

THE X-O-O HEURISTIC IN GAME TREE ANALYSIS

J. Denbigh Starkey
Computer Science Department
Washington State University
Pullman, Washington 99164

Programs for two-person games, such as chess or Go, are heavily dependent on Minimax evaluation of large game trees, which is combined with heuristics in an attempt to prune these trees as severely as possible. In particular, alpha-beta pruning, usually combined with the killer heuristic, is used in all major minimax-based programs. In this paper we shall propose a new heuristic which could be incorporated into minimax-based programs, and which should significantly increase the pruning possible during the analysis of game trees. This new heuristic, which can be combined with alpha-beta and the killer heuristic, is called the X-O-O heuristic. The development of the heuristic was prompted by the difficulty of analyzing large game trees in Go, and is based on a formalism and extension of the Go proverb "the enemy's play is my own key play".

1. Introduction.

Current programs for complex two person games such as chess or Go rely heavily on minimax-based lookahead to evaluate board positions and to select moves. In chess programs this is usually an analysis performed over good moves throughout the whole board, while in most Go situations lookahead must be restricted to local battles. In order to increase the number of nodes that can be analyzed with the computing resources available, alpha-beta tree pruning is invariably utilized, usually in conjunction with the killer heuristic [1,4].

In this paper we will introduce a new heuristic named the X-O-O heuristic. The purpose of this heuristic is to anticipate and protect against the opponent's strong moves, both offensive and defensive. In section 2 we will informally describe the heuristic, and will give examples of it in use in chess and Go. We will then define the algorithm for the heuristic precisely in section 3. We will assume that the reader knows the basic rules of chess and Go.

2. The X-O-O heuristic.

There is a Go proverb "The enemy's play is my own key play" [3]. A simple example of this is

shown in Figure 1.

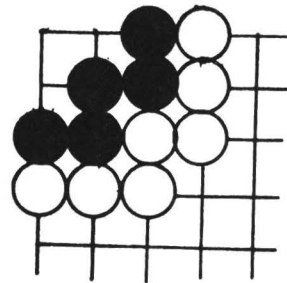


Figure 1

If it were black's move, he would play in the corner, guaranteeing the safety of his group. If it is white's move he uses the proverb to play in the corner. In its simplest form, the X-O-O heuristic is similar to the proverb in that it attempts to recognize the opponent's strongest move, and to take it away from him.

A chess example is shown in Figure 2.

BR						BK
				BP	BP	BP
		BP				
		WR		WR		
		WP	WP		WB	WB
				WP	WP	WP
						WK

Figure 2

It is White's move. In this situation current chess programs will perform extensive lookahead analysis attempting to find a mating sequence. In the same position, even a poor human player will quickly recognize the R-R8 threat, and will decide that the only way to block that move is R-R5, and so the block move is forced. No lookahead is necessary to make the move. The thought process that has occurred is:

"If it were black's move instead of mine, does he have any obvious strong moves? Yes, he would move his rook to R8 and check me. Would I then be able to do anything about that? No, it would be checkmate. How can I stop that move? I must either capture the rook, block its path, delay him by putting him in check, guard R1, or block the path between R1 and my king. The only one of these that I can do is to block the path of the rook with R-R5, and so this is a forced move."

Figure 3 shows a similar board position.

BR						BK
				BP	BP	BP
		BP				
	BP	WR		WR		
				WB		
	WP	WP			WB	
				WP	WP	WP
						WK

Figure 3

Here the threat is the same as before, but if black plays R-R8, white needs two moves to block the threat; either P-R3 or P-R4 followed by K-R3. The X-O-O heuristic compels white to play either P-R3 or P-R4 at this move.

Figure 4 shows a Go position called the nose tesuji. It is black's turn to play, and he wants to capture the two white stones on the right. A tesuji pattern matching program decides that the probable best play is the stone marked with an X, on top of the white "nose". Black must now confirm that the white pieces cannot escape. Tackett [2,5] analyzed this position, and decided that its solution would take nearly a month on an IBM 360/67, using standard lookahead techniques. Using the X-O-O heuristic the analysis is very inexpensive. White can only escape in two directions, to the left or right. Consider an escape to the right: if it were black's move he would immediately play to the right of the two white stones, blocking all possibility of escape. Therefore the X-O-O heuristic compels white to block that move by playing there. A simple template pattern match should now show that white is trapped. A similar analysis on the left shows that white cannot escape that way either, and so black's nose tesuji is confirmed.

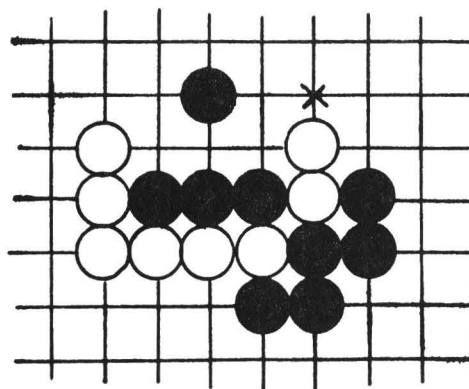


Figure 4

3. The X-O-O algorithm.

We are now ready to give the formal algorithm for the heuristic. Assume that the two players are X and O, and that it is O's turn to play.

The algorithm can then be written:

Find all very strong X moves, $X_1, \dots, X_n$.

For each X_i

- (a) If there is an O move which nullifies the effect of X_i then X_i can be ignored.
- (b) If X_i must be blocked now, add blocking O moves to the forced move list for X_i .
- (c) If two O moves are needed to nullify the effect of X_i , say O' and O'' , then add O' to the forced move list for X_i .

If there are no forced move lists, then use alpha-beta lookahead as usual.

If there are forced move lists, then play a move which is in the intersection of all of the forced move lists.

4. Summary

The success of the heuristic in any given game will depend on the care with which the good X moves are selected. The list of moves should be kept as small as possible, and n should usually be much smaller than the branching factor of the minimax game .ee. We are currently experimenting using the game of Go-Moku in order to obtain a good estimate of the improvements which can be expected using X-O-O.

REFERENCES

- [1] Aki S.G. and Newborn M.M. "The principal continuation and the killer heuristic." Proc. of ACM National Conference, (Seattle 1977), 466-478.
- [2] Benson D.B., Starkey J.D. et. al. "Computerizing the game of Go." Washington State Univ. Computer Science Dept. Tech. Report CS-76-031 (Pullman Wa 1976)
- [3] Davies J. "Tesuji." Ishi Press (Tokyo 1972).
- [4] Knuth D.E. and Moore R.W. "An analysis of alpha-beta pruning." Artificial Intelligence 6,4 (Winter 1975), 293-326.
- [5] Tackett M.J. "A pattern recognition approach to tesuji programming in Go." Proc. of ACM/CIPS Pacific Regional Symposium (Seattle 1976).