Learning Opponent-Type Probabilities for PrOM Search

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

Department of Computer Science,
Institute for Knowledge and Agent Technology,
Universiteit Maastricht, P.O. Box 616, 6200 MD Maastricht, The Netherlands.
email: {donkers,uiterwijk,herik}@cs.unimaas.nl

Abstract

Opponent-type probabilities play an important role in probabilistic opponent-model search (PrOM search): they constitute an approximation of the real opponent. The research question is: can we learn opponent-type probabilities? When the opponent uses a mixed strategy of known opponent types, it appears possible to learn opponent-type probabilities off-line from game records. Next to off-line learning, we briefly describe two approaches for on-line learning of opponent-type probabilities. The results described in this paper suggest that PrOM search may become a strong alternative for plain opponent-model search.

1 Introduction

Probabilistic Opponent-Model search [4] is an extension of Opponent-Model search (OM search). In OM search [2, 7, 8], the opponent is assumed to use a weaker evaluation function than the player's own evaluation function. OM search tries to exploit the opponent's weaknesses. In fact, the opponent is modelled as precisely as possible by his evaluation function. PrOM search tries to exploit the weaknesses of the opponent too, but uses a more sophisticated model. In PrOM search the opponent is modelled by a mixed strategy of N known opponent types ω_i ($i = 0, 1, \ldots, N-1$), each ω_i is represented separately by an evaluation function. The mixture of strategies is typified by a probability distribution $\mathrm{P}(\omega_i)$ over the opponent types. The opponent is assumed to pick at every move one of the opponent types ω_i according to probability $\mathrm{P}(\omega_i)$, and to act like this opponent for the current move.

In brief, PrOM search works as follows: at max nodes, the child node with the highest value is selected, like in Minimax. At min nodes, first, the value of the min node for every opponent type ω_i is established separately (using any Minimax-equivalent algorithm). Next, using the probabilities $\mathrm{P}(\omega_i)$, the *expected value* of the min node is computed. At leaf nodes, the player's evaluation function is used.

Figure 1 gives an example search tree in which PrOM search and Minimax are compared. The squares denote max nodes and the circles min nodes. The values inside the nodes are standard minimax values. In the example, there are two opponent types: ω_0 and ω_1 with probability 0.3 and 0.7, respectively. To the right

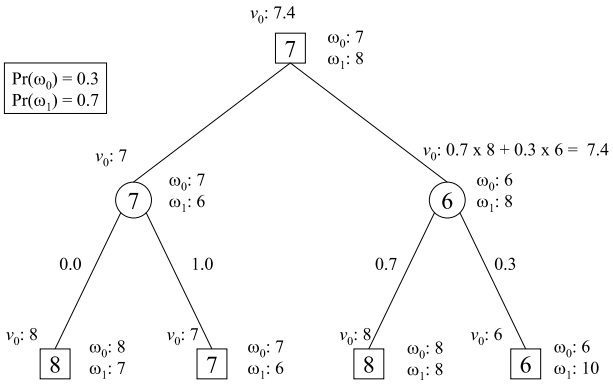

Figure 1: PrOM search and Minimax compared.

of the nodes, the minimax values for these opponent types are given. The values for ω_0 are chosen to be equal to the minimax values. The PrOM-search values are mentioned above the nodes. At the right-hand min node, ω_0 would select the right-hand child ($v_{\omega_0} = 6$), but ω_1 would select the left-hand child ($v_{\omega_1} = 8$). The expected value of this min node is equal to 7.4, which is higher than the minimax value of this node and even higher than the minimax value of the left-hand min node (7). In passing we remark that ω_0 and ω_1 both select the right-hand node from the left-hand min node. Therefore we see that the distribution for the two branches is: 0.0 and 1.0. In summary, at the root, PrOM search will select another move than Minimax does.

The mixed-strategy semantic of PrOM search serves as an *approximation* of the real behaviour of the opponent. Although it is possible to approximate the opponent by a single evaluation function tuned to the opponent's observed behaviour, as done by Carmel and Markovitch [3], the resulting plain OM search can lead to large risks, as pointed out in [5]. This is especially true when the own evaluation function has a poor quality. The mixed-strategy approach of PrOM search incorporates an implicit kind of risk avoidance [4].

The individual opponent types (e.g., the evaluation functions) to be used by PrOM search can be achieved in two ways, either by *construction* or by *learning*. Construction can be based on a model of opponent behaviour or on a set of mistakes often made. Learning can be done by using previously recorded games [6, 1]. It is a good idea to use the MAX-player's evaluation function as one of the opponent types. By convention we will denote this opponent type by ω_0. The larger the probability $P(\omega_0)$ is, the less PrOM search will yield results different from Minimax. When $P(\omega_0) = 1$, PrOM search is equivalent to Minimax.

In this paper we concentrate on learning of the probability distribution over a given set of opponent types from the observations of the moves of an unknown opponent. The research question we wish to answer is: given a set of opponent types $\omega_0 \ldots \omega_{N-1}$, what is the best probability distribution over these types to be used by PrOM search against a given target opponent Ω? This research question is in fact a learning task with two manifestations: *off-line* learning and *on-line*

learning. In off-line learning, we have available a large set of positions together with the moves that our target opponent Ω selects. During on-line learning, we only have the few moves played by Ω so far. In section 2, we focus on experiments with off-line learning. Section 3 discusses some possible ways to implement on-line learning. Finally, in section 4, we provide some conclusions and describe future research.

2 Off-line learning

The objective of off-line learning is to find a probability distribution such that PrOM search plays best against a given target opponent Ω. The definition of PrOM search strongly suggests that this probability distribution is identical to the probability distribution that best *predicts* the behaviour of Ω. This assumption is straightforward, but only real game-playing can prove its correctness. In our experiments we concentrate on finding the best predicting probability distribution.

To find the best distribution, it is not necessary to perform an extended PrOM search actually and repeatedly. We use a set-up in which the selection of a move by the opponent types $\omega_0 \ldots \omega_{N-1}$ and by the target opponent Ω is simulated by a simple selection of a number m from a given range $0 \ldots M-1$. The opponent types ω_i are modelled by a mechanism that selects a number from the range at random, and Ω is modelled by a probability distribution $\mathrm{P}(\omega_i)$ over the opponent types ω_i. The learning task now is to estimate the probability distribution $\mathrm{P}(\omega_i)$ from the observed moves produced by the ω_is and by Ω.

The simulations were conducted as follows. At every time step, (1) all opponent types ω_i selected a number m_i, (2) the target opponent Ω selected one opponent type ω_Ω, according to $\mathrm{P}(\omega_\Omega)$ and (3) Ω produced the number, m_Ω, that was selected by ω_Ω. The observed m_i's and m_Ω's lead to an estimation $\hat{\mathrm{P}}(\omega_i)$ of $\mathrm{P}(\omega_i)$ by counting the number of times that m_i agrees with m_Ω and divide this by the total number of times that any m_j agrees with m_Ω (if several distinct m_j's agree with m_Ω at the same time, they are dealt with separately):

$$\hat{\mathrm{P}}(\omega_i) = \frac{\#(m_i = m_\Omega)}{\sum_j \#(m_j = m_\Omega)} \tag{1}$$

As a measure of the quality of the distribution to be learned, a learning error ϵ is introduced, being the Euclidian distance between $\hat{\mathrm{P}}(\omega_i)$ and $\mathrm{P}(\omega_i)$. The range of ϵ is $[0 \ldots \sqrt{2}]$. In the test set-up we abstract from real games and investigate how much can be learned using only the knowledge given. In the actual games, more information is available and might be used in another series of experiments for better learning.

The moves that two opponent types (evaluation functions) in a given position select are not independent since the game-tree search influences the choice and since there are always some common elements in both evaluation functions that lead to a comparable (the same) ordering of moves. In the experiments we therefore compared sets of independently behaving opponent types to dependently behaving ones. Independent opponent types select a number on the basis of a uniform

probability distribution. The opponent types behaving dependently select the moves according to the following scheme. The first opponent type (ω_0) uniformly selects a move m_0. Opponent type ω_i, $i > 0$, selects a move from the set $\{(m_{i-1}+k) \mod M \mid k \in [-4, 4]\}$.

In actual positions, the opponent is not likely to use a mixed strategy of opponent types. Probably, the true opponent's evaluation function is not even in the set $\{\omega_0 \ldots \omega_{N-1}\}$. To investigate whether it is possible to learn a probability distribution off-line, we tested a situation in which the target opponent picked a fixed number from the given range, while the opponent types behaved dependently as described above. Since an exact learning of the probabilities is not at stake, we used the sample variance as a measure of learning quality.

In the set of experiments, the number of opponent types N varied from 2 to 20. The probability distribution $\mathrm{P}(\omega_0, \omega_1, \ldots, \omega_{N-1})$ of the opponent types was of the form $(a, b, \ldots, b)$ (where a varied between 0 and 1, and $b = (1-a)/(N-1)$). The range of available numbers, M, was fixed to 20. The learning time T varied from 10^1 to 10^5 runs. The sample size was 100 everywhere.

2.1 Experimental results

Figure 2 shows the result of the experiments for $N = 5$. As is clear from the figure, increasing the learning time from 10^2 to 10^5 decreases the variance of the learning error, but does not decrease the error. The figure shows that the learning

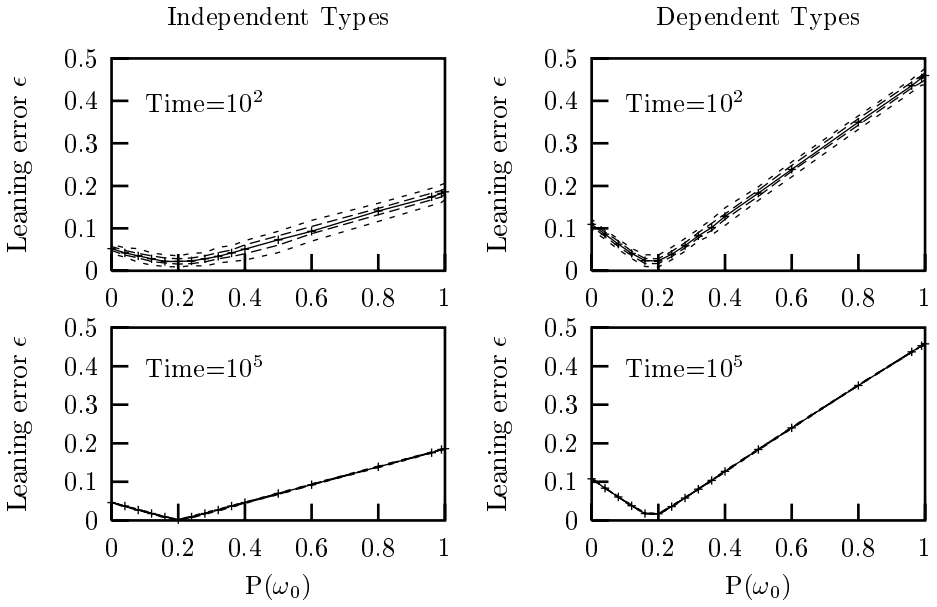

Figure 2: Learning error ϵ as a function of the opponent-type probability $\mathrm{P}(\omega_0)$ for a set of 5 opponent types, with ambiguous events. The learning time is 10^2 (top) and 10^5 (bottom).

error becomes larger if $\mathrm{P}(\omega_0)$ has a greater distance to 0.2 (equal probability for all types). This effect can be explained by the occurrence of ambiguous events, i.e., events in which more than one opponent type agree with Ω. It is possible to compute the size of this error easily for the independent case where $\mathrm{P}(\omega_0) = 1$ as follows. The probability for any opponent type other than the selected one, to agree with Ω is $1/M$, so $\#(m_i = m_\Omega)$ is on average T/M for every $i > 0$, for $i = 0$, this number is necessarily equal to T. Substituting this in equation 1 gives $\frac{\mathrm{T}/M}{T+(N-1)\cdot\mathrm{T}/M}$. This means that $\hat{\mathrm{P}}(\omega_i)$ will become $1/(M+N-1)$ for $i > 0$ and $M/(M+N-1)$ for $i = 0$. The learning error for $M = 20$ and $N = 5$ will consequently be 0.1864, which agrees with the experimental results. (For $\mathrm{P}(\omega_0) = 0.2$, the learning error is zero.) The only way to remove the learning error completely is to disregard the ambiguous events since in the case of an ambiguity any other strategy would give some positive probability to an opponent type other than ω_0.

Figures 3 and 4 shows the results for 5 opponent types when the ambiguous events are disregarded. The y-scale in this figure is logarithmic to indicate that the error is approaching zero with increasing learning time. In this case, the

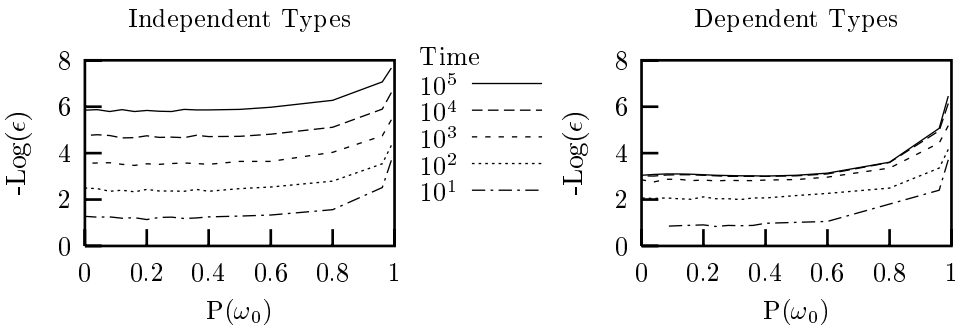

Figure 3: Learning error ϵ for a set of 5 opponent types as a function of the opponent-type probabilities, without ambiguous events.

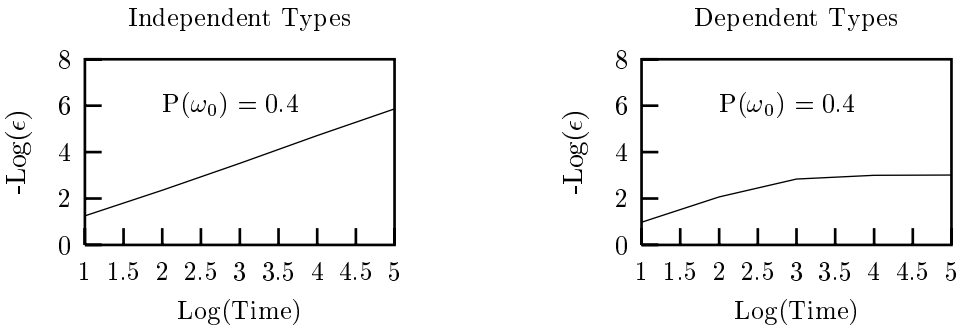

Figure 4: Learning error ϵ for a set of 5 opponent types as a function of the learning time, without ambiguous events.

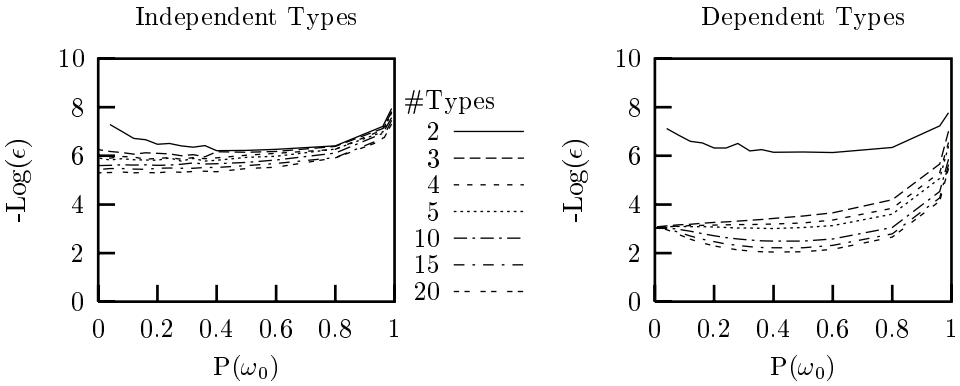

Figure 5: Learning error ϵ as a function of opponent-type probabilities and the number of opponent types for $T = 10^5$.

probabilities can be learned exactly, although in the case of dependent opponent types, there appears to be a limit to the precision.

Of course, disregarding these ambiguities decreases the learning speed, and too many of them would stop the learning. In cases in which too few rounds are counted, other, less accurate, methods for learning the probabilities must be used. In the experiments, the fraction of disregarded events with independent opponent types is constantly 18.6%, in the case of dependent opponent types, the fraction of disregarded events varies between 48.7% and 53.8%.

Figure 5 shows the results for varying numbers of opponent types. There appears to be a gap between the cases of two and more than two opponent types. This gap is larger in the case of dependent types.

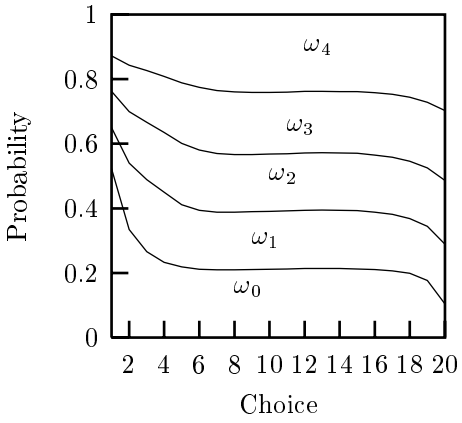

Figure 6: Learned probability distribution over opponent types as a function of opponent's choice.

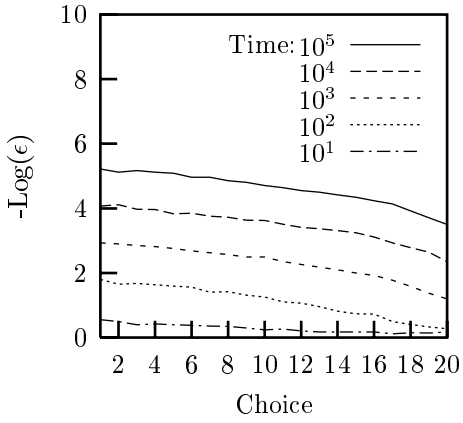

Figure 7: Learning error ϵ as a function of opponent's choice for varying learning times.

Figures 6 and 7 give the results for the case in which the opponent did not use a mixed strategy, but used a strategy that was different from any of the opponent types, e.g., selected a fixed number. From figure 7 it appears that the learning results in a stable probability distribution over the opponent types and that learning takes place with reasonable speed.

3 On-line learning

During a game, the learning of opponent-type probabilities is limited since the number of observations is low. It can, however, be useful to adapt probabilities achieved by off-line learning if the behaviour of the observed opponent is rather different from the expected behaviour.

On-line learning is performed during a game just after the opponent's move is observed. Two types of on-line learning are possible: *fast* and *slow*. In fast learning only the best move of every opponent type is used, in slow learning the search value of all moves is computed for all opponent types.

Fast on-line learning happens as follows: start with the off-line or prior obtained probabilities. At every move by the opponent do: for all opponent types detect whether their best move is equal to the actually selected move. If so, award that opponent type with a small increase of the probability. If not so, punish the opponent type. The size of the award or punishment should be not too large because this type of learning will lead to the (false) supremacy of one of the opponent types.

Slow on-line learning would be an application of the naive Bayesian learner. Again start with the off-line or a priori obtained probabilities. At every move by the opponent do: for all opponent types compute the search values of all possible moves. Compute conditional probabilities from these values. Use Bayes' rule to compute the opponent-type probabilities, given the observed move by the opponent. Use these a posteriori probabilities to update the opponent-type probabilities. In slow on-line learning, the adaptations of the probabilities should not be too large, because then it will lead to the (false) supremacy of one of the opponent types too.

It is evident that slow on-line learning adjusts the probabilities better than the fast on-line learning (among others because it uses more information).

4 Conclusions and Future Research

From the experiments and from theoretical analysis it appears that the only way to learn (estimate) the probabilities correctly is to disregard all events in which more than one opponent type agree with the target opponent. Our experimental environment shows that under this condition it is possible to learn probabilities for opponent types effectively off-line, based on stored game records. Dependently behaving opponent types increase the difficulty of learning partly because they increase the number of ambiguous events. When the actual opponent is not acting

like the model that PrOM search assumes, it is still possible to learn a probability distribution over the set of given opponent types.

Future research. Although the goal of this paper is to show that learning of opponent-type probabilities is possible, the applicability of the off-line learning has to be tested further, especially in a real domain were additional knowledge is available to speed up learning.

The subject of on-line learning of opponent-type probabilities has not yet been discussed thoroughly, but the two approaches mentioned seem to allow some learning during a game. It will however strongly depend on the game characteristics (such as game length, branching factors, mobility) whether on-line learning is meaningful.

Together with the learning of opponent-type probabilities, PrOM search can become a strong alternative to plain Opponent-Model search.

References

[1] Anantharaman, T.S. (1997). Evaluation Tuning for Computer Chess: Linear Discriminant Methods. *ICCA Journal*, 20(4), pp. 224–242. ISSN 0920-234X.

[2] Carmel, D. and Markovitch S. (1998). Pruning Algorithms for Multi-Model Adversary Search. *Artificial Intelligence*, Vol. 99, pp. 325–255. ISSN 0004-3702.

[3] Carmel, D. and Markovitch S. (1998). How to explore your opponent's strategy (almost) optimally. *Proceedings of the Third International Conference on Multi-Agent Systems, ICMAS'98*, pp. 64–71. IEEE Computer Society Press.

[4] Donkers, H.H.L.M., Uiterwijk, J.W.H.M., and Herik, H.J. van den (2001). Probabilistic Opponent-Model Search. *Information Sciences*, 135(3-4), pp. 123–149. ISSN 0020-0255.

[5] Donkers, H.H.L.M., Uiterwijk, J.W.H.M., and Herik, H.J. van den (2001). Admissibility in Opponent-Model Search. *Proceedings BNAIC 2001* (ed. B. Kröse, M. de Rijke, G. Schreiber, and M. van Someren), pp. 373–380. Universiteit van Amsterdam, Amsterdam, The Netherlands.

[6] Fürnkranz, J. (1996). Machine Learning in Computer Chess. *ICCA Journal*, 19(3), pp. 147–161. ISSN 0920-234X.

[7] Iida, H., Uiterwijk, J.W.H.M., Herik, H.J. van den, and Herschberg, I.S. (1993, 1994). Potential Applications of Opponent-Model Search. Part 1: The Domain of Applicability. *ICCA Journal*, 16(4), pp. 201–208. Part 2: Risks and Strategies. *ICCA Journal*, 17(1), pp. 10–14. ISSN 0920-234X.

[8] Iida, H., Kotani, I., Uiterwijk, J.W.H.M., and Herik, H.J. van den (1997). Gains and Risks of OM Search. In *Advances in Computer Chess 8* (eds. H.J. van den Herik and J.W.H.M. Uiterwijk), pp. 153–165. Universiteit Maastricht, Maastricht, The Netherlands. ISBN 90-6216-2347.