# Investigating Probabilistic Opponent-Model Search*

H.H.L.M. Donkers J.W.H.M. Uiterwijk H.J. van den Herik

Institute for Knowledge and Agent Technology (IKAT)
Department of Computer Science, Universiteit Maastricht
email: {donkers,uiterwijk,herik}@cs.unimaas.nl

An often-made assumption in computer game playing is that the opponent plays as well as we play. By this is meant that the opponent uses all information that we use, plays as rational as we do, and uses heuristics that are as effective as ours. In a game-theoretic setting this might be a correct assumption, but playing games like chess does not happen in a game-theoretic setting. Although chess is basically a zero-sum game with perfect information, in practice the sheer size of the game tree makes it impossible to compute exact game values. Therefore, in computer game playing an evaluation function is used that computes an approximation of the true game value at a given node in the game tree. These approximations are then treated as real game values and the game tree is solved using some variant of the minimax algorithm, a procedure that does not guarantee the highest probability of winning the game.

One might do better if some knowledge about the opponent is used during the game-tree search. In incomplete-information games like poker, opponent modelling is an important part of the game, but in perfect-information games like chess, opponent modelling is not used often. One approach of using an opponent model in game-tree search is opponent-model search [1, 2]. In this approach, it is assumed that the opponent uses a given evaluation function. The OM-search algorithm then exploits weak spots in the evaluation function.

In this contribution we propose an extension of opponent-model search, called *probabilistic opponent-model search* (PrOM search) that incorporates uncertainty in the model of the opponent's evaluation function. This approach is not only closer to the human approach, it also creates opportunities for learning an opponent model during an actual game.

PrOM search uses multiple evaluation functions together with a probability distribution to model the player's uncertainty on the opponent. To enable this, three assumptions are made. The first assumption is knowledge of $n$ well-defined opponent types $\omega_0 \ldots \omega_{n-1}$, each opponent type $\omega_i$ having a separate fixed and well-known evaluation function $V_{\omega_i}$. One opponent type ($\omega_0$) is using the same evaluation function as MAX uses ($V_{\omega_0} \equiv V_0$). Second, it is assumed that MAX has subjective probabilities $\Pr(\omega_i)$ on the opponent being of one of these types, such

*Appeared in: Proceedings JCIS 2000 (ed. P.P. Wang), pp. 982-985. Association for Intelligent Machinery, Inc. ISBN 0-9643456-9-2.

that $\sum_i \Pr(\omega_i) = 1$. The third assumption is that MIN is using a *mixed strategy* consisting of $n$ minimax strategies, one for every opponent type. From these strategies MIN picks one randomly at every move, according to the probabilities $\Pr(\omega_i)$.

The probabilities on opponent types lead to probabilities on moves at the min nodes in the game tree. Assume that at a min node $P$ there is a set of $m$ available moves $M(P) = \{P_1, \ldots, P_m\}$. The opponent plays minimax with one of the opponent types and hence will select a move with a minimal value for that type. When more than one move have the same minimal value, one of the moves is selected at random. The probability that the opponent selects a certain move is thus given by:

$$\Pr(P_j) = \textstyle\sum_i \Pr(\omega_i) \cdot W_i(P_j), \text{ where}$$

$$W_i(P_j) = \begin{cases} 0, & \text{if} \quad v_{\omega_i}(P_j) > \min_k v_{\omega_i}(P_k) \\ 1/\mathrm{card}\{k \mid v_{\omega_i}(P_k) = v_{\omega_i}(P_j)\}, & \text{if} \quad v_{\omega_i}(P_j) = \min_k v_{\omega_i}(P_k) \end{cases}$$

$$v_{\omega_i}(P) = \begin{cases} \max_j v_{\omega_i}(P_j) & \text{(max node)} \\ \min_j v_{\omega_i}(P_j) & \text{(min node)} \\ V_{\omega_i}(P) & \text{(leaf node)} \end{cases}$$

Then PrOM search is given by:

$$v_0(P) = \begin{cases} \max_j v_0(P_j) & \text{(max node)} \\ \sum_j v_0(P_j) \cdot \Pr(P_j) & \text{(min node)} \\ V_0(P) & \text{(leaf node)} \end{cases}$$

To investigate the behaviour of PrOM search in comparison with minimax ($\alpha$-$\beta$ search) and OM search we performed a series of experiments on random game trees. The experiments show that the number of evaluations in PrOM search is higher than the number of evaluations needed for $\alpha$-$\beta$ search or OM search. But when it is taken into account that performing PrOM search with three opponent types includes one $\alpha$-$\beta$ search (for $\omega_0$) and two OM searches (for $\omega_1$, and $\omega_2$), then the number of evaluations needed for PrOM search is not unexpectedly high.

# References


[1] Iida H., Uiterwijk, J.W.H.M., and Herik, H.J. van den (1993). Opponent-Model Search. *Technical Reports in Computer Science*, Report CS 93-03, Dept. of Computer Science, Maastricht University, Maastricht.

[2] Iida H., Uiterwijk, J.W.H.M., Herik, H.J. van den, and Herschberg, I.S. (1993). Potential Applications of Opponent-Model Search. Part 1: The domain of applicability. *ICCA Journal*, 16(4), pp. 201-208, ISSN 0920-234X