Combining Entropy Based Heuristics with Minimax Search and Temporal Differences to Play Hidden State Games

Gregory Calbert and Hing-Wah Kwok

Defence Science and Technology, Edinburgh, South Australia, Australia, 5111.
Greg.Calbert@dsto.defence.gov.au

Abstract

In this paper, we develop a method for playing variants of spatial games like chess or checkers, where the state of the opponent is only partially observable. Each side has a number of hidden pieces invisible to opposition. An estimate of the opponent state probability distribution is made assuming moves are made to maximize the entropy of subsequent state distribution or belief. The belief state of the game at any time is specified by a probability distribution over opponent's states and conditional on one of these states, a distribution over our states, this being the estimate of our opponent's belief of our state. With this, we can calculate the relative uncertainty or entropy balance. We use this information balance along with other observable features and belief-based min-max search to approximate the partially observable Q-function. Gradient decent is used to learn advisor weights.

Introduction

One of the first applications of temporal difference methods in reinforcement learning was in the field of game playing (Samuels, 1967). Games such as checkers, backgammon, chess and go have been used as a template under which the performance of these methods could be assessed. In this paper, we look at the combination of two current avenues of research in reinforcement learning to develop strategies in playing hidden checkers, a variant of checkers where each side has a number of hidden pieces. Both agents know at all times the location of their hidden pieces, whereas we are forced to infer a probability distribution over probable opponent piece locations.

The two avenues of reinforcement learning are those of multi-agent learning, (Shoham, Powers and Grenager, 2003) in which the actions of other agents intrinsically affect an agents states and actions, and that of reinforcement learning with hidden or partially observable states (Kaelbling, Littman and Cassandra, 1997). Our approach to game playing with hidden states is to first construct a belief of our opponents states and further to this, construct a belief of our opponent's belief of our states (as we will have hidden pieces). Given this, we are able to calculate the best move based on what subsequent move we believe the opponent will make, calculated through min-max search.

Before describing the methods used in estimating the state of the game and the development of strategy, we describe the template game, hidden checkers, used to test our algorithms. A variant of the checkers game is used in which each side has a number of hidden pieces. This means that each playing agent may move its hidden piece without communicating the location to the opposing agent. As is the case in classical checkers, each player takes turns in making a move and upon reaching the baseline of the opponent's defending side, a piece is kinged, implying additional mobility. Because we have included hidden pieces, there are four piece types that occur during the game, the unkinged or hidden unkinged and the kinged or hidden kinged pieces.

When there is uncertainty in the opponent's state, it is possible to attempt a move that violates the rules of the game. Such an attempt occurs when we make a move based on a positive probability estimate for an opponent's state, whereas the opponent is actually in another state. Thus in hidden checkers there are three move types:

1. A legal move done in accordance with the rules of checkers.
2. An illegal move, outside the rules of checkers, when both sides have perfect state information about the opponent's state.
3. An illegitimate move being an illegal move done because of the imperfect information of the opponent's state.

In the hidden checkers game, we assume an external agent communicates if a move attempted was illegitimate. If an illegitimate move was made, a player is not allowed to correct it, the pieces remain in exactly the same position as before the move was made. However if the opponent makes an illegitimate move we are not told. We also assume that both sides have perfect knowledge of the number of pieces on both sides thus are aware of success or failure if the capture of an opponent hidden piece is attempted.

When an opponent makes a move, there are three possibilities, the first being a legal move of an exposed piece. The second and third possibilities are the move of a hidden piece and an illegitimate move. Both are not

observed. The difficulty in playing this game is the determination of whether a hidden piece or illegitimate move was made.

Following this Introduction, we describe the estimator of our opponent's state in the following section. Next we develop move strategies by defining a value function with entropy balance as a feature. Subsequent we describe the methods to apply the belief based min-max version of the TD(λ) algorithm, TDLeaf(λ) for a limited number of hidden pieces (Baxter, Tridgell and Weaver, 1998). Finally we report on the advisor values of various features through the application of this algorithm, and discuss conclusions.

Belief State Dynamics

For partially observable problems, an agent possesses a belief state and transition equations that specify the dynamics of this belief state as observations are made. Thus for possible states $s_1, s_2, \cdots, s_n$ an agent has a probability distribution $b(s_1), b(s_2), \cdots, b(s_n)$ and a state transition probability $\Pr(b' \mid b, a, o)$ which determines the probability of the new belief state b' given current belief state b and action/observation a, o. It has been shown that the combination of the belief state and transition functions are statistically sufficient to describe the dynamics of the system in the future (Monahan, 1982).

For this paper it is useful to describe a single state of the checkers board to be the vector (s_u, s_o) where s_u describes the positions of our pieces (us) and s_o describes the positions of our opponent's pieces. Clearly all combinations of (s_u, s_o) are not allowed, as in checkers and in the hidden checkers variant, no two pieces can occupy the same position at the same time. Thus this state vector description is governed by the rules of the checkers game.

In the game of hidden checkers, owing to the large number of states, we do not possess such transition probabilities. Instead transitions must be constructed from a number of or assumptions of the way the opposing agent determines strategy. Instead of storing just a belief state over the opponent's possible states, we store both this distribution and conditional on one of these states, a belief state that specifies the opponent's distribution of our pieces.

Suppose $S_o = \{s_{o1}, s_{o2}, \cdots, s_{oj}\}$ are the possible states of our opponent, given our state s_u and suppose $(s_{u1}' \mid s_{ok}), \cdots, (s_{ul}' \mid s_{ok})$ are our possible states, as perceived by the opponent, conditioned on the state of the opponent being $s_{ok} \in S_o$. Then for each transition we store the vector of probability densities or belief state $b = (b(s_{o1}), b(s_{o2}), \cdots, b(s_{oj}))$ and $b'(s_{ok}) = (b(s_{u1} \mid s_{ok}), \cdots, b(s_{ul} \mid s_{ok}))$ for each $k = 1, \cdots, j$.

In essence, this means we keep a probability distribution of the positions of our opponent's pieces and an estimate of our opponent's probability distribution of our pieces. The overall belief state, denoted by b is thus a tree structure, the root being the true state of the board, the first tier nodes being our belief of the opponent's state conditioned on the true state of our pieces. Finally the leaf nodes are the estimates of our state, conditioned on the first tier opponent states. Given these estimates are probabilities, the usual simplex conditions apply.

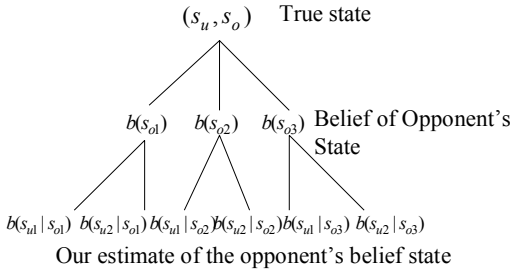

Figure 1: Belief state representation for partially observable games.

Now given that the opponent makes a legal visible move, it is easy to update our combined belief state through the use of conditional probability, however if a move is not observed, this means that either a hidden piece was moved or an illegitimate move was made. As no information is transferred as to what type of move occurred, we make the first assumption to infer the possibility of an illegitimate move:

Assumption 1: Each player attempts only to make legal moves, with an illegitimate move being made only because of incomplete information.

Let $M_o^h(s_u, s_o)$ be the set of hidden legal moves (actions) of the opponent, given our state is s_u and the opponent's is s_o. Now suppose we define a function that maps a set of possible moves onto 0 or 1 depending on the legality of each action in the set, so

$$L(M \mid s) = \begin{cases} 0 \text{ if all moves in } M \text{ are legal given } s, \\ 1 \text{ otherwise.} \end{cases} \quad (1)$$

We consider the possibility of an illegitimate move by testing the legality of each move by the opponent for which $b(s_{ui}' \mid s_{oj}) > 0$ on the board where $b(s_{oj}) > 0$. We

denote the set of these moves as $M_o^h(s_u', s_o)$. This means that if

$$L\left(\bigcup_{b(s_u'|s_o)>0} M_o^h(s_u', s_o) \mid s_o, s_u\right) = 1 \quad (2)$$

for a particular positive belief state $s_o : b(s_o) > 0$, then there is the possibility of an illegitimate move. Intuitively, this process considers all possible moves of our opponent based on our estimate of its beliefs of our state distribution. Under assumption one, the opponent considers all of these moves to be legal, however in the true state (which we know as they are our pieces) some are illegal, thus illegitimate.

Once we have ascertained the possibility of an illegitimate move for a particular opponent state belief (an element of the first level of the tree structure), we must assign probabilities to the combined set of hidden moves, found from considering all hidden moves in the set

$$M_o^h(s_u, s_o), \text{for } b(s_o) > 0. \quad (3)$$

and the illegitimate opponent move in which the spatial state of the opposition pieces remains the same. Again we must evoke an assumption in order to assign these probabilities.

Assumption 2: Hidden or illegitimate moves are made to maximize our uncertainty of the state of the opponent.

Suppose that

$$M_o^h(s_u, s_o) \cup \{m_{illeg}\} = \{m_1, m_2, \cdots, m_n, m_{illeg}\}$$

are the possible hidden or illegitimate moves of our opponent. In order to maximize our uncertainty of the state of the opponent, we assign probabilities to all moves in order to maximize the entropy,

$$H(p(m_1), p(m_2), \cdots, p(m_n), p(m_{illeg})), \quad (4)$$

which owing to its convexity of the entropy function, assigns equal probabilities to each of the moves.

By assumption 2, assigning equal probabilities to each move generates a transition probability from old opponent to new opponent states, $\Pr(s_o' \mid s_o, m)$ with a component of the new belief state (up to a normalizing constant) being

$$b(s_o') = \sum_m \sum_{s_o} \Pr(s_o' \mid s_o, m) b(s_o) p(m). \quad (5)$$

Once the first tier nodes are updated, the root nodes (our estimate of the opponent's estimate of our state) are also updated, which means we calculate the probabilities

$$b(s_u' \mid s_o') = \sum_{b(s_o)>0} \Pr(s_u' \mid s_o') b(s_o'). \quad (6)$$

Minimax Search

At a particular belief state, represented by a belief tree we conduct a minimax search to find the best possible move. In essence, this is done by considering all moves in the set

$$\bigcup_{b(s_o)>0} M_u(s_u, s_0). \quad (7)$$

this being the total number of moves available where the probability of the opposition being in state s_o is greater than zero (M_u denotes the set of moves available to us). Leaf nodes of the minimax search are evaluated by an approximation to the value function. This is constructed in two parts. First we take the expectation over a number of chosen features for each opposition state such that $b(s_o) > 0$. Further we add to this expected set of features another feature describing our estimated balance of uncertainty in state information. This term is called the entropy balance and has the form

$$E(H(b_u')) - H(b_o), \quad (8)$$

which is the expected difference between our estimate of our opponent's uncertainty of our state ($H(b_u')$) and our uncertainty of our opponent's state ($H(b_o)$).

Our value function, for our belief tree can then be written as the inner product of the our belief state of our opponent and a vector of observable and information based features for each state

$$V(b) = b \cdot \{V_{\text{obs}}(\mathrm{w}) + w_I (H(b_u) - \log(b_o))\} \quad (9)$$

where the vector of weighted observable features for each probable opponent state, $b(s_{oj}) > 0$, (w, ϕ are vectors also) is

$$V_{\text{obs}}(\mathrm{w}) = (\mathrm{w}.\phi(s_{o1}), \mathrm{w}.\phi(s_{o2}), \cdots, \mathrm{w}.\phi(s_{oj})), \quad (10)$$

the entropy of our estimate of the opponent's estimate of our state is

$$H(b_u) = -\sum_{b(s_o)>0} b(s_u \mid s_o) \log(b(s_u \mid s_o)) \quad (11)$$

and $\log(b_o) = (\log(b(s_{o1})), \log(b(s_{o2})), \cdots, \log(b(s_{oj})))$.

We note that the value function satisfies the general structure of the value function for a POMDP (Kaelbling, Littman and Cassandra, 1998).

In the games we simulated, the features included a number of balance terms, these being the difference between material numbers of our side and the opponent's. We considered the balance of exposed unkinged, kinged pieces as well as the balance of hidden unkinged and kinged pieces. The components of the combined vector of weights, (w, w_I) are often termed advisors, and the scalar weight w_I is called the *information theoretic advisor* (Bud, et al. 2001) is it considers the balance of uncertainty one side has over another as measured by the entropy difference. In (Bud, et al. 2001) the concept of an information theoretic advisor was introduced, however no attempt was made to determine the correct advisor value. Further to this, it should be noted that this decomposition

of the value function into observable and information features is similar to "weighted entropy control" seen in (Cassandra, 1998).

Having defined the value function, let us define the value function found from applying minimax search at varying search depths. There are several approaches to minimax search when there is partial information about the opponent, as is illustrated by a two-ply search. One way, which we term the risk sensitive approach is to back-up the minimum value of each possible move by the opponent. Alternatively one could back-up an expected value, done over an appropriate probability distribution. In the case of a two-ply search, the Q-function for our move m would be

$$Q(b,m) = \mathrm{E}(V), \tag{12}$$

where V is the value function after the opponent's move. An appropriate probability distribution over which this expectation might be generated is a modified Gibb's distribution, where

$$\Pr(V_i) = \exp((V_i T)^{-1}) \Big/ \sum_j \exp((V_j T)^{-1}). \tag{13}$$

The limit as $T \to \pm\infty$ corresponds to equal weights of all opponent's moves, making no assumption (hence risk-prone) about our opponent's intentions. In the limit as $T \to 0$ corresponds to the risk averse assumption that our opponent tries to minimise our value function, as the probabilities are all zero except at the minimum value. In this paper we took the risk-averse assumption, that of the opponent attempting to minimise our value function.

Minimax search proceeds along the same lines as that of deterministic minimax tree search except that it is done over a series of belief trees. Over some belief states our move may be legal or illegitimate, and therefore we back-up the expected values of such a move, conditioned on it either being legal or illegitimate (Russell and Norvig, 2002).

Our experiments have shown that for a small number of hidden pieces on each side, the branching factor of moves is not as great as some have thought, as there is some dependence between moves. The branching factor of moves is also restricted because of the requirement of compulsory captures in checkers. The following table lists the estimated branching factor for a small number of hidden pieces, given two-ply search.

Number of Hidden Pieces	Branching factor
Zero versus zero	7.6
One versus one	8.3
Two versus two	8.5
Three versus three	9.5

Figure 2: Branching factor as a function of hidden piece number.

The Application of Belief Based TDLeaf(λ)

Having defined a value function dependent on a number advisor values, we use reinforcement learning to find appropriate values for these weights. We approximate the Q-function, defined as

$$Q(b,m) = \Pr(\text{win} \mid \text{belief state } b, \text{move } m). \tag{14}$$

Clearly, the Q-function at terminal belief states, denoted by b_T has the values of one for a win, and zero for a loss. We define the terminal value for a draw as a half. Functional approximation is used to estimate the Q-function, with

$$Q \approx \tilde{Q}(b,m) = sig(Q_n((b,m,w)) \tag{15}$$

where $Q_n(b,m,w)$ is the Q-function of a n-ply minimax search from belief state b applying move m with advisor weights w. We smooth the approximation by applying the sigmoidal function $sig(x) = (1+\exp(-x))^{-1}$.

The approximation to the optimal Q-function is done via on-line gradient decent, in which the advisor weights are altered with Q-function version of the TDLeaf(λ) algorithm (Baxter, Tridgell and Weaver, 1998) ($TD(\lambda)$ with probabilistic minimax tree search) :

$$\Delta w = \alpha(\tilde{Q}(b_{t+1}, m_{t+1}) - \tilde{Q}(b_t, m_t)) \sum_{k=t}^{0} \lambda^{t-k} \nabla_w \tilde{Q}(b_k, m_k). \tag{16}$$

The value of $\lambda = 0.75$ was chosen as a balance between altering the advisor values for long term prediction of the terminal state and prediction of state-action value of subsequent states, short term prediction. One can show that the functional approximation, of the probability of winning or losing, is shifted towards the terminal state, ($\tilde{Q}$ moves towards $Q(b_T, \bullet)$) when $\lambda = 1$, and shifted towards subsequent states, ($\tilde{Q}(b_t, m_t)$ moves towards $\tilde{Q}(b_t, m_t)$) when $\lambda = 0$ (Sutton, 1993). In other examples from the cited literature of learning in the game of chess, authors have found that values between 0.7 and 0.8 have the greatest success, though more research is required on this topic (Beal and Smith, 1997).

As is the case of general machine learning, the rate of which the advisor values change is determined by the coefficient α. In general, this coefficient is dependent on the number of games played during the learning process. We slowly anneal this coefficient sequence to zero. If n is the number of games played, then the sequence $\alpha_n \to 0$ in such a way that satisfies the stochastic convergence conditions given in (Barto and Sutton, 1999).

In our experiments to learn advisor values, initially all were set to unit (1.0) value. A set of two thousand five hundred games were played to learn the values of advisor weights for exposed unkinged or kinged piece balance as well as hidden unkinged or kinged piece balance and the

value of the information theoretic advisor for the entropy or information balance. Each side possessed one hidden piece at the commencement of the game. Minimax search was conducted at three-ply. The following graph shows us the outcomes.

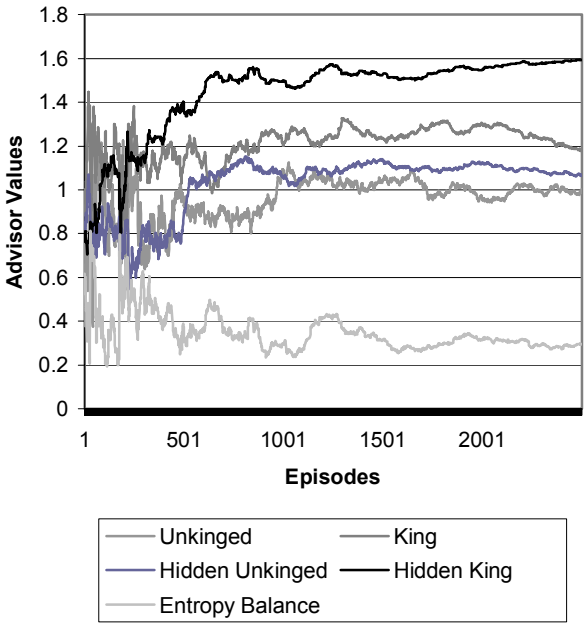

Figure 3: Advisor values for 2500 games, normalized with an exposed unkinged piece of unit value.

On observation of the graph several important points may be noted. In ascending order, we have the advisor values for entropy balance (0.3), unkinged (1.0), hidden unkinged (1.1), kinged (1.2) and hidden kinged (1.3) pieces respectively. Intuitively we would accept such results, as kings have the advantage of mobility, hidden pieces stealth, thus the combination of the two generates the highest value.

At this stage of our research, we would like to consider several ways to grade the performance of policies based on our learnt advisor weights. First it would be interesting to conduct a series of experiments against human players trained with a number of hidden pieces on the board or capture their expertise through likelihood methods (Jansen, et al., 2000). Second, a comparison with other methods for developing strategies, such as heuristics and policy gradient methods would be helpful.

Discussion and Conclusions

In this paper, we developed a series of methods of playing a game, when there is partial observability of the opponent.

The impact of large state numbers and hidden opposition dynamics made the use of various assumptions on the transitions of pieces critical, leading to a simple, risk-averse model of belief state dynamics. We stored a model of the opponent's belief of our states for two principal reasons. First one must consider, with imperfect information, the possibility that an illegitimate move is made. Second, an estimate of the opponent's entropy was calculated, enabling us to include the entropy balance as a feature of the game.

We chose a hidden variant of checkers, because it is a game of intermediate strategic complexity, even with perfect information of the opponent's state. Other games of imperfect state information have been studied. Examples include poker and bridge (Frank and Basin, 2001). However these games are strategically simple given perfect information.

If our approach is to scale up to games of further strategic complexity, several research avenues need to be pursued. In games with higher strategic complexity, the higher branching factor combined with hidden states results in an explosion in the number of moves that can be made in each round. It will therefore we necessary to prune this probabilistic game tree in a systematic way. Through forward pruning methods are considered unreliable in the perfect information context, they should be explored with hidden states, to determine sensitivity and bounds on such methods (Bjornsson and Marsland, 2000).

Another exciting avenue of research is to approximate the optimal policy directly, rather than finding a policy through an estimate of the Q or value functions. These methods, termed direct or policy gradient methods should be explored, as mixed strategies may be found, as opposed to just the restriction on pure strategies found with the current approach (Baxter and Bartlett, 2001). Policy gradient methods also easily extend to partially observable and multi-agents domains.

In summary, a combined belief state of our opponent's pieces and model of our opponent's beliefs allowed us to estimate the dynamics of a game with hidden pieces. In turn, we decomposed our approximate value function into observable and information based features. Combined with probabilistic min-max search and gradient decent, we were able to learn advisor values for observable and information based features. Future research will focus on probabilistic tree pruning and policy gradient methods for scaling to domains of increased complexity.

Acknowledgements

We are grateful to Jason Scholz, Drew Mellor and a number of anonymous referees for taking the time to look through the paper and point out typos, grammar and concept corrections.

References

Barto, A.G. and Sutton, R.S. 1998. *Reinforcement Learning: An Introduction.* MIT Press.

Baxter, J.; Tridgell, A.; and Weaver, L,; 1998. TDLeaf(λ): Combining Temporal Difference Learning with Game Tree Search. *Proceedings of the Ninth Australian Conference on Neural Network:* 168-172.

Baxter, J. and Bartlett, P.L.; 2001 Infinite Horizon Gradient Based Policy Search. *Journal of Artificial Intelligence Research.* 15:319-350.

Beal, D.F. and Smith, M.C..; 1997 Learning Piece Values Using Temporal Differences. *ICCA Journal* 20(3): 147-151.

Bjornsson, Y. and Marsland, T.A. 2000. Risk management in game tree pruning. *Information Sciences Journal* 122(1): 23-41.

Bud, A.E.; Albrecht, D.W.; Nicholson, A.E.; and Zukerman, I. 2001. Information Theoretic Advisors in Chess. *AI and STATISTICS 2001, Eighth International Conference on Artificial Intelligence and Statistics.* Florida, U.S.A.

Cassandra, A.R. 1998. *Exact and Approximate Algorithms for Partially Observable Markov Decision Processes.* Unpublished Ph. D. Thesis. Brown University.

Frank, I. and Basin, D. 2001. A theoretical and empirical investigation of search in imperfect information games. *Theoretical Computer Science.* 252:217-256.

Jansen, A.R.; Dowe, D.L.; Farr, G.E., 2000. Inductive Inference of Chess Player Strategy. *Pacific Rim International Conference on Artificial Intelligence.*61-71.

Kaelbling, L.P.; Littman, M.L.; and Cassandra, A.R. 1998. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence* (101): 1-45.

Monahan, G.E. 1982. A Survey Of Partially Observable Markov Decision Processes: Theory, Models, and Algorithms. *Management Science* 28(1):1-16.

Russell, S. and Norvig, P. 2002. *Artificial Intelligence: A Modern Approach,* New Jersey: Prentice-Hall.

Samuel, A.L. 1967. Some studies in machine learning using the game of checkers. II-Recent progress. *IBM Journal on Research and Development*: 601-617.

Shoham, Y.; Powers, R.; and Grenager, T. 2003. Multi-Agent Reinforcement Learning: a critical survey. Preprint found in www.stanford.edu/~grenager/.

Sutton, R. S. 1988. Learning to Predict by the Methods of Temporal Differences. *Machine Learning* 3: 9-44.