Learning to take risks

Sandip Sen & Neeraj Arora
Department of Mathematical & Computer Sciences
600 South College Avenue
Tulsa, OK 74104-3189
University of Tulsa
Phone: 918-631-2985
e-mail: sandip@kolkata.mcs.utulsa.edu

Abstract

Agents that learn about other agents and can exploit this information possess a distinct advantage in competitive situations. Games provide stylized adversarial environments to study agent learning strategies. Researchers have developed game playing programs that learn to play better from experience. We have developed a learning program that does not learn to play better, but learns to identify and exploit the weaknesses of a particular opponent by repeatedly playing it over several games. We propose a scheme for learning opponent action probabilities and a utility maximization framework that exploits this learned opponent model. We show that the proposed expected utility maximization strategy generalizes the traditional maximin strategy, and allows players to benefit by taking calculated risks that are avoided by the maximin strategy. Experiments in the popular board game of Connect-4 show that a learning player consistently outperforms a non-learning player when pitted against another automated player using a weaker heuristic. Though our proposed mechanism does not improve the skill level of a computer player, it does improve its ability to play more effectively against a weaker opponent.

Keywords: Opponent model, game playing, learning models

Introduction

Though competition is not a desirable characteristic of multiagent domains, it is often an inescapable one. In situations where multiple agents are competing for resources, benefits, power, etc., agents who can identify and exploit the weaknesses of their competitors are likely to further their interests at the expense of those competitors. Whether it be game situations or business settings, the ability to acquire knowledge about the competition and to utilize or to exploit this information can be the key to successful performance.

Though explicit communication can provide invaluable information that aids the development of a more comprehensive model of the opponent, the uncertainty regarding the reliability of the communicated information makes the learning agent vulnerable.

In this paper, we address the problem of learning in the absence of explicit communication in adversarial domains. Board games provide a canonical format for a number of adversarial domains. We propose and evaluate a method for learning and exploiting opponent models in game playing programs (which can be represented as autonomous agents).

Game playing programs

Game playing is considered a representative intelligent activity and studies in game theory have found repeated applications in a variety of social and economic settings (Eichberger 1993). Most AI approaches that study learning in games address the problem of learning to play better (Samuel 1959; Tesauro 1992; Tesauro & Sejnowski 1989). A number of different approaches to improving the level of play with experience has been attempted in a wide variety of games like backgammon, bridge, checkers, chess, go, othello, etc. However, another typical human characteristic is to be able to recognize and exploit the weaknesses in the playing strategy of its competitor. In this form of learning, the human is not improving its level of play, but applying its knowledge of the game to learn about the shortcomings of its opponent. Then it attempts to take advantage of those shortcomings by taking calculated risks. In the following we outline our plan to enable a game playing agent to attempt the same.

Game theoreticians have espoused the use of the *maximin strategy* for matrix games to arrive at equilibrium points (Luce & Raiffa 1957). This strategy recommends that a player plays a move that is guaranteed to give it the maximum value if the opponent plays its optimal move. Let two players A and B be required to choose from move sets $\{\alpha_1, \alpha_2, \ldots\} = \alpha$ and $\{\beta_1, \beta_2, \ldots\} = \beta$ respectively, and the utility received by A for a (α_i, β_j) pair of moves be $u(\alpha_i, \beta_j)$. Then the *security level* for A of move α_i is given by $\min_{\beta_j} u(\alpha_i, \beta_j)$, i.e., the minimum value A can receive irrespective of what B chooses to play. The maximin strategy recommends choosing that move which has the maximum security level: $\arg\max_{\alpha_i \in \alpha} \min_{\beta_j \in \beta} u(\alpha_i, \beta_j)$. The maximin strategy is extremely conservative in its selection of moves. Note that if the opponent was not playing optimally, then A will get more than its security level for any given move. In these situations, it is preferable to take calculated risks than strictly adhere to the maximin strategy. As one book aptly describes it: "... minimax seems somewhat unadventurous. In preventing the worst, it throws away golden opportunities."((Finlay & Dix 1996), page 109). The algorithm used by most AI-based game-playing programs is the minimax algorithm derived directly from the maximin strategy mentioned above.

The concept of dominance of strategies have been proposed to choose between two moves that have the same security level (Luce & Raiffa 1957). Move α_k dominates move α_l if $\forall \beta_j \in \beta, u(\alpha_k, \beta_j) \geq u(\alpha_l, \beta_j)$. The concept of dominance can be used to choose the most preferred move from the set of maximin strategy moves (moves that all have the same maximum security levels). But this choice is still conservative because we are choosing one of the maximin strategy moves. However, if the player chose another criterion to optimize instead of optimizing security levels, it can possibly improve its performance. We discuss these possibilities next.

Maximum Expected Utility player

Decision theoretic principles can be used by agents to make rational action choices in the face of environmental uncertainity. An agent is rational if and only if it chooses an action that yields the highest expected utility, averaged over all the possible outcomes. This is called the principle of Maximum Expected Utility (MEU) (Russell & Norvig 1995). If the outcome of an action a at a state E is uncertain and can result in one of several possible outcomes $Result_i(a)$, and the function $U(S)$ measures the utility of state S to the agent, then the expected utility of taking action a in state E can be expressed as

$$EU(a|E) = \sum_i Pr(Result_i(a)|E, Do(a)) \times U(Result_i(a)),$$

where $Pr(.)$ represents probabilities, $Do(a)$ represents the proposition that action a is executed in the current state, and i ranges over different outcomes. Let A be the set of actions available to the agent. The MEU principle prescribes that the agent would choose the action that maximizes expected utility:

$$\text{MEU}(A, E) = argmax_{a \in A} EU(a|E).$$

In game-playing programs, part of the uncertainty arises because the opponent strategy is not known. This situation is analogous to the general case of decision making under uncertainty mentioned above. The MEU principle can then be applied to game playing programs. We now propose the following method of choosing moves in a game as an alternate to the maximin strategy:

$$\arg\max_{\alpha_i \in \alpha} \sum_{\beta_j \in \beta} p(\beta_j|\alpha_i)\ u(\alpha_i, \beta_j),$$

where $p(\beta_j|\alpha_i)$ is the conditional probability that the competitor chooses the move β_j given that our agent plays its move α_i.

One interesting observation is that the maximin strategy choice can be seen as a special case of the MEU strategy where $p(\beta_x|\alpha_i)$ is non-zero only if $u(\alpha_i, \beta_x) = \min_{\beta_j \in \beta} u(\alpha_i, \beta_j)$, i.e., the competitor is assumed to choose only from one of its optimal responses to any move of the learner. Our approach to developing MEU players is to learn the conditional probabilities by repeated game playing with the same competitor. In Figure 1 we present two situations to illustrate the difference between MEU and maximin choices. In this figure numbers in ovals represent minimax values corresponding to that move (for leaf nodes, this is the same as the utility of that node), numbers in squares represent the expected values, and the numbers next to the leaf nodes represent the conditional probabilities that the competitor will play that move.

In situation (a), moves A2 and A3 have the same security level, and A2 dominates A3. But the MEU choice is to play the dominated move A3 because of the particular nature of the opponent (as modeled by the conditional probabilities). Let us consider why this situation arise?

Consider Figure 1(a). For the dominant move A2 of the first player, the optimal move for the second player is A21. The corresponding non-optimal moves, A22 and A23, are so bad that it is highly unlikely (as observed by the corresponding probabilities of 0.03 and 0.02 respectively) that the second player will make such mistakes. Hence, the second player will most likely choose its best move and the value obtained by the first player would be 2. For the dominated move A3 of the first player, however, the non-optimal moves of the second player are not that bad. As a result, there is a higher probability (as observed by probabilities of 0.35 and 0.25 respectively) that the second player will make

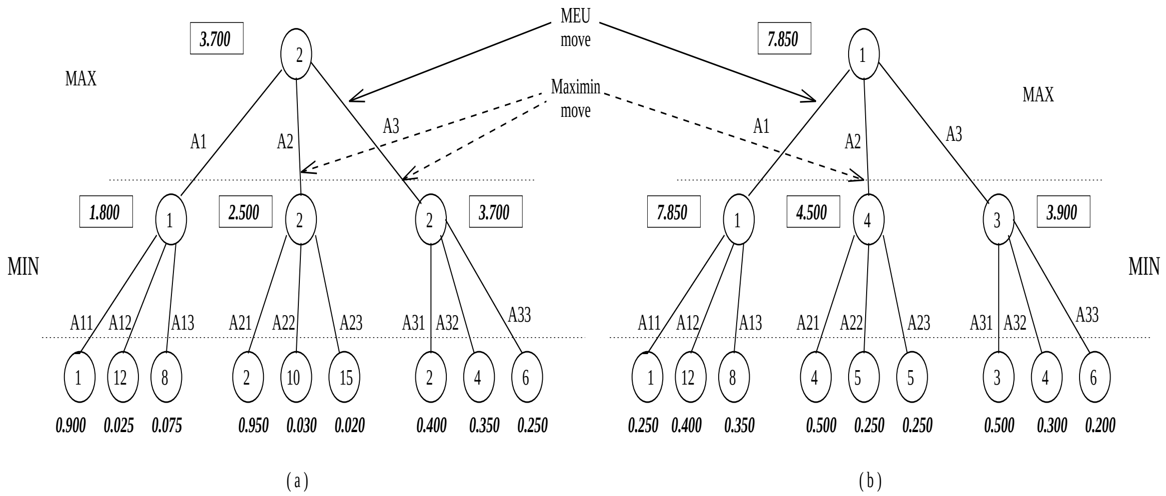

Figure 1: MEU choices in two-person games.

one of its non-optimal moves, A32 or A33. As such, the first player will obtain a greater value if it chose the dominated move A3 over the dominating move A2. In this example, we have seen that the probability of the second player making a mistake depends on the actual move being played by the first player. This fact is overlooked by dominance analysis, and is utilized by the MEU strategy.

In situation (b) of Figure 1, A2 is the minimax move, but the MEU choice is A1. The particular opponent is definitely using a flawed evaluation function because it is frequently choosing moves that it should not consider. The interesting aspect of the MEU strategy is that the worse the heuristic used by the opponent, the more effective will it be in forcing more favorable situations. This is precisely the characteristic we desire in an opportunistic risk-taker. If the conditional probabilities are accurate, the MEU player is very likely to outperform the maximin player against these types of opponents.

Learning mechanism

The goal of our learning mechanism is to develop a probabilistic model of the move selections of an opponent. We start off by assigning prior probabilities such that the MEU move is chosen from the set of maximin moves. Over time, probabilities are updated to correctly reflect the level of play of the opponent. We investigate an approach to learning these probabilities where conditional probabilities are approximated by using move counts that the opponent made in different discrepancy ranges. Consider a move α_1 of the learner corresponding to which the opponent can make moves β_1 and β_2 with $u(\alpha_1, \beta_1) < u(\alpha_1, \beta_2)$. If the competitor fails to choose its best move, β_1, we increase the count of it taking a move with the discrepancy value $u(\alpha_1, \beta_2) - u(\alpha_1, \beta_1)$, and the count of not taking the optimal move (represented as a discrepancy value of 0).

Note that this form of learning makes no restrictive assumption about the nature of the other player. In particular, none of the following simplifying assumptions are used:

Deterministic strategy: Often, it is assumed that the exact move of the opponent can be accurately predicted after observing its behavior for some period of time. These approaches assume that the the other player is using a deterministic strategy to select its actions.

Static strategy: Batch learning procedures are not effective in situations where the opponent behavior can change over time. Since the proposed learning mechanism can be used on-line, it should even be able to adapt to an opponent who changes its strategy. We can improve the tracking characteristic of this learning method by only considering data points in a moving window.

Strategy class approximately known: In some cases, researchers assume that the other player is using the same evaluation function to rate board positions, but some other termination conditions for their game tree search. At other times, researchers assume that the opponent strategy can be accurately modeled by a particular kind of computing machine with limited expressiveness.

Experimental Results

Our initial domain of study involves the two-player zero-sum game of Connect-Four. Connect Four is a popular two-player board game. Each player has several round tokens of a specific color (black or white). The board is placed vertically and is divided into six slots (the actual game sold in the market has seven slots, but most of the AI programs use the six slot version of the game). Each slot has a room for six tokens. Players alternate in taking moves. A player wins if either one of the player is able to line up four of its tokens (form a *quad*) horizontally, vertically, or diagonally. Game ends in a draw if the board fills up with neither player winning.

We designed and executed a series of experiments in the Connect-4 domain to demonstrate the effectiveness of the MEU player over the minimax (MM) player. Both MM and MEU player were made to play with a simple player (SP). The simple player uses a relatively weak heuristic to make its move choice: it tries to block any vertical or horizontal quads, but does not look for diagonal quads. The simple player performs no lookahead search and does not use any model of its opponent. A much more sophisticated board evaluation routine is used by both the MM and the MEU player. For a given board, this routine looks for all different ways it can improve the player's position, giving exponentially higher values if it finds 1, 2, or 3 different tokens in a row, column, or diagonal. We will not discuss the routine in detail. As long as the heuristic used by the MM and MEU player are more powerful than that used by SP, our learning scheme should work.

Since all the players used in our experiments were deterministic players (i.e., given the same input configuration again, the player will repeat its last move from that position), we had to collect statistics over several initial board configurations to substantiate our claim about the effectiveness of MEU vs. MM player.

Training Phase

A necessary pre-requisite for an agent to develop a good model of an opponent is for it to observe the opponent making moves from a representative sample of the space of all possible board configurations. To roughly approximate this breadth of experience, we had the players play games from randomly generated board configurations with $n \in \{2, 4, 6, 8, 10, 12, 14, 16\}$ moves already played in the game. A total of 200 different starting configurations were used during training. thus providing a wide variety of starting and mid-game configurations in Connect-4. The two players MEU and SP played against each other a set of 400 games. During the first 200 games, the MEU made the first move in the game and for the rest of the games, SP started first.

During the training phase, the MEU player incrementally updated its probabilistic model of the opponent. Figure 2 shows the information gathered by MEU about SP over a set of 400 games. We observe that the probability that the opponent will choose a

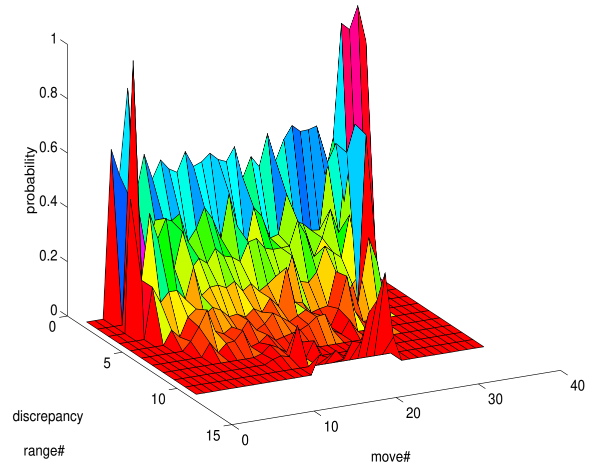

Figure 2: Probabilistic model of opponent moves generate by the learning player.

move with 0 discrepancy (its best move) is less than 1 for all move numbers. We now inspect some of the details of the learned model. It seems that this particular simple player plays many more optimal moves during the start of the game (for small move numbers), or when the board is almost full (for large move numbers).

One type of mistake made by the simple player stands out: around move number 20, it is often making the worse mistakes (choosing the moves with highest discrepancy range) though it avoids less obvious mistakes! When we looked at this kind of mistakes by the simple player, we were pleasantly surprised. Recall that the SP strategy blocks only diagonal or vertical quads, but do not look for diagonal quads. As a result, at times it plays a move which actually contributes to the immediate win by the learner by forming a diagonal quad. Such mistakes were observed more frequently only after a number of moves have been made on the board. Though the learner had no idea about the SP strategy, the probability model does capture, albeit indirectly, a major flaw in the SP strategy. This 'discovery', however, is not a claim on our part that every possible flaw in the opponent strategy can be accurately uncovered by our learning scheme.

Testing Phase

After MEU acquired enough knowledge about the opponent, it was tested on a number of randomly generated boards. In the testing phase, 30 different boards were used, each of which had two tokens, one token for the MEU or MM player and the other for the SP. 16 of the 30 boards produced identical results with the MEU or MM player. In 10 games the MEU player won in less number of moves, and in 4 games it took more moves to win than the minimax player. Since the model developed by the MEU player is a probabilistic one, it is to be expected that at times the risks taken by the MEU player will be counter-productive. In these situations, the MEU player took more moves to win than the MM player. In a significantly larger number of cases, though, the MEU player was successful in exploiting the weaknesses of the SP, and win in less moves than taken by the conservative MM player.

Examples of risks taken At this point, we recall that there are two classes of situations in which the probabilistic opponent model can be used by MEU player to choose a move different than that chosen by a MM player:

Prudent move: The situation in Figure 1 (a), where the MEU player is choosing a move of the same security level as the minimax move. We call this a prudent move because it does not risk losing the performance guarantee of the minimax move, but stands to gain more if the opponent makes a mistake consistent with the probability model being used by the MEU player.

Risk: The situation in Figure 1 (b), where the MEU player is choosing a move with a lower security level than the minimax move. This is a 'true' risk in the sense that the MEU player is giving up the performance guarantee of the minimax move because of the expectation that the opponent will make a mistake predicted by the learned probabilistic model.

In figure 3 we present an actual game situation where a prudent move taken by the MEU player leads to a win in less number of moves than that taken by the MM player from the same board. In this figure, the MEU player has to pick one of six moves with corresponding board configurations shown at the second level. The numbers next to the boards are the minimax values and the values in the boxes are the expected values returned by the MEU evaluation routine. It is shown that the MEU strategy picks a move different from the MM strategy, and after three more moves, wins the game with a diagonal quad. For simplicity of presentation, we have showed only the actual moves chosen by the two players for the last few moves.

A corresponding situation with an actual risk taken is presented in Figure 4.

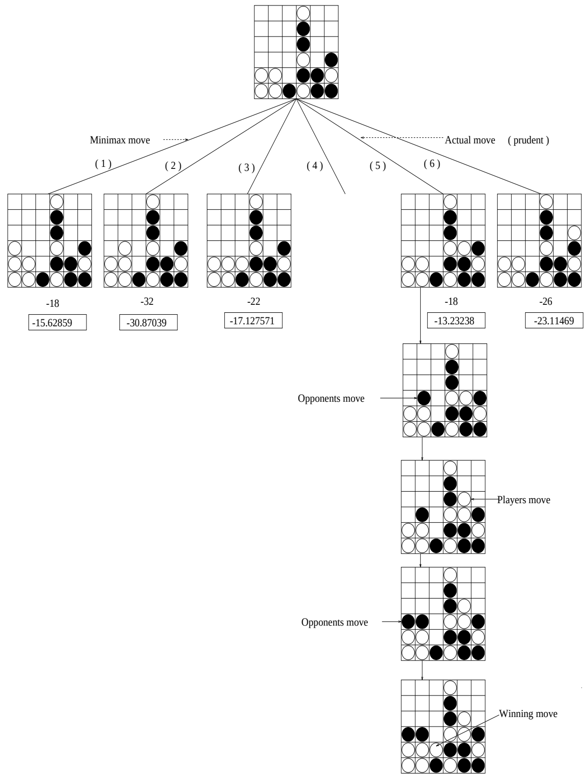

Figure 3: Winning in less moves by taking prudent move ("Actual move" is move taken by MEU player whereas "Minimax move" is the move taken by MM player).

Related Work

Our own larger research agenda encompasses different learning scenarios in multiagent systems. Though we are interested in both cooperative and adversarial learning situations, the current work should be compared and contrasted to multiagent learning research in non-cooperative domains. Our past attempts at using traditional reinforcement learning schemes in adversarial domains gave us important insights into the effectiveness and limitations of this approach (Sekaran & Sen 1994). Our experiments reinforced the notion that Q-learning and other such algorithms are not capable of handling non-stationary environments (Kaelbling, Littman, & Moore 1996). Other researchers exploring the possibility of using Q-learning to explicitly model competitors have also stumbled against the limitations of such approaches when competitors are learning concurrently (Sandholm & R.H.Crites 1995). An alternate approach to using reinforcement learning agents in a game situation utilizes the Markov games formulation (Littman 1994). Both approaches suffer from large training time and data requirements, making them unusable in most realistic problems. The Minimax-Q algorithm developed by in the latter work (Littman 1994) is interesting in that it tries to train a player to be as conservative as a minimax approach suggests. In

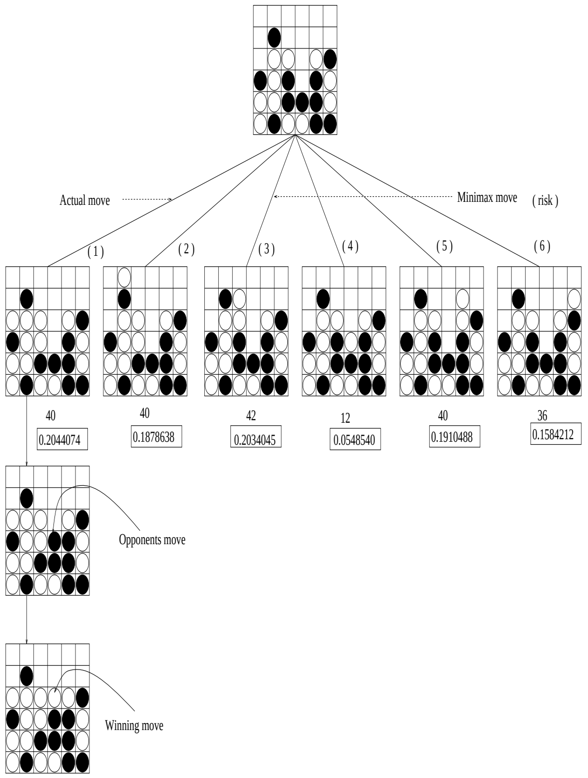

Figure 4: Winning in less moves by taking risk ("Actual move" is move taken by MEU player whereas "Minimax move" is the move taken by MM player).

sharp contrast, our goal is to train a player to exploit a particular opponent.

Conclusions

In this paper, we have proposed a Maximum Expected Utility (MEU) strategy for playing adversarial board games. This approach requires a probabilistic model of opponent player's move choices. If an accurate model is available, the MEU strategy always chooses the rational move, i.e., the move that is expected to maximize the utility of the player. The goal of the MEU approach is to exploit the weaknesses in the strategy used by the opponent. This approach is contrasted to the widely used minimax algorithm, which is extremely conservative in nature, and thus never takes any risks. The expected advantage of the MEU strategy over the minimax strategy in game-playing situations would be to produce wins in less number of moves or avoid losses.

The use of the MEU strategy demands the existence of an accurate probabilistic model of opponent behavior. We propose a learning scheme to develop such a model by repeatedly playing against a given opponent. To demonstrate the effectiveness of our learning approach, we have used the common board game of Connect-4. Results from a series of experiments are encouraging. The MEU strategy is shown to outperform the minimax by forcing wins in less moves.

We would now like to make some clarifications regarding our domain choice and the applicability of our proposed learning schemes: though Connect-Four is a perfect information game, our proposed learning scheme is perfectly suitable for games that include chance elements (e.g., games where the progression of the game depends on dice rolls); our approach can be extended to cover n-player games in general and is also applicable to non-zero-sum games.

We would also like to evaluate our learning scheme in another domain like othello. In a further set of experiments in the Connect-4 domain we want to compare the different probabilistic models learned by our proposed scheme when playing a number of different opponents.

References

Eichberger, J. 1993. *Game Theory for Economists.* Academic Press Inc.

Finlay, J., and Dix, A. 1996. *An Introduction to Artificial Intelligence.* London, UK: UCL Press.

Kaelbling, L.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement learning: A survey. *Journal of AI Research* 4:237–285.

Littman, M. L. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, 157–163.

Luce, R. D., and Raiffa, H. 1957. *Games and Decisions.* John Wiley & Sons.

Russell, S., and Norvig, P. 1995. *Artificial Intelligence: A Modern Approach.* Printice Hall.

Samuel, A. L. 1959. Some studies in machine learning using the game of checkers. *IBM Journal* 3(3).

Sandholm, T., and R.H.Crites. 1995. Multiagent reinforcement learning in the iterated prisoner's dilemma. *Biosystems* 37:147–166.

Sekaran, M., and Sen, S. 1994. Learning with friends and foes. In *Sixteenth Annual Conference of the Cognitive Science Society*, 800–805.

Tesauro, G., and Sejnowski, T. 1989. A parallel network that learns to play backgammon. *Artificial Intelligence* 39(3):357–390.

Tesauro, G. 1992. Practical issues in temporal difference learning. *Machince Learning* 8(3–4):257–277.