How to Explore Your Opponent's Strategy *(almost)* Optimally

David Carmel and Shaul Markovitch
Computer Science Department,
Technion, Haifa 32000, Israel
{carmel,shaulm}@cs.technion.ac.il

Abstract

This work presents a lookahead-based exploration strategy *for a* model-based learning agent *that enables exploration of the opponent's behavior during interaction in a* multi-agent system. *Instead of holding one model, the model-based agent maintains a* mixed opponent model, *a distribution over a set of models that reflects its uncertainty about the opponent's strategy. Every action is evaluated according to its long run contribution to the expected utility and to the knowledge regarding the opponent's strategy. We present an efficient algorithm that returns an* almost *optimal exploration strategy against a given mixed model, and a learning method for acquiring a mixed model consistent with the opponent's past behavior. We report experimental results in the* Iterated Prisoner's Dilemma game *that demonstrate the superiority of the lookahead-based exploration strategy over other exploration methods.*

1. Introduction

Consider an open market where buyers and sellers meet and interact for trade. The *agents* involved in the market are completely autonomous and consider only themselves. They may also have different preferences, different negotiation skills and different levels of knowledge. In such a *multi-agent system* (MAS), an agent should be designed to efficiently interact with other agents who may be potential partners, or alternatively, competing opponents.

Multi-agent systems are typically complex with respect to their structure and functionality. Designing an effective interaction strategy requires a prior knowledge about any agent that might be involved in the system. However, for agents with private knowledge and goals, such a requirement is not reasonable. One way to deal with this difficulty is to endow the agent with the ability to adapt to the other agents.

Reinforcement learning (RL) is a popular technique used by adaptive agents in MAS for learning interaction strategies with other agents [13]. RL is based on the idea that the tendency to produce an action should be reinforced if it produces favorable results, and weakened if it produces unfavorable results. RL spends little computation resources per example but requires large number of examples. A *model-based* approach for learning interaction strategies [5] reduces the number of examples needed for adaptation, by investing more computational resources in deeper analysis of interaction experience. The learning agent makes use of an explicit model of the other agent's strategy to generate expectations about its behavior. At any stage of the game, the agent updates the opponent model to be consistent with current sample of the opponent's behavior. It then follows the optimal response against the current model.

The model-based approach gives priority to actions with the highest expected utility, according to the current accumulated knowledge. But such policy does not consider the effect of the agent's behavior on the learning process, and ignores the contribution of agent's activity to the exploration of the opponent's strategy. In essence, every action affects the interaction process in two ways:

1. The effect on the expected reward according to the current knowledge held by the agent.
2. The effect on the acquired knowledge, and hence, on future rewards expected to be received due to better planning.

The agent therefore must make a tradeoff between the wish to exploit its current knowledge and the wish to explore other alternatives, to improve its knowledge for better decisions in the future. This tradeoff is known in decision-theory as the *exploration versus exploitation dilemma*. Agents that engage in exploration to the exclusion of exploitation pay the cost of experimentation, without gaining benefits of the accumulated knowledge. Conversely, agents that engage in exploitation to the exclusion of exploration, are likely to find themselves trapped in sub-optimal behavior.

The exploration vs. exploitation problem has received significant attention within the RL research. Several methods were developed for incorporating exploration strategies into RL agents [14]. *Undirected* methods are based on incorporating randomness into the decision procedure by assigning each action a positive probability of being selected and performed. *Directed* methods use the interaction history to evaluate the expected contribution of every action to exploration. In previous work [3] we have shown how to incorporate such exploration methods into model-based learning.

One problem with the existing exploration methods is that they do not take into consideration the risk involved in exploration. An exploratory action taken by the agent tests unfamiliar aspects of the other agent's behavior which can yield a better model of the other agent. However, such an action carries also the risk of putting the agent into a much worse position. In this work we present a *lookahead-based exploration strategy* that evaluates actions according to their expected utility, their expected contribution to the acquired knowledge, and the risk they carry. Instead of holding one model, the agent maintains a *mixed opponent model* – a distribution over a set of models that reflects its uncertainty about the opponent's strategy. Every action is evaluated according to its long run contribution to the expected utility, and to the knowledge regarding the opponent's strategy, expressed by the posterior probabilities over the set of models. Risky actions are detected by considering their expected outcome according to the alternative models of the opponent's behavior.

The rest of the paper is organized as follows: Section 2 briefly describes our earlier work [5] on *model-based learning* in repeated games. Section 3 describes the *lookahead-based exploration strategy* that deals with exploration using utility-based criteria usually used in decision-theory. In Section 4 we show experimentally the necessity of exploration for model-based learning and the superiority of lookahead-based exploration over uninformed exploration methods. Section 5 discusses some related work and concludes the work.

2. The Basic Framework

To formalize the notion of interacting agents we consider a framework where an encounter between two agents is represented as a *two-player game* and a sequence of encounters as a *repeated game*. At any stage t of the game, the players choose their moves simultaneously. A history of the repeated game is a finite sequence of joint moves chosen by the agents up to the current stage of the game. A *strategy* for an agent is a function that takes a history and returns an action.

A pair of strategies (s_1, s_2) defines an infinite sequence of joint moves (a path), while playing the game. The *discounted-sum* function is a common utility function for repeated games:

$$U_i^{ds}(s_1, s_2) = \sum_{t=0}^{\infty} \gamma_i^t u_i(r_1^t, r_2^t) \quad (1)$$

$0 \leq \gamma_i < 1$ is a discount factor of player i for future payoffs, and $u_i(r_1^t, r_2^t)$ is the expected utility of player i for the joint move performed by the players at stage t of the expected game-path. A strategy $s^{opt}(s_j, U_i)$ will be called *best response* for player i with respect to strategy s_j and utility U_i, iff for any strategy s for player $i, U_i(s^{opt}(s_j, U_i), s_j) \geq U_i(s, s_j)$.

For example, the *Prisoner's dilemma* (PD) is a two-player game, where each player has two actions, cooperate (c) and defect(d). The utility functions of the players are described by the payoff matrix shown in Figure 1. The *Iterated Prisoner's Dilemma (IPD)* [1] is an example of a repeated game based on the Prisoner's dilemma game. Tit-for-tat (TFT) is a simple well known strategy for IPD that has been shown to be very successful in IPD tournaments. It begins with cooperation and imitates the opponent's last action afterwards. The *best-response* against TFT with respect to U^{ds} depends on the discount parameter γ [1]:

	II c	II d
I c	3 3	5 0
I d	0 5	1 1

Figure 1. The payoff matrix for the Prisoner's Dilemma game.

$$s^{opt}(\text{TFT}, U^{ds}) = \begin{cases} \text{all-c} & \frac{2}{3} \leq \gamma \\ \text{all-d} & \gamma \leq \frac{1}{4} \\ \text{“Alternate between c and d”} & \text{otherwise} \end{cases}$$

One of the basic factors affecting the behavior of agents in MAS is the knowledge that they possess about each other. We assume that each player is aware of the other player's actions, while the players' preferences are *private*. In such a framework, while the history of the game is *common knowledge*, each player predicts the future course of the game differently. The prediction of player i is based on the player's strategy s_i and on the player's belief about the opponent's strategy, $\hat{s}_j$. $\hat{s}_j$ will be called an *opponent model*.

How can a player acquire a model of its opponent's strategy? One source of information available for the player is the history of the game. Another possible source of information is observed games between the opponent and other

agents. Given a learning algorithm L_i that infers an opponent model based on a sample of its behavior, and a utility function U_i, we can define a model-based player (MB-agent) that adapts its strategy during the game. At any stage t of the game, the MB-agent infers an updated opponent model by applying its learning algorithm to the current sample of the opponent's behavior. It then finds the best response against the current model, with respect to U_i, and plays according to the best response.

Generally, computing the *best response* against a given model is too complicated for *bounded rational agents* (agents with limited computational resources). One way of making the computation feasible is by restricting the complexity of the opponent strategies. We follow Rubinstein [12] and restrict the class of strategies available for the opponent to *regular strategies*, i.e. strategies that can be represented by deterministic finite automata (DFA). For example, Figure 2 shows a DFA that implements the strategy TFT for the IPD game.

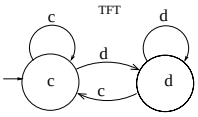

Figure 2. A DFA that implements the strategy TFT for the IPD game.

In previous work [5], we have presented a model-based strategy against regular opponents. The strategy includes an efficient best-response procedure against any regular opponent model, and a learning algorithm that constructs a regular model consistent with a given history. We have also shown how to incorporate exploration methods, originally developed for RL, into model-based learning. The model-based strategy, combined with exploration, outperformed non-adaptive strategies and reinforcement-learning strategies in the Iterated Prisoner's Dilemma game.

3. Lookahead-based Exploration

The exploration methods described in [3] were developed to handle the problem of being stuck in local-minima. An exploring agent is willing to perform sub-optimal actions in order to acquire a better model of the opponent yielding a better utility in the long run. It is quite possible, however, that the exploratory action will indeed yield a better model, but will also lead the agent to a poor state where even optimal play yields low utility. Sometimes, knowledge is too expensive – taking a step into the dark can lead you to the sought treasure, or into a deep chasm.

For example, the left part of Figure 3 describes a strategy for IPD called *Grim*. A Grim player cooperates as long as its opponent cooperates, but never forgives defection – after one single defection of its opponent it reacts by repeatedly defecting. Assume that an exploring agent holds the model described in the right part of the figure. Defection has lower utility than cooperation according to the model, therefore, a non-exploring agent will always cooperate yielding utility of $\frac{3}{1-\gamma}$. An exploring agent will eventually try d and acquire a perfect model of Grim. However, the exploratory action d takes the agent into the "defection sink" without any opportunity to get out, yielding utility of $\frac{5-4\gamma}{1-\gamma}$ which is lower than the utility of cooperation for any $\gamma \geq \frac{1}{2}$. Thus, exploration indeed yielded better knowledge, but the cost of acquiring this knowledge is too high.

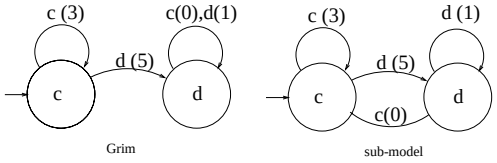

Figure 3. Left: The *Grim* strategy for IPD. Right: the current model held by the agent. An exploratory action against grim (d), will be followed by falling into the "defection sink" without any opportunity to get out.

The above example clearly demonstrates that exploratory behavior requires a better mechanism for predicting the risk involved in taking sub-optimal actions. One of the problems with the model-based strategies is the use of a utility function that assumes a stationary opponent model throughout the expected course of the game. This assumption is not rational for a learning agent who continuously modifies its opponent model. In order to develop a risk-sensitive exploration strategy we need a mechanism that allows the agent to take into consideration the expected revision of its belief while computing the expected utility. Such a mechanism requires a method for representing the agent's uncertainty regarding its opponent's strategy. In the next subsection we describe *mixed strategies* for representing uncertain opponent models.

3.1. Mixed strategies

The model-based agent described in Section 2 maintains a regular model of the opponent's behavior. The regular model is incapable of representing the uncertainty of the learning agent regarding the model's prediction. To represent such uncertainty we apply a stronger mechanism for modeling the opponent strategy: The game-theoretic concept of *mixed strategy*.

Definition 1 *A* mixed strategy *for repeated game $G^{\#}$ is a pair, $(\overline{S}, \overline{\pi}) : H(G^{\#}) \rightarrow \Delta(R_j)$, that maps the set of the game histories, $H(G^{\#})$, into the set of distributions over the player's actions, $\Delta(R_j)$. $\overline{S} = \{s_1, \ldots, s_k\}$ is a finite set of (pure) strategies for $G^{\#}$ called the* set of support *(SOS). $\overline{\pi} = (\pi_1 \ldots \pi_k)$ is a probability distribution over SOS called the* belief distribution. *The probability associated with action $r_j \in R_j$ is $Pr(r_j) = \sum_{l=1}^{k} [\pi_l | s_l(h(t)) = r_j]$. A* regular mixed strategy, *$(\langle \overline{M}, \overline{q} \rangle, \overline{\pi})$, is a mixed strategy whose set of support includes k automata, $\overline{M} = (M_1, \ldots, M_k)$, in states $\overline{q} = (q_1, \ldots, q_k)$.*

Mixed strategies can be used as opponent models in two different ways. In the first one, we assume that the opponent applies a mixed strategy, i.e., it randomly chooses one of the strategies from the set of support at any stage of the game. Gilboa [9] shows how to play against n regular opponents by solving the best response problem against the product automaton of the opponents' DFA. A similar method can be used to find the best response against a regular mixed strategy. First, construct the product automaton of the automata belonging to the set of support, with an output function which is probabilistic according to the given distribution. Second, find the best response automaton against the probabilistic product automaton. The problem of finding the best response against a *probabilistic action automaton* (PAA) is equivalent to the best response problem against a deterministic automaton [8].

The above interpretation still assumes that the agent does not modify its opponent model throughout the game-path while computing the utility, and is therefore not suitable for the belief revision framework. The second interpretation considers the opponent's strategy to be one of the models belonging to the set of support, and the belief distribution to reflect the subjective beliefs of the player in the alternative opponent models. This interpretation is suitable for the learning framework studied in this work. By choosing a specific action, the agent forces the opponent to respond. The opponent's response enables a revision of the belief distribution over the set of the opponent models. Following the opponent's response, r_j^t, the beliefs on all models that do not expect to perform r_j^t should be reduced to zero. The beliefs on all models that expect to output r_j^t should be increased.

Let $(\overline{S}, \overline{\pi})$ be the current mixed strategy. The posterior belief, $\overline{\pi}^{r_j}$, can be computed directly by Bayes law for every $1 \leq l \leq k$:

$$\pi_l^{r_j} = \begin{cases} 0 & s_l(h(t)) \neq r_j \\ \dfrac{\pi_l}{Pr(r_j)} & \text{otherwise} \end{cases} \qquad (2)$$

We can now define the utility of the expected game-path between a learning strategy and a mixed model, while considering the modified beliefs throughout the path:

$$\hat{U}_i^{ds}(s_i, (\overline{S}, \overline{\pi})) = \sum_{r_j \in R_j} Pr(r_j)[u_i(s_i(h(t)), r_j) + \gamma_i \hat{U}_i^{ds}(s_i, (\overline{S}, \overline{\pi}^{r_j}))]$$

Ben-Porath [2] shows that under this interpretation, the best response problem against a regular mixed strategy is NP-complete. He also shows that the problem becomes polynomial in the size of the product automaton when fixing the size of the set of support. However, the best response procedure described by Ben-Porath is impractical since the product automaton is extremely large even for a small set of support. In the next subsection we describe an alternative approach that is more suitable for a computational bounded agent. We present an efficient algorithm that returns an almost best response automaton against a regular mixed model. The level of approximation can be determined in advance, according to the computational resources available for the agent, at any stage of the game.

3.2. An almost-best response strategy against a regular mixed model

In this subsection we develop an algorithm that returns an almost-best response automaton against a regular mixed model. Given an opponent's strategy s_j, and an approximation parameter, ϵ, an ϵ-best response strategy, with respect to s_j, guarantees the player a utility which is less than the maximal possible utility against s_j, by at most ϵ.

Definition 2 *A strategy $s_\epsilon^{opt}(s_j, U_i)$ will be called* ϵ-best response *for player i with respect to strategy s_j and utility function U_i, iff for any strategy s, $U_i(s_\epsilon^{opt}(s_j, U_i), s_j) \geq U_i(s, s_j) - \epsilon$.*

Let u_{max} be the maximal value of the stage game G, and $U_{max} = \frac{u_{max}}{1-\gamma}$ be the maximal utility in the repeated game $G^{\#}$. The algorithm receives a regular mixed model, $(\langle \overline{M}, \overline{q} \rangle, \overline{\pi})$, and a depth parameter d that is determined by the approximation parameter ϵ, $d \geq \frac{\ln(\frac{\epsilon}{U_{max}})}{\ln(\gamma)}$. The algorithm returns a pair of an ϵ-best response automaton and the expected utility of the game against the given mixed model.

States of the mixed strategy are classified to *certain states* where all the models predict the same output, and *uncertain states* where at least two models disagree on the opponent's current action. For *uncertain states*, the algorithm splits the models of SOS into disjoint sets according to their expected output, $\overline{M}^{r_j} = \{M_l \in \overline{M} | M_l(h(t)) = r_j\}$. Any opponent's action reduces the number of models in the corresponding set of support since the beliefs in all models that are not consistent with this action is reduced to zero. For each

pair (r_i, r_j) of a player's action and an opponent's action, we construct a posterior mixed strategy, $(\langle \overline{M}^{r_j}, \overline{q}^{r_i} \rangle, \overline{\pi}^{r_j})$. The posterior belief, $\overline{\pi}^{r_j}$, over $\overline{M}^{r_j}$, can be computed using Equation 2. The player's action, r_i, determines the new state of the posterior mixed model. The algorithm calls itself recursively with a reduced depth limit. The recursion stops when the search reaches the depth limit, or, when the set of support includes only one model, for which the problem is reduced to the best response problem against a single automaton.

Certain states do not reveal information to the player, since all the models predict the same output. Therefore, the set of support and the belief distribution cannot be modified. For such states, the models' current states are modified according to the player's action and the best response is found recursively with a reduced depth limit.

Denote the automaton returned by the recursive call for the pair (r_i, r_j) by A^{r_i,r_j}, and the expected payoff of the game to be played by U^{r_i,r_j}. Let r_i^* be the action that maximizes the expected utility of performing action r_i at the given state of the mixed model:

$$r_i^* = arg\max_{r_i} \sum_{r_j \in R_j} Pr(r_j)[u_i(r_i, r_j) + \gamma U^{r_i,r_j}]$$

The ϵ-best response automaton begins with r_i^* at the initial state and then plays $A^{r_i^*,r_j}$, according to the opponent's action r_j.

Note that the automaton described above provides a plan of d steps for the player for exploring and exploiting the mixed model. The plan optimizes the player's behavior by considering the expected utility, and also the expected revealed information throughout the alternative game-paths. This plan is in contrast to the infinite plan returned by the best response procedure against a *single automaton*, but which does not direct the agent's behavior when the opponent does not play as predicted. Figure 4 describes a pseudo-code of the algorithm.

Figure 5 demonstrates how the ϵ-best response strategy can avoid falling into the defection sink when playing the IPD against Grim. The top part of Figure 5 shows two models held by the exploring agent, after t mutual cooperations in the IPD game. M_0 is TFT. M_1 is the Grim strategy. The bottom part of the figure shows the computation of the best response. The player's actions are marked by solid lines and the opponent's actions are marked by dashed lines. To save space, duplicated subtrees are drawn only once. Action c has no exploration benefit since all the models predict cooperation of the opponent after c. The utility of cooperation is $U(c) = 3 + 5\gamma + \frac{2\gamma^3}{1-\gamma}$. The utility of defection is $U(d) = 5 + \frac{2\gamma^2}{1-\gamma}$. Hence, c is prefered over d for any $\gamma > 0.5$. Note that the exploration strategy returned by the algorithm is "c then all-d". When the algorithm searches deeper, the returned strategy will postpone defection for later stages of the game-path.

Procedure ϵBR$\big((\langle \overline{M}, \overline{q} \rangle, \overline{\pi}), d\big)$
  $k \leftarrow |\overline{M}|$
  if $(d = 0)$ or $(k = 0)$ **then return** $\langle A^0, 0 \rangle$
  if $(k = 1)$ **then return** PureBR$(\langle M, q \rangle)$
  else
    if uncertain($\overline{q}$)
      for each $r_j \in R_j$
        $Pr(r_j) \leftarrow \sum_{l=1}^{k} [\pi_l | M_l(h(t)) = r_j]$
        $\overline{M}_{r_j} \leftarrow \{M_l \in \overline{M} | M_l(h(t)) = r_j\}$
        for $l = 1 \ldots, k$
          if $M_l(h(t)) = r_j$ **then** $\pi_l^{r_j} \leftarrow \frac{\pi_l}{Pr(r_j)}$
          else $\pi_l^{r_j} \leftarrow 0$
        for each $r_i \in R_i$
          $q_l^{r_i} \leftarrow$ (the new model's state following r_i), $1 \le l \le k$
          $\langle A^{r_i,r_j}, U^{r_i,r_j} \rangle \leftarrow \epsilon\text{BR}\big((\langle \overline{M}^{r_j}, \overline{q}^{r_i} \rangle, \overline{\pi}^{r_j}), d-1\big)$
        $U \leftarrow \max_{r_i} \sum_{r_j} Pr(r_j)[u_i(r_i, r_j) + \gamma U^{r_i,r_j}]$
        $r_i^* \leftarrow arg\max_{r_i} \sum_{r_j} Pr(r_j)[u_i(r_i, r_j) + \gamma U^{r_i,r_j}]$
        $A \leftarrow$ (a DFA that begins with r_i^* and plays $A^{r_i^*,r_j}$)
    else /* $\overline{q}$ is certain, all models predict the same action r_j^* */
      for each $r_i \in R_i$
        $q_l^{r_i} \leftarrow$(the new model's state following r_i), $1 \le l \le k$
        $\langle A^{r_i,r_j^*}, U^{r_i,r_j^*} \rangle \leftarrow \epsilon\text{BR}\big((\langle \overline{M}, \overline{q}^{r_i} \rangle, \overline{\pi}), d-1\big)$
      $U \leftarrow \max_{r_i} [u_i(r_i, r_j^*) + \gamma U^{r_i,r_j^*}]$
      $r_i^* \leftarrow arg\max_{r_i} [u_i(r_i, r_j^*) + \gamma U^{r_i,r_j^*}]$
      $A \leftarrow$ (DFA that begins with r_i^* and plays $A^{r_i^*,r_j^*}$ for any r_j)
    return (A, U)

Figure 4. The ϵBR algorithm that returns an ϵ-best response automaton against a regular mixed model.

The following theorem proves that algorithm ϵ-BR() returns an ϵ-best response strategy against the given mixed strategy, with respect to $\hat{U}^{ds}$.

Theorem 1 *Let ϵ be a positive number. Let $\epsilon' = \frac{\epsilon}{U_{max}}$, and let $d \ge \frac{ln(\epsilon')}{ln(\gamma)}$. Let $(\langle \overline{M}, \overline{q} \rangle, \overline{\pi})$ be a regular mixed strategy. The algorithm ϵBR$((\langle \overline{M}, \overline{q} \rangle, \overline{\pi}), d)$, returns an ϵ-best response automaton, against $(\langle \overline{M}, \overline{q} \rangle, \overline{\pi})$, with respect to $\hat{U}_i^{ds}$. The computation time is polynomial in $\frac{1}{\epsilon'}$ and the size of the maximal automaton belonging to the set of support.*

Proof. The proof can be found in [4]. □

3.3. Learning a mixed model

Describing the opponent model by a mixed strategy requires a learning procedure for acquiring alternative models

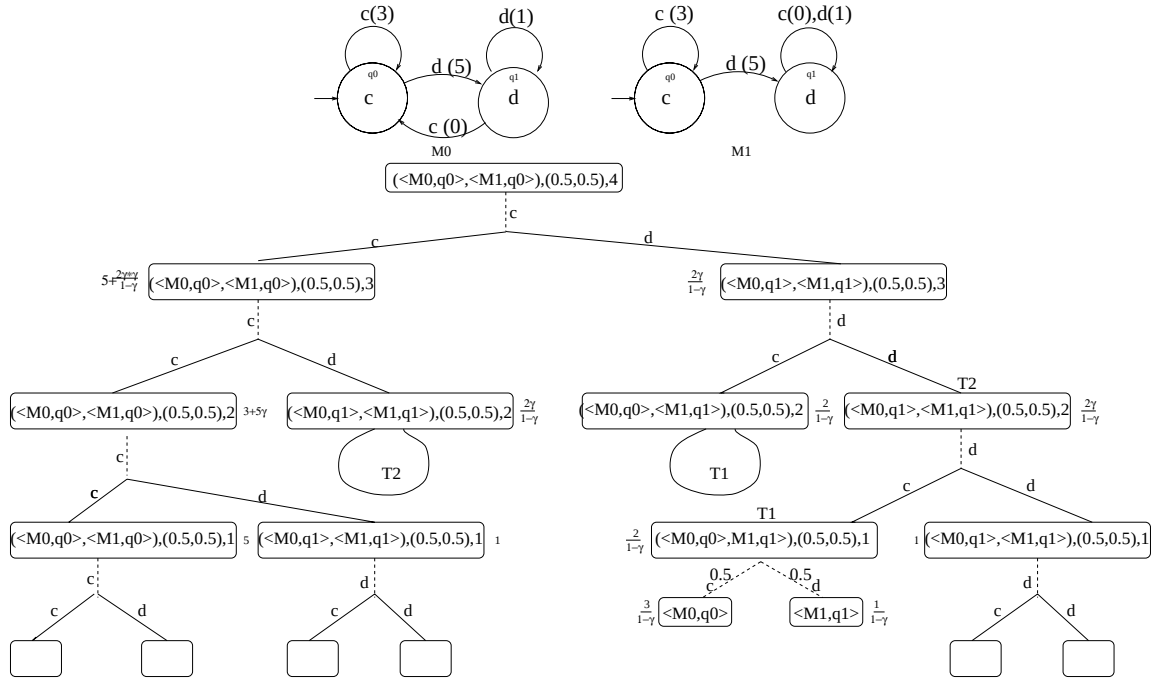

Figure 5. Top: The 2 models belong to the mixed model after t mutual cooperations in the IPD game. Bottom: The search tree spanned by ϵ-BR. The player's actions are marked by solid lines. The opponent's actions are marked by dashed lines. To save space, duplicated subtrees are drawn only once. Action c is preferred over d for any $\gamma > 0.5$.

for SOS, and a computational procedure for determining the belief distribution over SOS. Ideally, the set of support should include all the models consistent with the given history, but such a set of strategies is infinite.

For restricting the set of support we concentrate on a subset of models consistent with the given history that differ in predicting the expected opponent's action at the next stage of the game. We use the learning algorithm of the model-based strategy, L_i, for constructing these models. By concatenating all the possible pairs of actions, (r_i, r_j), to the given history, and by applying the learning algorithm L_i, to the new expanded histories, we acquire models that are consistent with the given history and predict the opponent's next action to be r_j at stage $t+1$ of the game. The player action at stage $t+1$, r_i, does not affect the learning algorithm since the opponent's action at stage $t+1$ is only affected by the previous player's actions. Hence, we can acquire at most $|R_j|$ different models for the set of support.

Definition 3 *Given a history $h(t)$, and a learning algorithm L_i.* The set of support, $SOS^1(L_i, h(t))$, *is defined to be the set of models returned by the algorithm L_i, applied to the history $h(t)$ concatenated with one more joint action (r_i, r_j):*

$$SOS^1(L_i, h(t)) = \{M | M = L_i(h(t) \circ (r_i, r_j)), \; r_i \in R_i, r_j \in R_j\}$$

After acquiring the different models for the set of support, the agent should determine its belief distribution over the set. Since all models are consistent with the given history, each one can serve as an opponent model. However, there are models that seem to be more reasonable than others. The agent can choose any way for computing its subjective beliefs. The only restriction is that its beliefs should be non-negative and should be summed to one. According to the Occam razor principle, a reasonable belief probability for a given model should be in inverse relation to its size, i.e. the smaller the model, the larger is its associated belief. The beliefs should also be in direct relation to the coverage of the models: the smaller the number of edges of the model that have been tried by the agent during the given history, the smaller the agent's belief in the given model.

$$\pi_j \propto \frac{cover(h(t), M_j)}{|M_j|}$$

The strategies acquired for the set of support were inferred by predicting different responses of the opponent to the given history. For testing the influence of the agent's behavior on the opponent's behavior, we should search some stages deeper, testing the effects of the agent's actions on the opponent's expected responses. When we search d stages forward, we shall look at all possible expanded histories of length $t+d$. By concatenating all the possible d pairs of

actions to the given history, and by applying the learning algorithm to the expanded histories, we acquire models that are consistent with the history and predict differently the opponent responses for the player sequences of actions.

To summarize, for exploring the opponent's strategy using a mixed model, the agent first searches d stages forward for collecting different opponent models to the set of support. It then infers a belief distribution over this set. Following that, it finds the ϵ-best response against the mixed model and performs a sequence of actions dictated by this strategy. By doing so, the agent rationally balances between exploration and exploitation, and reduces the risk involved in performing sub-optimal actions. This may carry additional computational cost, but the investment may be profitable when the risk involved in exploration is high.

4. Experimentation: The effect of exploration on the agent's performance

In this section we compare experimentally some exploration methods for model-based agents: undirected Boltzmann exploration, directed recency-based exploration, combined exploration [3], and the lookahead-based exploration. The stage-game used in these experiments is the PD-game. 100 random opponent automata were generated by choosing a random transition function and a random output function. The MB-agent begins with a random DFA as a model of its opponent's strategy and modifies the model whenever it fails to predict the opponent's actual play. It follows the *best response* against the inferred model incorporated with different exploration methods. We use the average cumulative reward achieved by the player to measure the quality of its learning strategy: $\frac{\sum_{k=0}^{t-1} u_i(r_i^k, r_j^k)}{t}$. Figure 6 shows the average cumulative reward of different exploration strategies attained during 400 stages of the IPD game. The results are averaged over 100 trials against 100 different random automata of size 60.

The exploration behavior of the undirected strategy is sensitive only to the length of the history and not to its content. Therefore it does not modify its exploration behavior when playing against different opponents. The directed strategy, on the other hand, utilizes the content of the history to modify its exploration behavior. This is the reason why it outperforms the undirected method. The directed method, however, does not reduce its exploration tendency with time. The combined strategy enjoys the advantages of both methods and outperforms them both. The lookahead-based exploration strategy outperforms significantly the other exploration methods. This advantage carries the cost of higher computational costs. Due to this higher costs we performed IPD games of length 200 instead of 400.

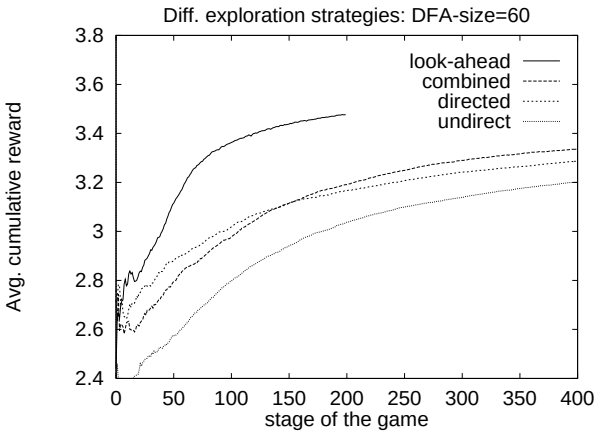

Figure 6. The average cumulative reward of different exploration strategies while playing the IPD against 100 random automata of size 60.

5. Conclusions

There are two issues that should be considered when dealing with incorporating exploration into model-based learning. The first one is the balance between exploration and exploitation. The second one is the risk involved in exploration. The lookahead-based exploration strategy presented in this work deals with these two issues. Every action is evaluated according to its long run contribution to the expected utility and to the knowledge regarding the opponent's strategy. Risky actions are detected by considering their expected outcome according to the alternative models of the opponent's behavior. The accumulated knowledge and utility throughout the expected game-paths are evaluated together and allow the agent to (almost) optimally control its behavior.

The lookahead-based exploration strategy drives exploration according to the player's uncertainty about its opponent model. Recently, some utility-based strategies has been suggested to guide exploration of an active agent by considering the effects of a given plan on the reduction of the agent's uncertainty about its model of the world. *Optimal experiments design* [7] is concerned with the design of experiments that are expected to minimize variances of the parameterized model, and therefore, to maximize the confidence in the given model. When applying this method to sequential decision making, the decision-maker can estimate how an addition of a new training example is expected to change the computed variance. Cohn [6] applies this method for selecting examples to train an artificial neural network. Karakoulas [11] presents a probabilistic algorithm that also applies a similar statistical selection procedure to

decide how much exploration is needed.

Model-based learning of interaction strategies in repeated-games has received a lot of attention in the game-theory literature. Gilboa & Samet [10] deal with bounded regular players. They describe a model-based learning strategy for repeated games that learns the best response against any regular strategy. Their procedure enumerates the set of all automata and chooses the current opponent model to be the first automaton in the sequence that is consistent with the current history. Exploration is achieved by designing a sequence of actions that distinguishes between the current model and the next consistent automaton in the enumeration. The risk involved in exploration is bypassed by assuming that the opponents' strategies are limited to strongly connected automata, where there are no "sinks" and there is always opportunities to regret. For such automata, the learning algorithm is guaranteed to converge to the best response in the limit. This learning procedure is based on exhaustive search in the space of automata, and therefore, is impractical for computational bounded agents.

The main role of the opponent model is to predict its behavior in the future. Choosing a proper class of strategies for modeling is essential for the success of the model-based strategy. If the model class is too restricted it will probably fail in prediction. On the other hand, a too general class will make the best response problem and the learning problem intractable. Often, there are many ways to model a given behavior. This paper concentrates on deterministic finite automata for modeling the agents' strategies. The question how the model-based framework can be extended for more powerful agents remains open for future research.

References

[1] R. Axelrod. *The Evolution of Cooperation*. Basic Books, New York, 1984.

[2] E. Ben-Porath. The complexity of computing a best response automaton in repeated games with mixed strategies. *Games and Economic Behavior*, 2:1–12, 1990.

[3] D. Carmel and S. Markovitch. Exploration and adaptation in multi-agent systems: A model-based approach. In *Proceedings of the fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 606 – 611, Nagoya, Japan, Aug. 1997.

[4] D. Carmel and S. Markovitch. Exploration strategies for model-based learning in multi-agent systems. Technical Report CIS9713, Technion, November 1997.

[5] D. Carmel and S. Markovitch. Model-based learning of interaction strategies in multi-agent systems. *Journal of experimental and theoretical Artificial Intelligence (JETAI) (to appear)*, 1997.

[6] D. A. Cohn. Neural network exploration using optimal experimental design. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 679–686. Morgan Caufmann, 1994.

[7] V. Fedorov. *Theory of optimal experiments*. New-york: Academic Press, 1972.

[8] Y. Freund, M. Kearns, Y. Mansour, D. Ron, R. Rubinfeled, and R. E. Schapire. Efficient algorithms for learning to play repeated games against computationally bounded adversaries. In *Proceedings of the Annual Symposium on the Foundations of Computer Science*, pages 332–341, 1995.

[9] I. Gilboa. The complexity of computing best response Automata in repeated games. *Journal of economic theory*, 45:342 –352, 1988.

[10] I. Gilboa and D. Samet. Bounded versus unbounded rationality: The tyranny of the weak. *Games and Economic Behavior*, 1:213–221, 1989.

[11] G. I. Karakoulas. Probabilistic exploration in planning while learning. In *Proceedings of the 11th Conference on uncertainty in Artificial Intelligence (UAI 95)*, pages 352 – 361, July 1995.

[12] A. Rubinstein. Finite automata play the repeated Prisoner's Dilemma. *Journal of Economic Theory*, 39:83–96, 1986.

[13] T. W. Sandholm and R. H. Crites. Multiagent reinforcement learning and the iterated Prisoner's Dilemma. *Biosystems Journal*, 37:147–166, 1995.

[14] S. B. Thrun. The role of exploration in learning control. In D. A. White and D. Sopfge, editors, *Handbook for Intelligent Control*. Multiscience Press Inc., 1992.