Tree Search in Two-Player Games – Using Bounded Common Interest to Prune

H. (Jeroen) H.L.M. Donkers

Department of Computer Science, Institute for Knowledge and Agent Technology, Universiteit Maastricht.
PO Box 616, 6200 MD Maastricht, the Netherlands. Email: donkers@cs.unimaas.nl.

Abstract

A special case of two-player games is proposed for symmetric opponent modelling: games with bounded common interest. It constitutes a new type of heuristic search as a further development of Opponent-Model search. The bound on the common interest of both players causes a bound on the possible equilibria and by that on positive and negative effects of the approach. The bound also allows pruning in an Alpha-Beta-like search algorithm.

Keywords: Games, Opponent Modelling, Pruning.

1. Symmetric Opponent Modelling

The concept of a Nash equilibrium [7] in two-player games of perfect information is based on the *rationality* of both players (i.e., the intention to maximize their pay-off), as well as on the principle of *common knowledge* [1]. The latter roughly means that both players know each others pay-offs and know that the other does so too. In zerosum games, common knowledge is assumed implicitly since both players receive the same pay-offs (albeit with opposite signs). Nash equilibria for games in extensive form (i.e., game trees) are determined by the Minimax procedure [8].

In the case of board games like chess, perfect information is at hand and the pay-offs of both players are opposite, so a zerosum game is the correct model and the Minimax procedure theoretically will lead to an optimal strategy for both players. Unfortunately, most board games are too large to apply Minimax. Therefore, heuristic search methods are used in practice. The standard approach to computer game-playing is to apply Minimax to *reduced* game trees in which pay-offs are replaced by heuristic evaluation values. Much effort is put into developing good evaluation functions and in optimizing the tree search. For actual game-playing, it is sufficient to determine the best next move. To find this move, only part of the game tree has to be inspected, the remainder is 'pruned away'. However, since reduced games are used, it is not obvious that these reduced games should be zerosum games, and several alternative approaches have been proposed [5].

One of the alternative approaches is called Opponent-Model search [2,3,4]. It is based on the idea that knowledge of the opponent's strategy should be exploited. In Opponent-Model search, the knowledge symmetry of the zerosum games was abandoned since only one of the players was assumed to have knowledge of the opponent's pay-offs. Carmel and Markovitch [2] proposed an extension to Opponent-Model search in which both players use opponent models of each other, but in their model, one of the players still has more knowledge than the other.

However, it might be more natural to assume that both players use an opponent model of each other of which they are mutually aware. In the context of heuristic search it means that both players agree that they have different (i.e., non-opposite) evaluation values for positions. The key concept is *common interest*. Evaluation values are based on many factors of a position. Some of these factors are pure competitive, such as the number of black pieces on a chess board, other factors are of interest of both players. Carmel and Markovitch [2] give the example of material exchange in Checkers. Another example is the degree to which a Chess position is 'open' or 'closed'. An open position (in which many pieces can move freely) is favoured by many players over closed positions. So, the openness of a position is a common interest of both players. Assume that the competitive factors of a position measure S and the common-interest factors C, then the value for the first player would be $C + S$. In the standard zerosum approach, the opponent would be assumed to use value $-(C + S)$ for the same position, which would mean that the opponent would award the common interest of the position with $-C$. However, it seems more plausible that the second player uses value $C - S$ for the position. In the model of Carmel and Markovitch [2], only one of the players is assumed to be aware of this fact. However, why should we not assume knowledge symmetry and let both players agree on the size of C and S?

When the two players receive different pay-offs (e.g., $C + S$ and $C – S$) and these pay-offs are common knowledge, we achieve a *non-zerosum* game of perfect information. In such a game there is both opponent modelling and knowledge symmetry, leading to symmetric opponent modelling. It should be noted that in any non-zerosum game, it is possible to describe the pay-offs in terms of competitive and common-interest factors. If player 1 receives A and player 2 receives B, the common interest C is equal to $(A + B)/2$ and the competitive part S is equal to $(A – B)/2$.

In section 2 we will look into non-zerosum games and their equilibria. A special class of non-zerosum games, namely bounded common interest (BCI) games will be presented in section 3. We will prove an important property of these games. Section 4 is dedicated to pruning in BCI games. We discuss gain and risk of applying BCI games in section 5 and in section 6 we end the paper with some conclusions.

2. Non-zerosum Games

Two-player non-zerosum games of perfect information in extensive forms are studied well in game theory [9]. The concept of a Nash equilibrium in these games is based on rationality and common knowledge, just as in zerosum games [1]. Moreover, the procedure for obtaining equilibria (i.e., backward induction) is basically the same as the Minimax procedure [8]. The main difference is that in a non-zerosum game, Nash equilibria with different values might co-exist, whereas in zerosum games, all equilibria have the same value (which is usually called the Minimax value). Backward induction can produce all (subgame-perfect) equilibria of a non-zerosum game in one pass [9]. For game-playing, a single one suffices.

```
BackInd(node p) returns [m, v1, v2]
1. if p is leaf, return [NIL, V1(p), V2(p)]
2. max ← -inf ; L ← {}
3. for all moves m at p
4.     [ _, v1, v2] ← BackInd (m(p));
5.     if (v_player(p) > max) L ← {[m, v1, v2]} ;
6.                            max ← v_player(p)
7.     if (v_player(p) = max) L ← L ∪ {[m, v1, v2]}
8. select [m, v1, v2] ∈ L
9. return [m, v1, v2]
```

Fig. 1: Backward induction to select a single equilibrium.

The recursive procedure as presented in Figure 1 determines the pay-off values of a single equilibrium (v_1,v_2) plus the move m at the root of the game which leads to that equilibrium. Which equilibrium actually is returned, depends on the selections made in line 8. Although the pseudo-code might suggest so, the procedure needs not to be executed in a depth-first manner.

The existence of multiple equilibria adds a fundamental difficulty to game-tree search in non-zerosum games. Namely, which selection should be made in line 8 when more than one entry is present in set L? Figure 2 presents the game tree of a small example game. The squares indicate player 1, the circles player 2. Observe that both players are maximizing their pay-offs. This game has two equilibria: R,BC and L,BD (the characters indicate the choices of both players). The first one has values (7,8) and the second (8,6). In the right-hand branch of the tree, player 2 seems indifferent between the choices C and D (it seems to be a tie-break), but the choice of player 1 between L and R depends on this decision. By selecting carefully between C and D, player 2 can increase his own pay-off from 6 to 8. When the game tree becomes more complex, the effects of the choices are more difficult to predict since the number of equilibria can grow quickly. Moreover, the differences between equilibria can be arbitrarily large. It means that selection of equilibria is an important issue.

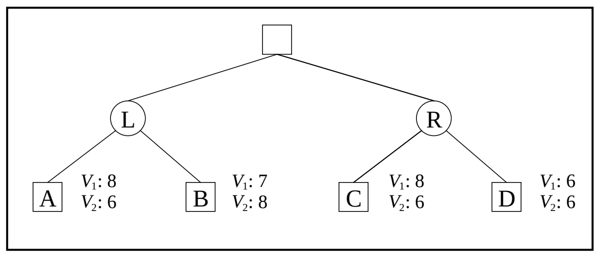

Fig. 2: Example game tree with two equilibria.

One way to deal with equilibria selection is to ignore global effects and use a rule such as: assume that a player, when breaking a tie, always selects the move with the lowest (or highest) value for the other player. Such local schemes for equilibrium selection are proposed in multi-player games (e.g., Max^n and Paranoia [10]). As follows from the example in Figure 2, such a local procedure is not guaranteed to find a globally good equilibrium.

The effect of a seemingly indifferent local choice having global effects in the tree does not appear in the classical zerosum games. The heuristic search algorithms derived from the Minimax procedure, such as $\alpha\beta$ search [6], do not take the problem of equilibrium selection into account. However, it does not mean that heuristic search techniques are of no use to non-zerosum games.

3. Bounded Common Interest

As stated in section 1, the pay-offs in any two-player non-zerosum games can be expressed in a common-interest part C and a competitive part S, such that the pay-off for player 1 is $C + S$ and for player 2 is $C - S$. It is clear that if C is zero in all pay-offs, the game is in fact a zerosum game. Therefore, C must be larger than zero in at least one pay-off. When S is zero for all pay-offs, the game becomes a combined maximization problem. We will not consider these degenerated games.

For heuristic-search purposes it is useful to define games with *bounded common interest*. These are non-zerosum games in which the value of C is bounded to an interval $[-B/2, B/2]$, where B is (much) smaller than the largest absolute value of S in any pay-off. The profit of using this bound is that it allows for pruning during game-tree search since the difference between the value for player 1 and 2 in every equilibrium is restricted to B. Moreover, the range of values that equilibria can take on is restricted, as we will show below. Below, we will call this type of games: BCI games.

Forcing a bound on C means that for any leaf position, the sum of v_1 and v_2 is bounded, namely: $-B <= v_1 + v_2 <= B$. We will prove by induction to the depth of the game tree that the ranges of values for v_1 in any subgame equilibrium are bounded to $v^* \pm B(d\text{-}1)$, and that the values for v_2 are bounded to $-v^* \pm Bd$, where d is the depth of the tree. Observe a game tree of depth 1. For all leaves having maximal value v^* for v_1, the value for player 2 of that move is bounded to $-v^* \pm B$. So the hypothesis holds for d=1. In a game tree of depth d, for each of the m subgames at depth 1, separate ranges around $v^*(1)$, $v^*(2)$, … $v^*(m)$ exist for their equilibria, each having the same width, namely $v^*(i) \pm B(d\text{-}1)$ for player 1 and $-v^*(i) \pm B(d\text{-}2)$ for player 2. At the root of the tree, player 1 will pick a move with maximum value v^*. This move can be any of the m moves available, but its value will always lie inside the range of $v^*(j) \pm B(d\text{-}1)$ where $v^*(j)$ is maximal among all $v^*(i)$. If v^* would be smaller than $v^*(j) - B(d\text{-}1)$, it could not have been selected, because any equilibrium in game j would be preferred. This means that the range of values for v_1 at the root are bounded to $v^* \pm B(d\text{-}1)$. Since the value of v_2 in any equilibrium is bounded to $v_1 \pm B$, the range for values v_2 for all equilibria at the root is bounded to $-v^* \pm Bd$.

The bound on the possible values for the equilibria means that the 'damage' of selecting a non-optimal equilibrium is bounded to $B(d\text{-}1)$. This implicates that a heuristic selection of equilibria based on local decisions is less harmful for a game in which the common interest is small than for a game with unrestricted common interest.

In computer game-playing, the move to select at the root of the game tree is more important than the exact values of an equilibrium. Since multiple equilibria can exist, it is possible to have more than one move at the root that can lead to an equilibrium. The value ranges for equilibria of the subgames at depth 1 can be used to decide which moves at the root never can lead to an equilibrium. Assume that value ranges at depth 1 have the same width for v_1 ($v^*(i) \pm B(d\text{-}1)$). Let move j lead to the maximum lower bound $D = v^*(j) - B(d\text{-}1)$. Any move i that leads to an upperbound $v^*(i) + B(d\text{-}1)$ that is smaller than D can not lead to an equilibrium, and will never be selected at the root.

4. BCI - Alpha-Beta Pruning

The bound on the sum of v_1 and v_2 in any equilibrium makes it possible to construct an Alpha-Beta [6] type of algorithm for BCI games that, together with a local heuristic equilibrium selection, can be used in computer game playing. Our approach is based on the pruning algorithm ($\alpha\beta^*$) that is described by Carmel and Markovitch as an improvement of their M* algorithm [2]. It is also related to the pruning algorithm for Max^n as described by Sturtevant and Korf [10]. However, having only 2 players enables more pruning than in the general case.

```
BciAb(node p, α, β) returns [m, s, c]
if (p is leaf)  s ← S(p) ; c ← C(p)
        if  (player(p)=2)   s ← –s
        return [NIL, s, c]
s* ← -inf ; c* ← 0 ; L ←{}
for all moves m at p
   a  ← s* –  c* ; if  (a<α)   a ←α
   [ _ , s, c] ← BciAb (m(p), β, a) ; s ← –s
   if (s + c > s* + c*) L ← {[m, s, c]} ;
                              s* ← s ; c* ← c
   if (s + c = s* + c*) L ← L ∪ {[m, s, c]}
   if  (s ≥ β + B)   L ← {[_, s, c]} ; break
select [m, s, c] ∈ L
return [m, s, c]
```

Fig. 3: BCI $\alpha\beta$ pruning.

The algorithm in Figure 3 shows the pseudocode for $\alpha\beta$ pruning in BCI games. The algorithm is presented using *s,c*-notation for the pay-offs ($v_1 = c + s$, $v_2 = c - s$), since it makes the pruning mechanism more clear. Furthermore, the algorithm uses the negamax principle, but only for the s values since both players maximize c values.

The justification for pruning is as follows. In standard Alpha-Beta pruning [6], the current value for one player is used to predict the value for the other player. This prediction is used to deduce whether the node under consideration cannot contribute to the equilibrium and can be pruned. The prediction in the case of zerosum games is trivial since the values are opposite. In BCI games, such prediction is also possible since we know that both values lie within a distance of B from each other. During the deduction process, the bound B has to be taken into account.

The actual pruning takes place in line 11, but the pruning is prepared in line 6 where the value of a is used to set the β limit in the next search level. The bound on common interest B occurs only in line 11 where it is used to extend the upper limit. When B is zero, all c's will be zero too and the algorithm boils down to plain $\alpha\beta$ Search (in negamax notation).

Depending on the selections made in line 12, the algorithm will find any equilibrium that backward induction will find, since pruning only takes place for branches that will not influence the current equilibrium.

It is obvious that the amount of pruning is related to the size of B since this bound is used to enlarge the effective $\alpha\beta$-window. With every level of depth the window is enlarged, leading to less pruning as search proceeds. If B is zero, pruning is optimal (for the given ordering of the tree), and when B is very large, no pruning might take place at all.

The algorithm can be enhanced by, for example, move ordering, iterative deepening, and transposition tables. The effectiveness of zero-windows (and thereby of the PVS and MTD frameworks) needs special attention since a zero-window is opened up by B at every search level.

5. Gain and Risk in BCI games

When a BCI game is used as a means of heuristic search, it can lead to a higher value than when using Minimax. In the example of Figure 2, player 1 can reach a value of 8 if player 2 cooperates. If Minimax is used (with $v_2 = -v_1$), player 1 will receive a value of 7 only. Our expectation is that obtaining a higher value with a BCI game corresponds to more effective game play since common interest of the two players is taken into account. However, the size of the gain is bounded by the bound on common interest.

The use of BCI games assumes that the pay-offs and the rationality of both players are common knowledge. In actual computer game-playing, such common knowledge is not very plausible. This constitutes a risk in using BCI games because the opponent can (and will) behave different from expected. Moreover, the risk of estimation errors might cause similar problems as in Opponent-Model search [4]. The effect of both risk types depends on the size of the bound on common interest.

6. Conclusion

We showed that two-player non-zerosum games of perfect information can be used for symmetric opponent modelling. A fundamental difference with the standard zerosum games is that several equilibria can exist in one game and that selecting a good equilibrium is very hard. We proved that when bounded common interest (BCI) is assumed, the range of values that equilibria can take on is also bounded. Furthermore, BCI games allow pruning during the determination of equilibria in a game tree.

BCI games offer an alternative to Minimax-based algorithms and to Opponent-Model search in heuristic tree search, but experimental evidence has to be collected on the practical usability and effectiveness of the approach. The BCI game also offers an opportunity to apply a range of search techniques from Artificial Intelligence to a class of games that is of interest to a broader audience than the traditional one.

7. References

[1] R.J. Aumann, Backward Induction and Common Knowledge of Rationality, *Games and Economic Behavior*, Vol. 8, pp. 6–19, 1995.

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

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

[4] H.H.L.M. Donkers, J.W.H.M Uiterwijk, and H.J. van den Herik, Admissibility in Opponent-Model Search, *Information Sciences*, Vol. 154, Nos. 3–4, pp. 119–140, 2003.

[5] A. Junghanns, Are there Practical Alternatives to Alpha-Beta?, *ICCA Journal*, Vol. 21, No. 1, pp. 14–32, 1998.

[6] D.E. Knuth and R.W. Moore, An Analysis of Alpha-Beta Pruning. *Artificial Intelligence*, Vol. 6, No. 4, pp. 293–326.1975.

[7] J. Nash, Equilibrium Points in n-Person Games, *Proceedings of the National Academy of Sciences*, Vol. 36, pp. 48–49, 1950.

[8] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, Princeton, NJ, 1944.

[9] M.J. Osborn, *Am Introduction to Game Theory*, Oxford University Press, New York, NY, 2004.

[10] N.R. Sturtevant and R.E. Korf, On Pruning Techniques for Multi-Player Games, *Proceedings AAAI 2000*, pp. 201–208, 2000.