OPPONENT-MODEL SEARCH IN BAO: CONDITIONS FOR A SUCCESSFUL APPLICATION

H.H.L.M. Donkers, H.J. van den Herik, J.W.H.M. Uiterwijk
Institute for Knowledge and Agent Technology, Department of Computer Science, Universiteit Maastricht, P.O. Box 616, 6200 MD Maastricht, The Netherlands
{donkers,herik,uiterwijk}@cs.unimaas.nl, http://www.cs.unimaas.nl/~donkers/

Abstract Opponent-Model search is a game-tree search method that explicitly uses knowledge of the opponent. There is some risk involved in using Opponent-Model search. Both the prediction of the opponent's moves and the estimation of the profitability of future positions should be of good quality and as such they should obey certain conditions. To investigate the role of prediction and estimation in actual computer game-playing, experiments with Opponent-Model search were performed in the game of Bao. After five evaluation functions had been generated using machine-learning techniques, a series of tournaments between these evaluation functions was executed. They showed that Opponent-Model search can be applied successfully, provided that the conditions are met.

Keywords: opponent models, search, evaluation functions, Bao

1. Introduction

This contribution investigates under what conditions the usual form of Opponent-Model search (OM search) can be made successful. To understand the matter we provide a condensed introduction to OM search in Section 2. In Section 3 we give a brief overview of the family of mancala games to which Bao belongs and we describe the Bao rules. In Section 4 we explain how we obtained five evaluation functions for Bao. Section 5 gives the tournament setup and in Section 6 we present and discuss the results. The contribution ends with conclusions in Section 7. 1.2.

2. Opponent-Model Search

OM search (Carmel and Markovitch, 1993; Iida, Uiterwijk, and Van den Herik, 1993; Carmel and Markovitch, 1998; Donkers, Uiterwijk, and Van den Herik, 2003) is a game-tree search algorithm that uses a player's hypothesized model of the opponent in order to exploit weak points in the opponent's search

H. J. Van Den Herik et al. (eds.), *Advances in Computer Games*

strategy. The original formulation of the OM-search algorithm is based on three strong assumptions concerning the opponent and the player:

(1) the opponent (called MIN) uses minimax (or an equivalent algorithm) with an evaluation function (V_{op}), a search depth and a move ordering that are known to the first player (called MAX);
(2) MAX uses an evaluation function (V_0) that is *better* than MIN's evaluation function;
(3) MAX searches at least as deep as MIN.

This OM search procedure prescribes that MAX maximizes at max nodes, and selects at min nodes the moves that MAX believes MIN would select. Below we provide a short technical description of OM search, its notation, the relations between the nodes in the search tree, and some hints for an efficient implementation. For an extensive description of OM search we refer to Donkers et al. (2003).

OM search can be described by the following equations, in which $V_0(\cdot), V_{op}(\cdot)$ are the evaluation functions, and $v_0(\cdot), v_{op}(\cdot)$ are the node values. Subscript '0' is used for MAX values, subscript '*op*' is used for MIN values.

$$v_0(P) = \begin{cases} \max_j v_0(P_j) & \text{max nodes,} \\ v_0(P_j), \quad j = \min \arg_i v_{op}(P_i) & \text{min nodes,} \\ V_0(P) & \text{leaf nodes.} \end{cases} \tag{1}$$

$$v_{op}(P) = \begin{cases} \max_j v_{op}(P_j) & \text{max nodes,} \\ \min_j v_{op}(P_j) & \text{min nodes,} \\ V_{op}(P) & \text{leaf nodes.} \end{cases} \tag{2}$$

If P is a min node at a depth larger than the search-tree depth of the opponent, then $v_0(P) = \min_j v_0(P_j)$.

2.1 Implementation

For a search tree with branching factor w and even fixed depth d, OM search needs exactly $n = w^{d/2}$ evaluations of $V_0(\cdot)$ to determine the root value, since the search strategy is as follows: in each max node *all* w children are investigated and in each min node, *only one* child is investigated (see Donkers, Uiterwijk, and Van den Herik, 2001). Because the OM-search value is defined as the maximum over all these n values of $v_0(\cdot)$, none of these values can be missed. This means that the efficiency of OM search depends on how efficient the values for $v_{op}(\cdot)$ can be obtained.

A straightforward and efficient way to implement OM search is by applying α-β *probing*: at a min node it starts performing α-β search with the opponent's evaluation function (the *probe*), and thereafter it performs OM search with the

move that α-β search has returned; at a max node, it maximizes over all child nodes. The probes can be efficiently implemented by using an enhanced form of α-β search. Because for every min node, a separate probe is performed, many nodes are visited during multiple probes. (For example, every min node P_j on the principal variation of a node P will be probed at least twice.) Therefore, the use of transposition tables leads to a major reduction of the search tree.

The α-β probes at a min node P and at each grandchild (min nodes P_{j_k}) are not independent since the α-β value of P, $v_{op}(P)$, is necessarily equal to or larger than all α-β values of P_{j_k}. This means that $v_{op}(P)$ can be used to reduce the window of the probes at the grandchild nodes by setting the β parameter of the probe at P_{j_k} to $v_{op}(P) + 1$.

OM search assumes that MAX speculates on all min nodes about the move that MIN is going to choose. In deeper parts of the search tree, the prediction of MIN's move is based on shallower α-β probes than higher in the tree. It could therefore be justified to speculate only in the upper portion of the search tree.

2.2 Risk in Opponent-Model Search

Although using knowledge of the opponent during search seems obvious and OM search looks like a reasonable approach, there are three different types of risk involved. If these risks are not taken seriously, OM search is bound to fail.

First, OM search does not take into account any uncertainty about the opponent: the reasoning by the algorithm assumes perfect knowledge in the above sense. Since perfect knowledge of the opponent is hardly available in reality, this is a strong assumption. When the knowledge of the opponent is not perfect, the algorithm can still be used, but this will cause a certain amount of risk, depending on the quality of the knowledge. This first kind of risk has been described extensively in Iida, Handa, and Uiterwijk (1995). (In Donkers et al. (2001) an extension of OM search is described that does include uncertainty: *Probabilistic* Opponent-Model search.)

In Donkers et al. (2003), a second kind of risk in using OM search is investigated. It appears that even when MAX has perfect knowledge of MIN's evaluation function, using OM search may be unwise: when MAX makes a large overestimation of the profitability of a certain position while MIN is judging it correctly, then MAX is possibly attracted to that position. A condition that should prevent this from happening is called *admissibility* of the pair of evaluation functions: MAX should not overestimate a position that MIN not also overestimates.

A third kind of risk in using OM search (introduced in this contribution) is as follows. Perfect knowledge of the opponent's evaluation function is not equal to a perfect prediction of the opponent's moves. This is caused by the difference (normally one ply) in search depth between a player's prediction of

the opponent's move and the actual search that the opponent uses at the next move.

For OM search to be successful, the effects of these risks should be alleviated. In a series of experiments on the game of Bao, we investigate what the influence is of a good prediction of the opponent's moves and what the influence is of a better estimation of the own profitability of positions. Moreover, we study the effect of risk management. We assume perfect knowledge of the opponent's evaluation function but admissibility is not guaranteed.

3. The Mancala Game Bao

In large parts of the world, board games of the mancala group are being played in completely different versions (cf. Murray, 1952; Russ, 2000). Whatever the case, most mancala games share the following five properties:

(1) the board has of a number of *holes*, usually ordered in rows;
(2) the game is played with indistinguishable *counters* (also called pebbles, seeds, shells);
(3) players own a fixed set of holes on the board;
(4) a move consists of taking counters from one hole and putting them one-by-one in subsequent holes (*sowing*), possibly followed by some form of capture;
(5) the goal is to capture the most counters (for Bao it is to immobilize the opponent).

Mancala games differ in the number of players (1, 2 or more), the size and form of the board, the starting configuration of the counters, the rules for sowing and capturing, and in the way the game ends. The games of the mancala group are known by many names (for instance Wari, Awele, Bao, Dakon, and Pallankuli). For an overview of different versions and the rules of many mancala games, we refer to Russ (2000).

Among the mancala games, (Zanzibar) Bao is regarded as the most complex one (De Voogt, 1995). This is mainly due to the amount of rules and to the complexity of the rules. Bao is played in Tanzania and on Zanzibar in an organized way. There exist Bao clubs that own the expensive boards and that organize official tournaments.

The exact rules of the game are given in, for example, De Voogt (1995). Below, we summarize the properties that discriminate the game from the more widely known games Kalah and Awari.

Bao is played on a board with 4 rows of 8 holes by two players, called South and North, see Figure 1. Two square holes are called *houses* and play a special part in the game. There are 64 stones involved. At the start of the game each player has 10 stones on the board and 22 stones in store. Sowing only takes place on the own two rows of holes. The direction of sowing is not fixed. At

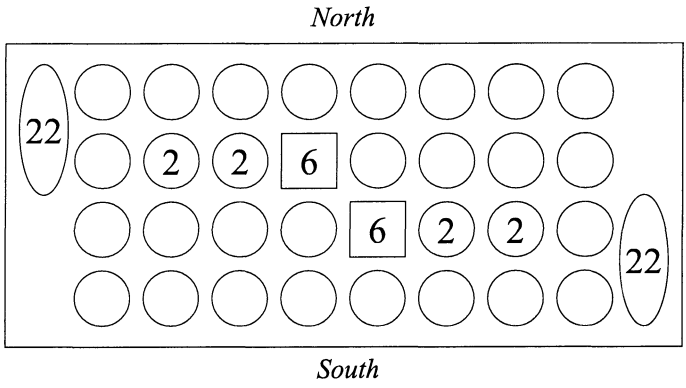

Figure 1. Opening position of Bao.

the start of a move, a player can select a direction for the sowing (clockwise or anti-clockwise). During sowing or at a capture, the direction can turn at some point. This is dictated by deterministic rules.

If a capture is possible then it is obliged in Bao. This means that a position is either a capture position or a non-capture position. Captured counters do not leave the board but re-enter the game. Counters are captured from the opponent's front row. These counters are immediately sown in the own front row. This implies that the game does not converge like Kalah and Awari.

Moves are composite. If at the end of a sowing, capture is possible, the captured counters are sowed immediately at the own side of the board. This second sowing can again lead to a new capture followed by a new sowing. If a capture is not possible, and the hole reached was non-empty, all counters are taken out of that hole and sowing continues. This procedure goes on until an empty hole is reached, which ends the move.

Moves can be endless because in a non-capture move, sowing can go on forever. The existence of endless moves can be proven theoretically (Donkers, Uiterwijk, and De Voogt, 2002). In real games, moves that take more than an hour of sowing also occasionally occur, but players usually make small mistakes during sowing or simply quit the game. So, real endless moves never lead to endless sowing.

Bao games consist of two stages: in the first stage, stones are entered one by one on the board at the start of every move. In the first stage, a game ends if the player to move has no counters left in the front row. As soon as all stones are entered, the second stage begins and a new set of rules applies. In the second stage, a game ends if the player has no more than one counter in any hole of both rows. A draw is not defined in Bao. Note that the goal of Bao is not to capture the most stones, but to immobilize the opponent.

In Donkers and Uiterwijk (2002), an analysis of the game properties of Bao is provided. The state-space complexity of Bao is approximated to be 1.0×10^{25}, which is much higher than those of Awari (2.8×10^{11}) and Kalah (1.3×10^{13}). The shortest game possible takes 5 ply, but most games take between 50 and 60 ply because they end (soon) after the start of the second stage. The maximum number of moves possible at any position is 32, but the average number of possible moves varies between 3 and 5, depending on the stage of the game. Forced moves occur quite often. The average game length (d) and branching factor (w) are normally used to estimate the size of a game tree that has to be traversed during search (w^d). For Bao the estimate is roughly 10^{34}. This number together with the game-tree complexity (10^{25}) places Bao in the overview of game complexities above checkers and in the neighbourhood of Qubic (Van den Herik, Uiterwijk, and Van Rijswijck, 2002).

4. Generating Evaluation Functions for Bao

In order to conduct the OM-search experiments, we created 5 different evaluation functions. We describe them below. (For operational reasons (see Section 5) we would like to have them ordered in increasing quality with respect to the strength of the resulting players.)

The first two evaluation functions were created by hand. The first one, called MATERIAL, simply takes the difference in the number of stones on both sides of the board as the evaluation score. The second hand-made evaluation function is called DEFAULT. This function incorporates some rudimental strategic knowledge of Bao. For instance, it is good to have more stones in your back row since this increases the mobility in the second stage of the game. The function awards 3 points to stones in the front row, 5 points to stones in the back row, and 5 additional points to opponent stones that can be captured. If the own house is still active, 200 extra points are given. The total score of the position is the score for MAX minus the score for MIN. There is a small asymmetry in this function: if MAX can capture MIN's house 100 points are rewarded, but if MIN can capture MAX's house, only 50 points are subtracted. This asymmetry is intended to produce a more offensive playing style.

The third evaluation function was created by using a genetic algorithm (Holland, 1975). The evaluation function was represented by an integer-valued chromosome of 27 genes: one gene for the material balance, one gene per hole for the material in the own back and front row, one gene per hole in the front row for capturing, one gene for an active house, and another gene for capturing the opponent's house. The total score of a position was the score for the player minus the score for the opponent. The fitness of a chromosome was measured by the number of games out of 100 that it won against a fixed opponent. In these matches, both players used α-β search with search depth 6. The genetic-

algorithm parameters were as follows: the population size was 100, only the 10 fittest chromosomes produced offspring (using a single-point crossover), the mutation rate is 5 per cent for large changes in a gene (i.e., generate a new random number for the gene) and 20 per cent for minor changes (i.e., altering the value of a gene slightly). The genetic algorithm was continued until no improvement occurs anymore. We conducted three runs: in the first run, the opponent was DEFAULT. In the second and third run, the opponent was the winner of the previous run. The name of the resulting evaluation function is GA3.

Thereafter we used another machine-learning technique to create the fourth evaluation function, namely TD-Leaf learning (Baxter, Trigdell, and Weaver, 1998). This is a temporal-difference method that is specialized for learning evaluation functions in games. The evaluation function trained was a linear real-valued function with the same parameters as the genes in the chromosomes above, except that there were separate parameters for the two sides of the board. Batch learning is applied with 25 games per batch. The reinforcement signal used to update the parameters was the number of games in the batch that the player wins against a fixed opponent, in this case GA3. The search depth used in the games was 10. The λ-factor and the annealing factor both were set to 0.99. This produced our fourth evaluation function, called TDL2B.

The last evaluation function was also produced by TD-Leaf learning, but this time we used a *normalized Gaussian network* (NGN) as evaluation function, similar to way in which Yoshioka, Ishii, and Ito (1999) trained an evaluation function for the Othello game. The NGN had 54 nuclei in a 54-dimensional space. Every dimension correlated with a parameter in the previous evaluation function. The reinforcement signal was the number of games out of 25 won against a fixed opponent, being TDL2B. The search depth used in the games was 6, because the computation of the output for an NGN is relatively slow. No batch learning was applied here. The λ-factor was set to 0.8 and the annealing factor was set to 0.993. This evaluation function is called NGND6.

5. Experimental Set-up

We conducted seven different tournaments between five players that each used one of the five evaluation functions. We denote the players by the name of their evaluation function. All tournaments followed a *double* round-robin system: every player was matched against every other player, one time playing South and one time playing North. Each match between two players consisted of 100 games; hence each tournament counted 2000 games. In the tables below the results are reported in a special way (see the caption of Table 1). One reason is that we can easy read off from the table any improvement by an evaluation function. The games began at the start positions given in the Appendix and

were played to the end. To prevent problems with infinite moves, any move that involves the sowing of more than 100 stones was considered infinite and illegal. A position at which a player could only perform one of these long moves was a loss for that player.

In the first two tournaments, both players used α-β search. In the other five tournaments, South used OM search with perfect knowledge of the opponent's evaluation function. North always used α-β search with search depth 6. The search depth of South differed per tournament. No time restrictions were given. We used an implementation of OM search with α-β probes and allowed only one ply of speculation. Since in Bao draws are not possible, and since we aimed to compare the performance of the different search algorithms used by South, the score of a match was just the number of games out of 100 that was won by South.

At every position at which South was to move, we also detected the move(s) that α-β search would select for South. In this way we were able to count the number of times that OM search differed from α-β.

In the implementation of the α-β probes for OM search we took care of the fact that (some of) the evaluation functions are asymmetric. The asymmetry implies that evaluating a position when South is MAX, is not the same as evaluating the same position when North is MAX and taking the negative of the value. Furthermore, we dealt with multiple equipotent moves for MIN: if MIN has multiple equal choices, MAX will select the move with the lowest value for v_0.

The exact set-up of each of the seven tournaments will be explained along with the results in the next section.

6. Results and Discussion

First tournament: α-β plain — Table 1 gives the outcome of the first tournament. Both South and North used α-β search with search depth 6. The table clearly shows that the evaluation functions differ in quality and that every following evaluation function is operationally better than any of the previous ones. (Since the size of each match is 100, the 95% confidence intervals are approximately plus/minus 10 per match and plus/minus 20 for the total scores.)

Second tournament: α-β extended — The second tournament was a checking tournament. South was allowed to search two extra plies (8 instead of 6). The results are presented in Table 2. The table shows that all players profited from the increased search depth. Only the match of DEFAULT against GA3 was less fortunate for DEFAULT. This illustrates the poor quality of this evaluator.

Third tournament: OM plain — In the third tournament, South used OM search with one ply of speculation and with search depth 6. The results in Table 3 show that three players, MATERIAL, GA3, and TDL2B, profited from using

OM search, but that the two other players, DEFAULT and NGND6, did not profit and played worse than in the first tournament.

Fourth tournament: OM extended — South was using OM search with one ply of speculation as in the third tournament, but in this tournament it was allowed to search two ply deeper. The α-β probes were still restricted to depth 6. This means that South had better knowledge over the game than North, a situation that is comparable to the second tournament. Table 4 shows that South was not able to profit fully from the extra search depth. Although all players performed better than in the third tournament where they were given a depth of 6 ply, only MATERIAL played better than in the second tournament. This indicates that searching deeper for yourself in OM search is not sufficient for success.

Fifth tournament: OM with perfect opponent prediction — The fifth tournament gave South a different advantage: it was allowed to extend the α-β probes to depth 7. The search depth (for the own evaluation) was 6. In this way, South not only had perfect knowledge of the opponent's evaluation function, but South could also predict almost perfectly what North would be doing in the next move. The search depth of the α-β probes (which was 6, because the probes started at depth 1) was namely exactly the same as the search depth of North. In the case of equal evaluated moves, South selected the move with the lowest own evaluation. This was not necessarily the move that North would play. Table 5 gives the results of this tournament. All players, except DEFAULT profited from this advantage, and played better than in tournament 1, albeit less good than in the second tournament in which they just searched deeper. The advantage also gave less good results than the advantage in tournament 4, except for player NGND6. From these results we can infer that knowing exactly the moves of the opponent does not help if the own judgement is too weak.

Sixth tournament: OM perfect — The sixth tournament combined the advantages of the fourth and fifth tournament for South. The search depth for the own

S \ N	MATERIAL	DEFAULT	GA3	TDL2B	NGND6	Score
MATERIAL	-	55	35	19	18	127
DEFAULT	48	-	54	30	28	160
GA3	55	61	-	36	30	182
TDL2B	69	65	57	-	39	230
NGND6	79	73	75	60	-	287

Table 1. Results of the first tournament between 5 evaluation functions for Bao. Each cell shows the number of games won (out of 100) by South (the row) against North (the column). The column on the right shows the number of games won (out of 400) by each evaluation function when playing South.

S \ N	MATERIAL	DEFAULT	GA3	TDL2B	NGND6	Score
MATERIAL	-	57	62	40	32	191
DEFAULT	71	-	52	49	34	206
GA3	80	75	-	62	49	266
TDL2B	86	76	69	-	57	288
NGND6	88	76	80	70	-	314

Table 2. Results of the second tournament between 5 evaluation functions for Bao. Both sides use α-β, but South searches 2 ply deeper (8) than North (6).

S \ N	MATERIAL	DEFAULT	GA3	TDL2B	NGND6	Score
MATERIAL	-	57	50	30	24	161
DEFAULT	46	-	46	26	25	143
GA3	59	57	-	40	35	191
TDL2B	78	64	60	-	46	248
NGND6	71	58	66	61	-	256

Table 3. Results of the third tournament between 5 evaluation functions for Bao. South uses OM search with perfect knowledge of the opponent's evaluation function. The search depth is 6 for both sides.

S \ N	MATERIAL	DEFAULT	GA3	TDL2B	NGND6	Score
MATERIAL	-	60	64	49	39	212
DEFAULT	63	-	47	44	41	195
GA3	70	66	-	57	40	233
TDL2B	80	69	70	-	56	275
NGND6	84	68	71	59	-	282

Table 4. Results of the fourth tournament between 5 evaluation functions for Bao. South uses OM search with perfect knowledge of the opponent's evaluation function. The search depth is 8 for South, with α-β-probes to depth 6, and the search depth is 6 for North.

evaluation was 8 for South and the α-β probes for the opponent extended to depth 7. The results in Table 6 show that the power of South was significantly increased. All players performed better than in tournament 1, and all players, except GA3 also played better than in tournament 2. The results of GA3 were only slightly less than in tournament 2.

Seventh tournament: OM with strict risk management — In the seventh and last tournament, South applied OM search with strict risk management. South only

S \ N	MATERIAL	DEFAULT	GA3	TDL2B	NGND6	Score
MATERIAL	-	59	57	31	27	174
DEFAULT	50	-	48	32	23	153
GA3	53	64	-	36	30	183
TDL2B	69	73	66	-	40	248
NGND6	77	82	77	63	-	299

Table 5. Results of the fifth tournament between 5 evaluation functions for Bao. South uses OM search with perfect knowledge of the opponent's evaluation function. The search depth is 6 for both sides, but South uses α-β probes to depth 7.

S \ N	MATERIAL	DEFAULT	GA3	TDL2B	NGND6	Score
MATERIAL	-	76	69	54	58	257
DEFAULT	59	-	66	48	46	219
GA3	75	77	-	56	55	263
TDL2B	79	88	83	-	57	307
NGND6	80	88	85	68	-	321

Table 6. Results of the sixth tournament between 5 evaluation functions for Bao. South uses OM search with perfect knowledge of the opponent's evaluation function. The search depth is 8 for South, with α-β probes to depth 7, and the search depth is 6 for North.

S \ N	MATERIAL	DEFAULT	GA3	TDL2B	NGND6	Score
MATERIAL	-	66	49	32	33	180
DEFAULT	49	-	45	37	31	162
GA3	63	62	-	35	33	193
TDL2B	72	66	64	-	46	248
NGND6	75	71	76	69	-	291

Table 7. Results of the seventh tournament between 5 evaluation functions for Bao. South uses OM search with perfect knowledge of the opponent's evaluation function and strict risk management. The search depth is 6 for both sides.

deviated from the strategy that α-β search imposed if the move that OM search advised had the same Minimax value. The search depth was equal to the third tournament. Since it occurred relatively often in Bao that multiple moves at the same position had the same Minimax value, South did have some room to speculate. The results in Table 7 show that this approach was successful too.

Tournament	MATERIAL	DEFAULT	GA3	TDL2B	NGND6
1: α-β Plain	127	160	182	230	287
2: α-β Extended	191	206	266	288	314
3: OM plain	161	143	191	248	256
4: OM extended	212	195	233	275	282
5: OM perf. opp.	174	153	183	248	299
6: OM perfect	257	219	263	307	321
7: OM no risk	180	162	193	248	291

Table 8. Overview of the seven Bao tournaments.

All players performed better than in the first tournament and also better (or equally poor in case of TDL2B) than in the third tournament.

A summary — Table 8 summarizes the results of the seven tournaments. Each cell contains the final score of a tournament (400 games), playing South. On all rows the scores are increasing from left to right (except for the first two columns). This means that the order of quality for the evaluation functions indeed is as follows: (MATERIAL, DEFAULT) < GA3 < TDL2B < NGND5. The ordering of MATERIAL and DEFAULT is unclear, but both evaluation functions are poor. The table shows that if only the search depth is increased (4: OM extended) or only the prediction of the opponent is improved (5: OM perf. opp.), the results are not as good as just using α-β with two additional ply of search. When both methods were combined, (6: OM perfect), the results were better. Furthermore, the table shows that using OM search with strict risk management (7: OM no risk) led to better results than using plain OM search and plain α-β.

Deviations — The last overview, in Table 9, provides insight into the number of times that OM search deviated from the α-β-search strategy. The table shows that searching more deeply for the own evaluation had a larger effect than searching more deeply for the prediction of the opponent. The table also shows that the number of deviations was larger in tournament 4 than in tournament 6. Since the results of tournament 4 were less good than the results of tournament 6, it seems that an incorrect prediction of the opponent leads to extra deviations that did not contribute to a positive outcome.

7. Conclusion

The experiments described in this paper are a follow-up to earlier experiments with OM search in other game domains. In Donkers et al. (2003) we described experiments in Lines of Action and in the chess endgame KQKR. The experiments in Lines of Action showed that OM search with evaluation functions of poor quality led to bad results. The experiments in the chess endgame KQKR

Tournament	MATERIAL	DEFAULT	GA3	TDL2B	NGND6
3: OM plain	3.27±2.1	3.36±1.8	2.38±1.8	2.47±1.7	2.21±1.6
4: OM extended	4.85±2.5	5.43±2.3	4.90±2.4	4.46±2.3	4.55±2.3
5: OM perf. opp.	3.30±1.9	3.12±1.8	2.52±1.6	2.43±1.6	2.26±1.6
6: OM perfect	4.35±2.1	5.14±2.5	4.47±2.2	3.93±1.9	3.98±1.9
7: OM no risk	1.83±1.5	0.82±1.0	0.29±0.5	0.73±0.9	0.50±0.7

Table 9. Overview of the average number of moves per game in which the move that OM search selected differed from the move that α-β search (with search depth 6) suggested. The standard deviation ranges between 0.5 and 2.5. The average number of moves per game for South is 19.8 over all games in the tournaments.

showed that OM search with a perfect evaluation function (i.e., an endgame database) for MAX can be useful, but the results are not conclusive.

The Bao experiments in this contribution were designed to identify those factors that influence the success or failure of OM search. Although the experiments were not encyclopedic and therefore did not produce firm qualifications of these factors, many effects are statistically significant. In all, the Bao experiments provide a good insight into the working of OM search. For instance, it appears that a combination of adequate opponent prediction and extended search depth is needed for good results. Of these two factors, the extended search depth seems to be more important than the good prediction. Moreover, the quality of the evaluation functions appears to be important for the effect of OM search. For plain OM search the results were not good for most of the players because the evaluation functions do not obey the admissibility requirement.

A generalisation of the results to other games leads to the statement that the search method can only be applied successfully when additional resources (e.g., search time) are available. The additional search time (in comparison with the opponent) must either be spent for the prediction of the opponent's move, or for the risk management. If these additional resources are not available, OM search cannot with certainty be applied successfully.

In order to measure the effects of opponent prediction and extended search more precisely, the sample size should be increased and more game details should be analysed, such as the number of times that the predicted move differs from the move played by the opponent. Furthermore, a deeper study of the properties of the trained evaluation functions and the matches between players themselves might provide more background information. A final suggestion for future research is to investigate the possibilities for risk management more deeply since this seems a promising approach.

References

Baxter, J., Trigdell, A., and Weaver, L. (1998). KnightCap: a Chess Program that Learns by Combining TD(λ) with Game-Tree Search. *Proc. 15th International Conf. on Machine Learning*, pp. 28–36, Morgan Kaufmann, San Francisco, CA.

Carmel, D. and Markovitch, S. (1993). Learning Models of Opponent's Strategies in Game Playing. *Proceedings AAAI Fall Symposion on Games: Planning and Learning*, pp. 140–147, Raleigh, NC.

Carmel, D. and Markovitch, S. (1998). Pruning Algorithms for Multi-Model Adversary Search. *Artificial Intelligence*, Vol. 99, No. 2, pp. 325–355.

Donkers, H. H. L. M. and Uiterwijk, J. W. H. M. (2002). Programmming Bao. *Seventh Computer Olympiad: Computer-Games Workshop Proceedings* (ed. J. W. H. M. Uiterwijk), Technical Report CS 02-03, Universiteit Maastricht, Maastricht, The Netherlands.

Donkers, H. H. L. M., Uiterwijk, J. W. H. M., and Herik, H. J. van den (2001). Probabilistic Opponent-Model Search. *Information Sciences*, Vol. 135, pp. 123–149.

Donkers, H. H. L. M., Uiterwijk, J. W. H. M., and Voogt, A. J. de (2002). Mancala Games – topics in Artificial Intelligence and Mathematics. *Step by Step. Proceedings of the 4th Colloquium 'Board Games in Academia'* (eds. J. Retschitzki and R. Haddad-Zubel), Editions Universitaires, Fribourg, Switserland.

Donkers, H. H. L. M., Uiterwijk, J. W. H. M., and Herik, H. J. van den (2003). Admissibility in Opponent-Model Search. *Information Sciences*, Vol. 154, Nos. 3–4, pp. 195–202.

Herik, H. J. van den, Uiterwijk, J. W. H. M., and Rijswijck, J. van (2002). Games Solved: Now and in the Future. *Artificial Intelligence*, Vol. 134, Nos. 1–2, pp. 277–311.

Holland, J. H. (1975). *Adaption in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

Iida, H., Uiterwijk, J. W. H. M., and Herik, H. J. van den (1993). Opponent-Model Search. Technical Report CS 93-03, Universiteit Maastricht, Maastricht, The Netherlands.

Iida, H., Handa, K.-i., and Uiterwijk, J. W. H. M. (1995). Tutoring Strategies in Game-Tree Search. *ICCA Journal*, Vol. 18, No. 4, pp. 191–204.

Murray, H. J. R. (1952). *A History of Board Games other than Chess*. Oxford University Press, Oxford, UK.

Russ, L. (2000). *The Complete Mancala Games Book*. Marlow & Company, New York, NY.

Voogt, A. J. de (1995). *Limits of the Mind. Towards a Characterisation of Bao Mastership*. Ph.D. thesis, University of Leiden, The Netherlands.

Yoshioka, T., Ishii, S., and Ito, M. (1999). Strategy Acquisition for the Game Othello Based on Reinforcement Learning. *IEICE Transactions on Information and Systems*, Vol. E82-D, No. 12, pp. 1618–1626.

Appendix

The following table gives the 100 start positions used in the Bao experiments. The positions are generated by playing 10 random legal moves for every player from the official Bao opening position. Each row gives the contents of the holes of one position. The numbering of the holes is according to De Voogt (1995). The last two columns indicate whether South and North have an active house.

Row b								Row a								Row A								Row B								House	
1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	S	N
1	1	0	0	1	1	1	1	0	0	0	0	1	2	10	0	0	3	9	1	2	4	0	0	0	0	0	0	0	0	1	1	F	F
1	1	1	1	1	1	1	1	1	3	7	0	4	0	6	2	0	1	0	1	0	5	0	0	0	0	0	0	0	0	1	1	F	F
3	2	1	0	0	0	0	0	0	1	1	11	1	0	1	0	0	2	2	0	0	2	4	0	1	1	1	1	1	1	2	1	F	T
0	0	0	0	0	0	0	0	1	5	6	0	1	3	0	1	0	0	1	7	1	1	5	0	1	1	1	1	1	1	0	2	F	F
1	1	0	0	0	1	1	1	0	0	0	1	1	0	0	0	1	0	0	3	15	1	1	5	1	1	0	0	0	1	2	2	T	F
1	0	3	3	1	0	4	1	6	0	1	1	10	5	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	F	F
0	1	1	0	2	1	3	3	0	3	6	6	0	0	0	1	1	1	0	1	0	4	3	0	1	0	0	0	0	0	1	1	F	F
2	2	0	1	1	0	1	1	0	1	0	1	1	1	1	0	1	0	0	8	12	1	0	0	2	1	1	1	0	0	0	0	T	F
1	1	1	1	1	1	1	3	3	1	1	1	0	4	3	1	0	4	1	2	0	6	2	1	0	0	0	0	0	0	0	0	F	F
2	1	0	0	0	1	2	0	0	0	1	3	4	1	0	4	1	1	5	0	2	5	0	0	1	0	0	0	0	1	2	3	F	F
3	0	1	1	0	0	0	0	0	2	0	9	0	0	3	1	1	0	2	0	8	0	0	0	1	1	1	1	0	2	0	3	T	T
1	1	0	0	0	0	1	1	0	1	1	1	6	1	3	2	1	1	5	3	1	2	0	1	2	2	1	0	0	0	1	1	F	F
2	1	1	1	1	1	1	1	1	1	0	1	2	0	5	1	2	1	1	0	0	4	0	3	0	2	0	2	0	2	2	1	F	F
2	2	1	0	0	0	0	0	0	3	2	11	1	0	1	0	0	0	0	1	13	0	0	0	0	0	0	0	0	1	2	0	T	T
0	0	0	1	1	1	1	1	0	0	7	0	1	1	1	0	2	6	0	0	1	0	6	0	0	2	0	2	0	2	1	3	F	F
1	1	0	0	0	0	0	0	1	2	4	14	4	4	1	1	0	0	0	0	1	0	1	0	0	2	1	0	0	0	1	1	F	T
0	2	2	0	1	1	0	0	1	4	2	11	0	0	0	0	0	0	0	0	9	1	1	0	0	0	0	0	1	1	1	2	T	T
0	0	0	0	0	0	1	1	2	2	1	1	2	2	1	1	1	7	0	0	13	0	0	1	1	1	1	0	0	0	0	1	T	F
1	1	0	1	1	1	1	1	1	4	1	1	1	5	1	0	0	0	0	0	13	0	5	0	0	0	0	0	0	0	0	1	T	F
3	1	1	1	0	0	1	1	0	1	0	11	0	0	0	0	0	0	0	0	10	1	1	0	2	0	1	1	1	1	0	2	T	T
1	1	0	0	1	1	1	1	2	4	1	6	0	4	3	3	1	3	6	1	0	0	0	0	0	0	0	0	0	0	0	0	F	F
1	0	2	1	0	2	0	2	0	4	0	0	7	5	0	1	0	0	6	2	0	1	1	1	0	0	0	0	1	1	0	2	F	F
2	2	1	1	1	1	1	1	1	2	2	0	1	2	0	2	0	0	0	0	12	1	0	1	1	1	0	0	0	1	1	2	T	F
0	1	3	0	1	4	4	0	1	5	1	4	2	4	5	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	F	F
1	1	0	0	0	1	1	2	0	3	0	9	0	0	2	1	1	0	0	0	11	0	0	0	0	0	1	1	0	2	0	3	T	T
0	2	2	1	3	0	1	1	6	1	0	2	1	0	1	2	2	1	6	0	1	0	3	3	0	0	0	0	0	0	0	1	F	F
0	0	0	0	0	0	0	0	1	0	3	10	0	5	0	1	1	1	0	0	15	0	2	1	0	0	0	0	0	0	0	0	T	T
1	1	0	2	0	2	2	2	0	1	4	5	7	4	0	1	0	0	0	0	0	1	1	3	0	0	0	0	0	1	2	0	F	F
2	2	2	1	1	1	1	1	2	3	1	3	6	2	1	0	0	0	0	0	0	1	3	0	1	1	1	1	1	0	1	1	F	F
0	0	0	0	0	0	0	1	1	0	8	0	0	0	3	0	0	8	0	4	4	0	2	2	1	1	1	1	0	2	1	0	F	F
2	1	2	2	1	1	1	1	0	1	4	0	1	0	6	1	1	0	0	0	2	9	0	0	0	0	0	0	1	1	0	2	F	F
0	0	0	1	1	1	1	1	0	0	0	6	0	1	1	0	0	9	1	0	2	1	5	0	2	2	0	2	0	2	1	0	F	F
0	0	0	0	0	0	0	0	1	2	1	3	1	3	1	1	1	2	0	0	6	1	0	2	2	0	2	2	2	1	4	2	F	F
1	1	0	1	1	1	1	1	0	5	4	0	5	1	7	0	0	1	0	1	0	6	0	0	0	0	0	0	0	1	2	0	F	F
0	1	1	0	1	1	1	1	3	1	0	0	0	0	0	0	0	4	3	4	5	5	3	2	1	0	0	0	0	1	1	1	F	F
0	2	1	2	0	2	0	2	0	5	1	1	5	0	6	0	0	0	0	3	0	1	1	3	0	2	1	0	0	0	1	1	F	F
0	0	0	0	0	0	0	0	1	3	1	3	1	1	1	1	2	1	4	0	1	3	2	5	2	1	2	0	4	0	1	0	F	F
1	1	1	1	1	0	1	1	0	1	4	0	0	5	0	1	4	2	0	3	1	1	0	6	2	1	0	0	0	0	1	1	F	F
3	0	2	0	2	0	2	2	1	3	1	0	0	0	0	2	1	0	0	0	11	2	0	0	0	0	0	1	1	2	0	4	T	F
1	1	2	0	4	2	4	0	0	6	0	1	9	4	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	1	1	F	F
0	0	0	0	0	0	0	0	4	1	1	3	4	0	0	5	1	2	0	0	0	3	3	3	1	1	1	1	1	2	2	1	F	F
1	1	1	0	0	0	0	0	1	0	0	14	0	0	1	0	1	6	0	0	0	1	3	0	2	2	2	2	0	1	1	0	F	T
1	1	1	0	0	0	0	1	0	0	0	1	1	3	0	5	2	1	2	3	2	0	0	0	0	2	1	3	1	3	3	3	F	F
1	1	0	0	0	0	0	0	1	0	2	12	5	4	0	1	0	0	0	0	0	0	3	1	0	1	2	0	2	0	2	2	F	T
0	1	2	0	1	1	0	1	0	1	2	11	0	4	2	0	0	1	1	2	0	0	1	0	2	0	2	0	2	0	1	2	F	T
2	2	2	2	1	0	0	0	0	2	1	3	1	1	3	1	0	1	3	0	10	0	0	1	1	1	0	0	0	0	1	1	T	F
1	1	0	0	0	0	0	0	0	0	0	0	2	6	1	1	1	6	9	4	2	0	1	0	1	1	1	1	1	0	0	0	F	F
0	0	0	0	0	0	0	1	0	0	6	1	2	2	1	1	4	1	3	4	0	0	0	4	3	1	0	1	3	0	1	1	F	F
0	0	0	0	0	0	0	0	0	6	1	1	0	4	3	1	1	0	1	3	15	0	1	0	0	0	0	0	0	1	1	1	T	F
1	0	0	0	1	1	1	1	0	4	2	2	2	0	1	1	0	0	0	3	3	0	8	0	0	2	0	2	0	4	0	1	F	F
1	1	0	0	0	0	1	1	2	1	0	1	2	5	2	0	2	3	3	6	2	0	1	0	1	1	0	0	1	1	0	2	F	F

Row b								Row a								Row A								Row B								House	
1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	S	N
0	0	0	0	0	0	0	0	0	0	1	1	1	3	0	1	1	5	1	9	4	2	0	0	3	1	2	0	2	2	0	1	F	F
0	0	0	0	0	0	0	0	5	0	3	5	4	0	0	1	0	0	0	0	0	9	3	0	1	3	1	0	1	3	1	0	F	F
1	0	2	0	2	0	2	0	0	3	1	1	5	0	1	1	0	0	3	1	2	1	12	0	0	0	0	0	0	1	1	0	F	F
2	1	1	2	2	0	2	0	0	3	1	1	0	7	0	0	0	0	0	0	15	0	1	0	0	0	0	0	0	0	1	1	T	F
2	1	1	1	0	0	0	0	0	5	0	13	0	0	0	1	0	1	4	0	0	2	2	0	1	1	1	1	1	0	1	1	F	T
2	0	1	1	0	0	0	0	0	4	1	8	0	5	1	0	0	0	0	0	9	0	0	0	0	0	1	1	1	3	0	2	T	T
2	1	2	1	0	1	3	0	0	6	1	5	5	1	3	1	0	0	0	0	0	0	0	3	0	0	1	1	0	2	1	0	F	F
0	1	2	0	1	1	0	0	0	3	1	13	1	0	1	0	0	0	1	0	9	0	0	0	0	0	0	0	1	1	1	3	T	T
2	0	1	0	2	0	2	2	0	3	5	1	3	2	1	1	0	3	0	0	0	0	5	1	0	0	0	1	1	1	1	2	F	F
1	0	3	2	1	0	0	0	0	1	1	10	0	0	0	0	0	0	0	0	12	1	2	1	0	0	0	0	0	1	2	2	T	T
0	2	1	0	0	0	0	0	0	1	1	1	1	3	4	0	0	5	8	2	4	0	0	0	1	0	0	0	1	1	2	2	F	F
1	1	1	3	1	0	3	3	1	4	2	0	0	4	0	0	0	0	0	1	11	0	0	0	1	1	0	0	0	0	1	1	T	F
1	0	0	0	0	1	1	1	2	0	3	1	1	6	1	1	1	0	1	3	1	1	2	2	0	1	1	0	2	0	3	3	F	F
2	2	3	1	3	1	2	0	1	2	1	0	4	0	1	5	3	1	0	0	0	0	0	0	0	0	1	1	1	1	2	2	F	F
2	2	0	2	1	0	0	0	0	0	3	1	0	1	3	1	5	0	2	4	1	0	0	1	1	1	1	0	2	0	3	3	F	F
0	2	0	1	1	0	0	0	3	1	1	0	0	0	0	0	0	6	3	5	0	4	8	2	1	1	0	0	0	0	0	1	F	F
1	0	0	0	0	0	1	1	1	2	3	3	3	0	2	0	0	7	1	0	0	1	5	0	1	0	3	1	2	0	2	0	F	F
1	1	1	0	2	1	3	3	0	2	0	0	0	1	4	1	2	1	1	2	1	1	0	0	1	1	1	1	1	2	2	3	F	F
0	3	0	3	1	0	1	3	1	3	1	3	1	9	3	1	0	4	0	0	1	0	0	0	0	0	0	0	0	0	1	1	F	F
0	2	3	1	0	1	3	1	1	1	0	4	6	1	3	0	0	5	0	2	2	1	0	0	0	0	0	0	0	0	1	2	F	F
2	1	1	1	0	0	0	0	0	0	0	8	1	1	0	0	0	2	1	0	10	1	6	0	1	1	1	0	0	0	1	1	T	T
0	0	1	1	1	1	1	1	0	10	0	1	1	3	1	1	1	4	3	4	3	0	1	1	0	0	0	0	0	0	0	0	F	F
1	0	4	1	2	1	0	2	0	0	0	6	1	1	0	0	1	1	0	0	10	2	0	0	1	1	0	1	1	2	0	1	T	T
1	0	1	3	2	0	2	1	1	2	3	1	2	4	1	0	0	1	0	0	2	0	5	0	1	1	1	1	1	1	1	1	F	F
0	1	2	0	2	0	2	0	0	5	7	3	1	2	0	1	0	0	0	0	0	0	4	1	2	2	1	1	1	1	1	0	F	F
1	1	2	0	2	0	2	0	3	2	1	0	0	4	0	1	0	4	0	4	5	1	0	2	0	0	0	0	0	1	2	2	F	F
0	0	0	0	0	0	0	0	0	0	0	13	2	1	1	1	1	3	0	1	0	4	4	1	2	0	2	0	2	0	1	1	F	T
1	0	0	0	0	0	1	1	1	2	3	1	6	0	1	1	0	4	1	3	0	4	0	1	1	1	1	0	1	1	2	2	F	F
2	2	1	0	0	0	0	0	0	1	7	13	0	0	0	1	0	5	1	0	0	4	1	0	0	0	0	0	0	0	1	1	F	T
1	1	0	0	0	0	0	0	1	2	1	3	1	2	1	0	1	8	2	0	11	0	0	0	0	0	0	1	1	2	0	1	T	F
2	1	0	0	0	0	0	0	2	1	2	7	1	0	0	0	0	1	8	0	5	2	3	2	0	0	0	0	0	1	2	0	F	F
0	2	0	2	1	0	0	0	2	0	0	11	0	2	1	1	0	0	0	1	12	0	1	1	0	0	0	0	0	1	1	1	T	T
2	0	1	1	0	0	0	0	1	0	0	10	3	0	0	1	1	3	4	1	0	0	3	1	1	1	2	0	1	0	2	1	F	T
1	1	0	0	0	0	0	1	7	1	8	0	3	5	3	0	0	0	0	0	2	0	5	0	0	0	0	0	0	0	1	2	F	F
0	2	1	1	0	0	0	0	1	0	0	9	0	2	0	0	0	4	1	0	11	0	2	1	0	0	1	1	0	2	1	0	T	T
1	0	0	0	0	0	0	0	2	4	1	0	1	1	2	1	1	0	1	3	13	1	1	1	1	1	1	1	1	0	0	1	T	F
0	0	0	0	0	0	0	0	1	1	4	0	0	5	0	1	1	1	0	4	13	0	3	0	1	1	0	0	1	1	0	2	T	F
0	2	1	0	0	0	1	1	0	0	3	12	4	1	1	0	2	3	0	1	0	2	4	0	0	0	0	0	0	1	1	0	F	T
1	1	0	0	0	0	0	1	0	6	1	1	1	1	2	0	1	0	2	3	0	5	6	0	1	1	1	1	0	1	1	2	F	F
2	2	1	1	1	1	0	0	0	7	1	1	0	0	0	0	0	0	1	0	11	2	1	1	1	1	1	1	0	1	2	0	T	F
1	1	0	0	0	0	0	0	0	1	1	14	0	0	0	0	0	8	1	0	8	1	0	2	1	0	0	0	0	0	0	1	T	T
2	0	1	1	0	0	0	0	0	0	0	0	1	1	3	0	1	4	0	0	12	3	1	0	1	2	0	3	1	2	1	0	T	F
0	0	0	0	0	1	1	1	0	4	4	10	1	7	3	1	1	1	1	0	2	0	0	0	0	0	0	0	0	0	1	1	F	T
2	1	2	1	1	0	0	1	0	3	4	1	7	1	0	0	0	1	1	1	1	2	0	3	2	1	1	0	0	1	2	0	F	F
1	1	0	0	0	0	1	1	1	5	3	1	2	0	1	0	0	0	0	0	2	2	7	1	2	0	2	0	2	1	4	0	F	F
0	0	0	0	1	1	1	1	0	0	0	1	2	1	5	0	0	10	6	2	1	0	0	0	1	1	1	1	1	1	1	1	F	F
4	3	1	1	1	0	1	1	2	0	0	2	5	5	0	0	0	0	0	0	0	0	0	1	0	1	1	0	2	2	3	4	F	F
0	0	0	0	1	1	1	1	2	3	1	1	3	4	2	1	1	1	4	3	0	1	1	1	1	0	0	0	1	2	3	0	F	F
1	3	2	2	1	1	1	1	2	3	1	0	1	4	1	2	0	2	4	0	0	0	0	1	0	0	0	1	1	0	2	3	F	F