions is dominant, since a full evaluation (commonly including a capture search) is needed there. On the other hand, in a capture search the terminal nodes are relatively few because the tree is tapered, and are cheaper than interior nodes because there are no more capture moves to generate. For that reason a count of the total nodes in a capture tree is taken as a measure of the search cost.

For our experiments $depth = 16$ was taken as the basis of comparison because Bettadapur (1986) has shown that unlimited-depth capture searches yield only marginally better results. At each of the 80000 terminal nodes a capture search was carried out to depths ranging from 2 to 10 ply. Whenever the search returned a value different from that of $depth = 16$, it was recorded as an error. The percentage error is the ratio of how often a different value is returned, to the total terminal positions searched. Similarly, a shallow search visits fewer nodes in the capture tree, resulting in savings. The percentage savings is the ratio of reduction of capture nodes at varying depths, to the total capture nodes in the tree when the depth is sixteen.

2.1. Depth and Accuracy

Figure 1 plots the percentage error in capture searches of various depths. As explained by the *visibility* argument, a deeper search gives more accurate results. Nodes that are deep in the search tree have fewer complex captures possible; hence they can be evaluated more precisely. As search depth is increased, many more nodes become quiescent, so that correct evaluation is done at an increasing number of nodes. Minimaxing more reliable values through the tree increases the accuracy at the root node by the *filtering* argument.

[Figures appear at end of document]

Figure 1. Change in Capture Search Error with Depth.

Some programs adopt a static evaluation of the terminal positions instead of a capture search. However, a static analysis may not consider the side effects of captures effectively. For example, a pinned piece may be identified only when a capture is made. Therefore, it is argued that a static analysis could be inaccurate, and thereby account for some apparent blunders in computer chess games. Figure 1 shows that even if the static analysis is equivalent to a full four ply capture search, the error can be as high as 5.5%.

2.2. Tree Size Reduction

Current chess programs have arbitrary termination criteria. Some programs stop early at four or eight ply into a capture search, others continue as long as exchanges are possible. It is not clear what depth limit to apply, whether the savings decline after a certain depth, or whether there is an optimum depth with increased savings but with no increase in error. To clarify these issues, consider the savings in the number of searched nodes for various depths, as plotted in Figure 2. Since capture trees are tapered, the savings measure cannot be applied to game trees. A search to $depth = 7$ saves about 15 to 20% with an error of about 0.5%. Thus, if 10000 capture nodes are examined from a typical terminal position, about 50 evaluations will be faulty and may lead to a wrong value at the root node of a game tree. Bettadapur (1986) has concluded that there is no optimum depth with maximum savings and low error. Figure 2 shows that the SET2 positions are generally more complex and lead to larger capture trees, thus resulting in bigger savings when the search is terminated prematurely.

[Figures appear at end of document]

Figure 2. Change in Nodes Omitted with Depth.

2.3. Search to Odd or Even Ply

Deeper search results in increased *visibility*, but this increase is non-uniform over odd and even ply. With an odd ply search, the side that makes the first move also makes the last, so more information is available for that side, and hence less error. This result is substantiated by the plot in Figure 3, which is for SET1 positions only. The dashed line represents the odd depth results, whereas the solid line is for even depths. The plot of percentage error to percentage savings shows that the odd depth search has consistently lower error values for a given savings, and greater savings for a given error. Thus, for a two-person, zero-sum game, it is better to end search at an odd ply.

[Figures appear at end of document]

Figure 3. Relationship between Error and Savings with Depth.

3. DISCUSSION

In pathological games, where searching deeper consistently degrades the solution quality, it may be better to accept the node evaluation reduction that comes from a shallow search. In capture search, however, there is no such degradation, as the error experiments show, and this supports the view that chess is a non-pathological game. Statistically, a deeper search is always better, and so capture search does not fall into the ensemble of games defined by the standard probabilistic model (Pearl 1985). That model uses the filtering argument to conclude that if the evaluation accuracy remains the same, even for deeper nodes, the final solution is degraded. However, if evaluation is more correct because of increased visibility, the game overcomes any pathological behaviour. As Nau (1981) points out, there are games that have increased visibility, but not enough to overcome pathology. However, with capture search there is no anomaly since terminal nodes are quiescent with respect to captures, and so are evaluated precisely.

The error measurements show that increasing the search depth results in improved visibility, and hence less error. Further, only about a 50% reduction in nodes searched is possible, even for shallow searches where the error is high. Our experiments show that every small increase in search effort gives more accurate results, but at least 9-ply is needed to reduce the error to statistical insignificance. Figure 3 shows that if a depth limit must be used, then it is better to stop at an odd-ply, since for any specified error limit the savings will be greater. Finally, the positive effect of visibility is much higher than the negative effect of filtering, therefore a capture analysis is not pathological, since it yields better results as the search depth increases.

Acknowledgements

We thank Dr. Jonathan Schaeffer for his insights and his helpful advice during the formative stages of this research. Financial support was provided through Canadian Natural Sciences and Engineering Research Council Grant A7902.

References

Beal, D.F. (1980).

An Analysis of Minimax. In Clarke, M.R.B. Ed., *Advances in Computer Chess 2,* pp. 103-109. Edinburgh: Edinburgh Univ. Press.

Beal, D.F. (1982).

Benefits of Minimax Search. In Clarke, M.R.B. Ed., *Advances in Computer Chess 3,* pp. 17-24. Oxford: Pergamon Press.

Bettadapur, Prakash (1986).

Experiments in Chess Capture Search, M.Sc. thesis, Computing Science, University of Alberta, Edmonton.

Bratko, I. & Gams, M. (1982).

Error Analysis of the Minimax Principle. In Clarke, M.R.B. Ed., *Advances in Computer Chess 3,* pp. 1-16. Oxford: Pergamon Press.

Kopec, D. & Bratko, I. (1982).

The Bratko-Kopec Experiment: A Comparison of Human and Computer Performance in Chess. In

Clarke, M.R.B. Ed., *Advances in Computer Chess 3,* pp. 57-72. Oxford: Pergamon Press.

Kopec, Danny, Newborn M.M. & Yu, Winston (1985).

Experiments in Chess Cognition, In Beal, D. Ed., *Advances in Computer Chess 4,* pp. 59-79. Oxford: Pergamon Press.

Marsland, T.A. (1986).

A Review of Game-Tree Pruning, *Intl. Comp. Chess Assoc. Journal,* **9(1)**, 3-19.

Nau, D.S. (1980).

Pathology in Game Trees: A Summary of Results, *Proceedings of the First National Conf. on Artificial Intelligence,* pp. 102-104.

Nau, D.S. (1981).

Pearl's Game is Pathological, Tech. Rep. 999, Computer Science Dept., Univ. of Maryland, College Park.

Pearl, Judea (1985).

Game Searching Theory: Survey of Recent Results. In Bramer, M.A. Ed., *Computer Game-Playing,* pp. 276-294. Chichester: Ellis Horwood.

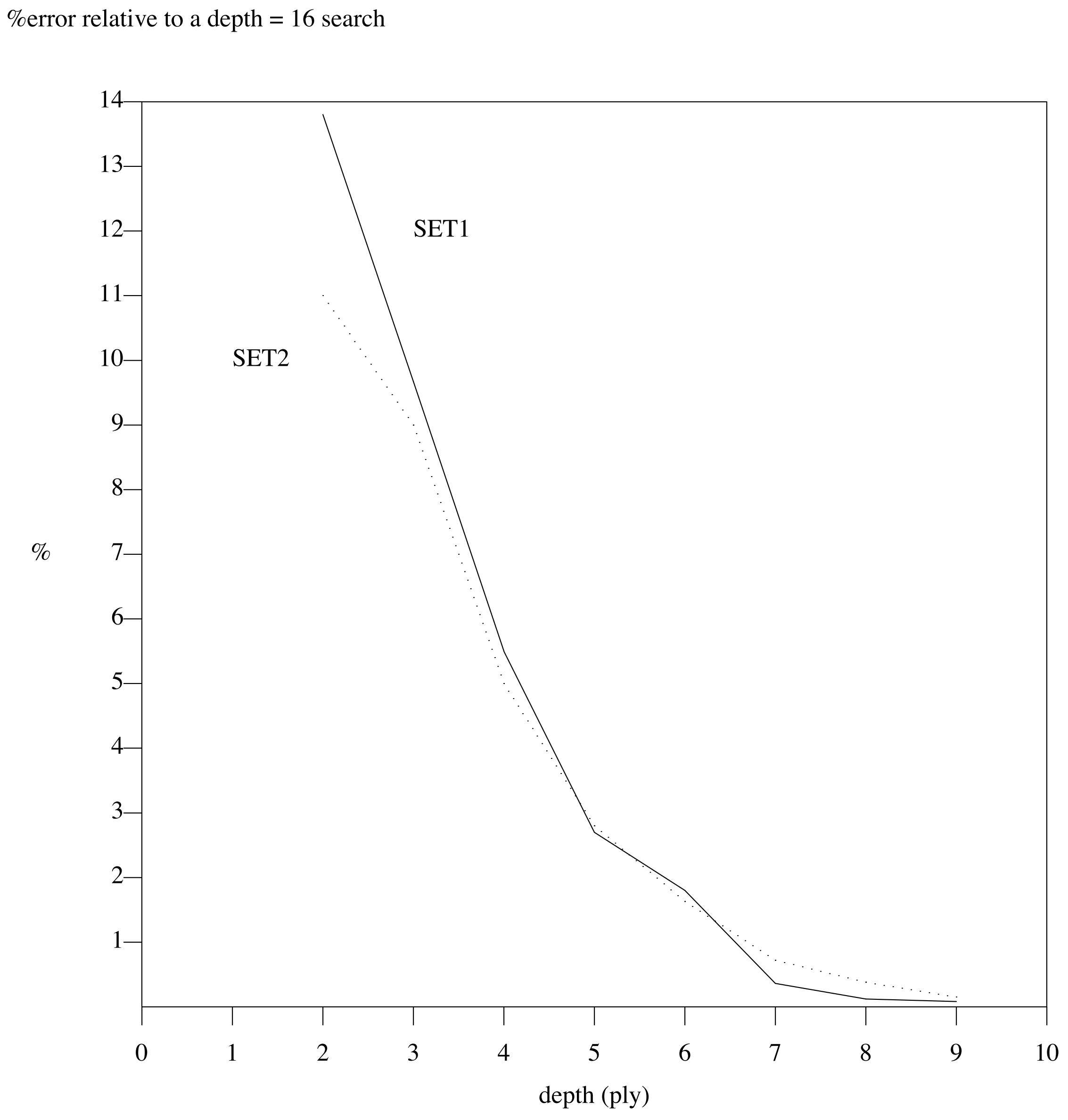

Figure 1. Change in Capture Search Error with Depth.

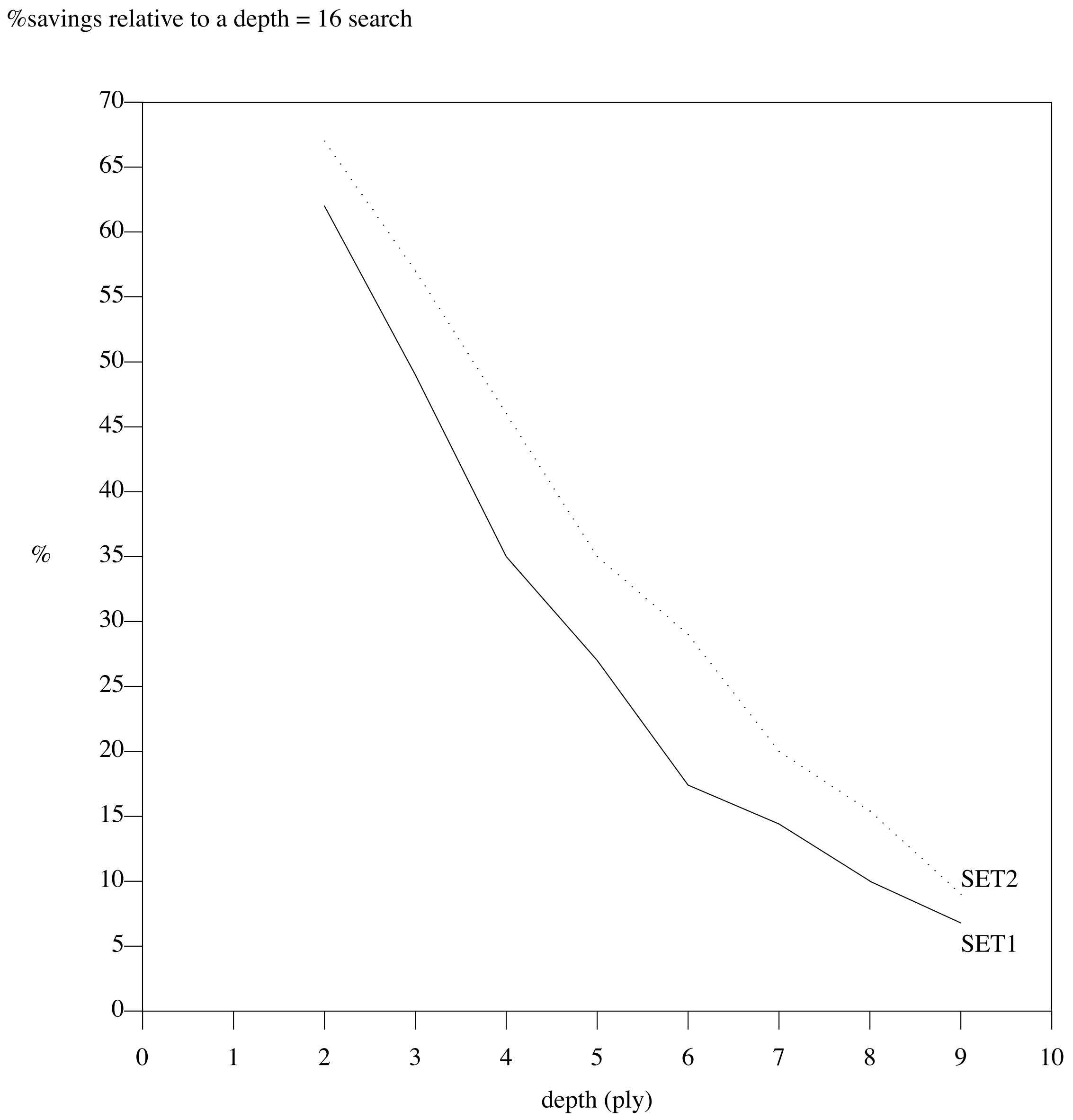

Figure 2. Change in Nodes Omitted with Depth.

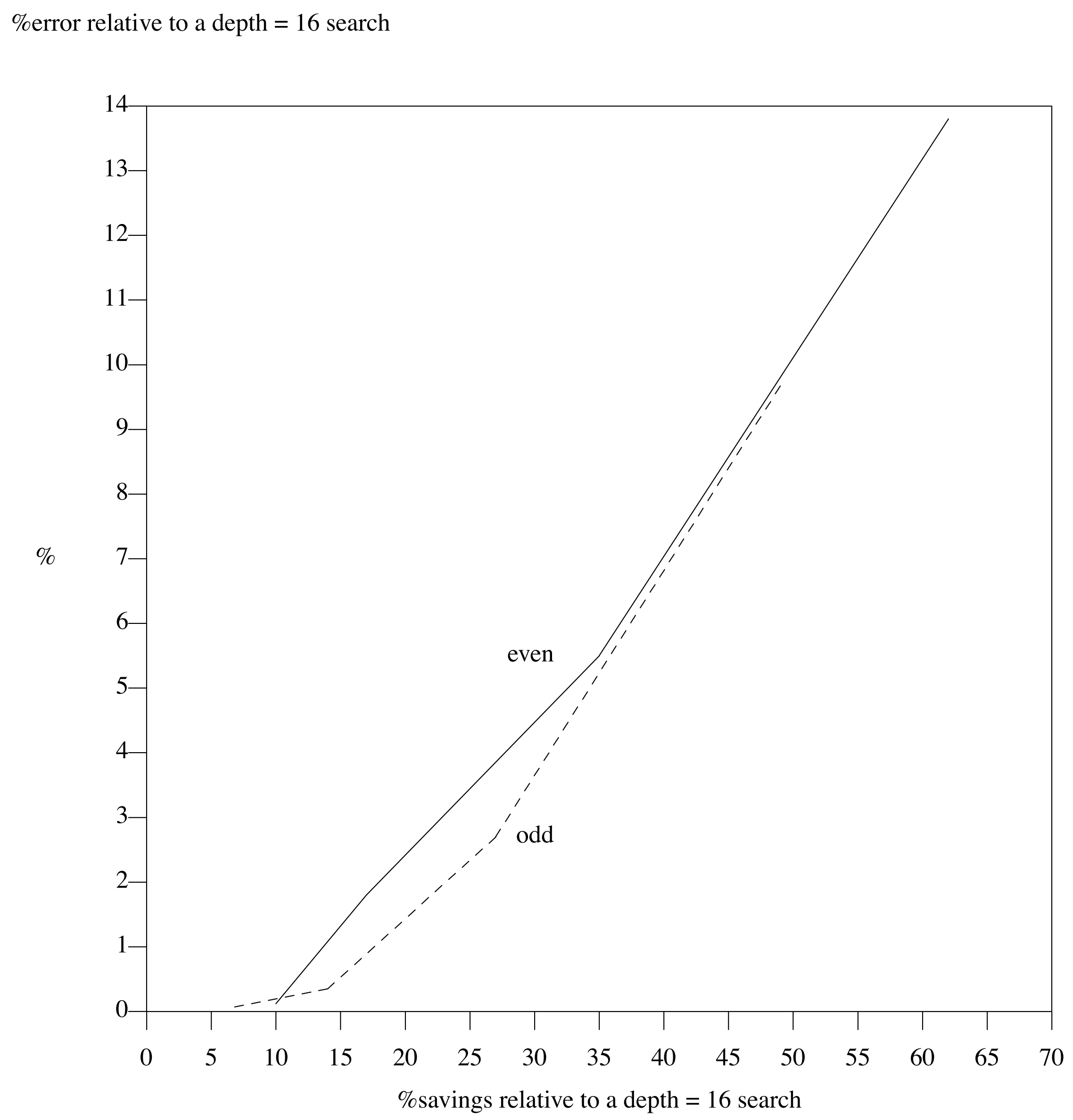

Figure 3. Relationship between Error and Savings with Depth.