Efficient Learning of Multi-step Best Response

Bikramjit Banerjee
Department of Electrical Engineering & Computer Science
Tulane University
New Orleans, LA 70118. USA
banerjee@eecs.tulane.edu

Jing Peng
Department of Electrical Engineering & Computer Science
Tulane University
New Orleans, LA 70118. USA
jp@eecs.tulane.edu

ABSTRACT

We provide a uniform framework for learning against a *recent history adversary* in *arbitrary* repeated bimatrix games, by modeling such an agent as a Markov Decision Process. We focus on learning an optimal non-stationary policy in such an MDP over a finite horizon and adapt an existing efficient Monte Carlo based algorithm for learning optimal policies in such MDPs. We show that this new efficient algorithm can obtain higher average rewards than a previously known efficient algorithm against some opponents in the contract game. Though this improvement comes at the cost of increased domain knowledge, a simple experiment in the Prisoner's Dilemma game shows that even when no extra domain knowledge (besides that the opponent's memory size is known) is assumed, the error can still be small.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Theory, Performance

Keywords

Multiagent Learning, Game Theory, Efficient Learning

1. INTRODUCTION

Learning to play a repeated game has been explored in context of computationally resource-bounded adversaries [9, 13, 14], notably finite state automata. This literature focuses on *learning automata*, i.e., to deduce reply automata that accumulate near-optimal rewards against the given adversaries over infinite horizon. Learning over infinite horizons effectively assumes that the learner is going to have unlimited interactions with the adversary. In this paper, we address learning against resource-bounded adversaries over *finite horizons* reflecting the reality in Multiagent Systems (MAS) that agent-pairings are short lived. We address adversaries that use the recent history of play to deduce their instantaneous replies. We show that such opponents can be uniformly modeled as Markov Decision Processes (section 3) in arbitrary repeated matrix games, if their decision functions are stationary. We then adapt an efficient T-step policy learning algorithm for MDPs to work in adversarial repeated games through our framework (section 5). This algorithm is thus useful to a learner facing any recent history adversary no matter what the adversary's underlying stationary computational model is. We demonstrate the efficacy of this new efficient algorithm by showing that it can score higher average payoff than a previous algorithm that scored close to only half of the optimal payoff in the game of contract [14] (section 6). Comparison of this algorithm with ours shows that our algorithm uses slightly greater domain knowledge. However, if we limit this to a minimum (just assume that an upper bound on the opponent's memory size is known, which is also assumed in [14]), a simple experiment in Prisoner's Dilemma game shows that the performance of our learned policy is still close to optimal.

The contributions of this paper are

- Reduction of the problem of learning multi-step best response against a recent history adversary in repeated games to that of learning a finite-horizon non-stationary optimal policy in a large MDP or POMDP (when the opponent's actions are not observable).
- Proving that an efficient algorithm (known) for the domain above can efficiently learn a better policy than another known efficient algorithm against a past sacrifice adversary in the game of contract.
- Experimentally demonstrating that even with minimal domain knowledge the output policy is close in payofffs to an optimal policy in the Prisoner's Dilemma game against a variant of the Tit-for-Tat strategy.

2. BACKGROUND & DEFINITIONS

Here we provide definitions of key concepts for our work. We refer to A_1 and A_2 as the sets of possible actions of the two agents. A *mixed policy* is a probability distribution over A. If the entire probability mass is concentrated on a single action, it is also called a *pure policy*. We consider a situation where there is a learner interacting with other agents. At any given time the learner interacts with only one other agent, referred to as the *opponent* or *adversary*. Usually, a will refer to the learner's action and o to the opponent's action.

DEFINITION 1. *A bimatrix game is given by a pair of matrices,* (M_1, M_2)*, (each of size* $|A_1| \times |A_2|$*) where the payoff of the* kth

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.

agent for the joint action (a, o) is given by the entry

$$M_k(a, o),\ \forall (a, o) \in A_1 \times A_2,\ k = 1, 2.$$

An example of a bimatrix game, the Prisoner's Dilemma, is shown in Table 1 and will be used in experiments in section 7. We will make the usual assumption that all matrix payoffs are bounded in magnitude by some $r_{\max}$. We consider the problem of learning in the context of repeated play of a bimatrix game by two agents. This is called a *repeated game*. The policy of the learner will be written as π and that of the opponent as ρ. The expected payoff of the learner is

$$V(\pi, \rho) = \sum_{(a,o) \in A_1 \times A_2} \pi(a)\rho(o)M_1(a, o)$$

assuming M_1 is the learner's payoff matrix. In a repeated game, the goal of the learner is to deduce a policy that maximises $V(\pi, \rho)$.

Table 1: Prisoner's Dilemma Game. (a, b) in the $(i, j)th$ cell is the tuple of payoffs for Row agent and Column agent (in that order) for each combination of their actions $(i, j) \in \{C, D\} \times \{C, D\}$.

Actions	Cooperate (C)	Defect (D)
Cooperate (C)	(3,3)	(0,5)
Defect (D)	(5,0)	(1,1)

DEFINITION 2. *A* best response *of the learner to the opponent's policy, ρ, is a set of probability vectors $BR(\rho)$ defined as $BR(\rho) = \{\pi^* \in \Delta(A_1) | V(\pi^*, \rho) \geq V(\pi, \rho), \forall \pi \in \Delta(A_1)\}$, where $\Delta(A_1)$ is the set of probability distributions over A_1.*

The best response is the set of optimal policies that a learner can play to maximize its expected payoff given the opponent is playing ρ. However, this definition of best response does not account for the effect on the opponent of the learner's play. Usually the opponent will follow an algorithm that takes into account the learner's play in its decision process. So if h^t represents the history of play at time t, i.e. the sequence of joint actions played by the two players till time t, then the opponent's policy at time t, ρ_t, will be a function of h^t, given by $F(h^t) \in \Delta(A_2)$. Hence, best response should entail maximization in context of this function F.

We shall refer to histories that are of bounded length, w. Thus for $t > w$, h^t_w will mean the history of w *most recent* joint actions [1], with all previous actions forgotten, i.e., $h^t_w = h^{t-1}_{w-1}.\{a_{t-1}, o_{t-1}\}$. We call any adversary that chooses its action according to some stationary function F of h^t_w, a *recent history adversary* or RHA. Given such an opponent that is affected by the learner's actions, it is insufficient for the learner to decide on actions based on its immediate returns. It should form a more informed decision based on lookahead over the opponent's future behavior. In this paper, our goal will be to find a T-sequence of deterministic decision rules for the learner, $\pi^T = \pi(0), \pi(1), \ldots, \pi(T-1)$, that maximizes the undiscounted sum of expected rewards over a finite horizon, T after the first w steps of play (at which point, the history is h^0_w say),

$$U_{\pi_T}(h^0_w) = \sum_{t=0}^{t=T-1} V(\pi(t), F(h^t_w))$$

[1]If $t \leq w$, the history is the t most recent joint actions, though we only use histories for $t > w$.

3. MODELING AN RHA AS AN MDP

Generally learning against opponents that are finite automata (bounded rationality) has received significant attention [13] especially model based approaches [8, 7, 9]. Freund et. al have addressed more complex opponent strategies that could require exponential models if represented as finite automatons [14]. They have explored learning against recent history adversaries (RHA) in the contract game and provided efficient learning algorithms against probabilistic state automatons as opponent models, with small cover time [14]. We provide a uniform framework for representing RHA models that applies to any game and show how this sort of modeling leads to efficient learning.

Since the opponent's decision is based on the current history, h^t_w (over the w most recent joint actions), this history defines a *state* of the opponent. Let S be the set of all possible recent joint histories, i.e. all possible states of the opponent. Note that the opponent's decision at time t depends on its state at that time and the transition to the next state, h^{t+1}_w depends on the actions of both agents. We assume that the opponent uses a stationary function $F(h^t_w)$ to compute its mixed policy, ρ, at time t and that the learner chooses a deterministic policy a_t simultaneously. Thus the transition from state h^t_w to state h^{t+1}_w is characterized by the distribution ρ, and there are $|A_1||A_2|$ possible next states constituting the support of the product of ρ and the learners decision. This gives a transition function $\delta : S \times A \to \Delta(S)$, identical to the usual transition function of an MDP. Thus if, $h^{t+1}_w = h^t_{w-1}.\{a_t, o_t\}$ where o_t is the opponent's action, $o_t \sim F(h^t_w) \equiv \rho_t$, then $\delta(h^t_w, a_t, h^{t+1}_w) = F(h^t_w, o_t)$. In other words, the opponent's decision function *is* the transition function of this model. As in any MDP learning problem, this distribution is unknown.

Now the reward function of this transition model is given by $R : S \times A \to \Re$, since the learner's reward for choosing actions a_t in state h^t_w is given by $R(h^t_w, a_t) = V(a_t, F(h^t_w)) = V(a_t, \rho_t)$. The above transition and reward functions together with the state and action sets (S and A) define a Markov Decision Process for the learner in the repeated game and it is sufficient for the learner to learn the optimal policy in this MDP to play the game well. We call this MDP an *Adversary Induced MDP* or AIM in short.

Note that the state space of the AIM is exponential in w, viz., $|S| = |A|^{2w}$. Hence learning the AIM (R, F unknown) using an efficient MDP learning algorithm generally cannot be polynomial in w, since even the best of such algorithms are polynomial in $|S|$. However, many of the states in S will possibly never be visited since the opponent's distribution $\rho_t = F(h^t_w)$ may not have full support; so practical efficient learning may be possible. Also note that the AIM is not necessarily ergodic under a given policy of the learner; hence it is generally not a unichain, even though every state in S is reachable from every other state in at most w transitions under *appropriate* policies of the learner.

4. LEARNING IN MDPS AND POMDPS

A strong body of literature exists on learning MDPs. An MDP whose transition and reward functions are known can be solved for optimal policy by linear programming in time $\mathcal{P}(|S|, |A|)$ [4], $\mathcal{P}$ stands for some polynomial function. Online learning (to accumulate near-optimal payoffs) in unknown MDPs for undiscounted settings have been addressed by E^3 [17] and $R - MAX$ [6] algorithms that work in time $\mathcal{P}(|S|, |A|, T_\epsilon)$ where T_ϵ is the *ϵ-return mixing time* [17]. Efficient online learning in unknown MDPs has also been addressed in [12]. Tesauro [20] has extended Q-learning to MAS (Hyper-Q learning) where values of mixed strategies are learned and Bayesian inference is used to estimate opponent strat-

egy. He has argued that some history independent opponents ($w \leq 1$) would present a stationary MDP environment (in a stochastic game sense, whereas our framework is for repeated games) to the learner to assure convergence.

Efficient offline algorithm for reinforcement learning has been explored in the PAC framework by Fiechter [11]. More recently Pivazyan and Shoham [18] have provided a uniform framework for offline reinforcement learning that improves on [11] and provides a polynomial dependence on $\min(T_\epsilon, r_s)$ where r_s is the *spectral radius* of the MDP, for undiscounted setting. For the discounted setting they prove a $\mathcal{P}(|S|, |A|, r_s)$ bound on the sample complexity.

All the above methods address infinite horizon problems, i.e. where the goal is to (almost) maximize the total asymptotic return, discounted or averaged. A finite horizon version of this problem seeks to maximize the sum (discounted or averaged) of returns over the first T steps. Kearns et al [16] provide offline algorithm for POMDPs - the trajectory tree method - that is independent of $|S|$ but exponential in T. Bagnell et. al [2] have improved on this bound to polynomial in T by assuming a known baseline distribution on states reflecting how often a "good" policy should visit those states. The problem of learning optimal mappings from states to actions in MDPs is replaced in POMDPs by that of learning optimal mappings from observations to actions. Observations generate *belief states*, or probability distributions over the state space. In [2], this distribution is assumed to be given as the baseline distribution and a policy that optimizes reward-sums relative to this distribution is learned. This allows for a Monte Carlo algorithm that is independent of $|S|$. Since in our problem the state space is exponential, we wish to have an algorithm that is independent of $|S|$ too. Assuming that the baseline distribution of [2] encodes some form of domain knowledge, their POMDP algorithm is actually suitable for our large MDP problem, even though the states are completely observable. It also allows for the possibility that the learner does not observe the opponent's actions turning it into a true POMDP [2] where this approach still works. Hence we adopt this approach in this paper.

5. LEARNING IN AIM OVER FINITE HORIZON

In this paper we consider the finite horizon version of MDP learning for two primary reasons. Firstly, learning over infinite horizon makes little sense in real multiagent systems which are dynamic with the set of opponents ever-changing. Agent interactions are effectively ephemeral where learning to perform over a finite horizon T, given a learner has an idea of how long it is going to interact with a given opponent, may enhance the efficieny of learning. Secondly, learning over infinite horizons can complicate the problem of learning best response. Consider the example of the "combination-lock" argument in Fortnow and Whang [13]. An RHA with w size history can play an action that is good for the learner only when the learner's last w actions match a certain pattern. Searching for this pattern can be $O(|A_1|^w)$ in the worst case, but this would be necessary to be able to play well for ever after. However if the learner knows that it needs to optimize payoff over only a finite horizon, it may not need/want to explore for the key pattern even if $T > |A_1|^w$. Using the method that we outline next, an agent can learn to achieve near optimal (restricted sense) rewards in polynomial time given some domain information.

Our goal is to learn to optimize the sum of rewards of a learner through the T interactions following the first w interactions. We assume that the MAS is dynamic in the sense that the opponents can be changed at the learner's will. The example scenario below describes a situation where the assumptions here make sense. We also assume that there is a class ($\mathcal{C}$) of recent history opponents for the learner to choose from, who do not necessarily play the same strategies, only strategies whose mean matches F. So if the probability of action o of any opponent $c \in \mathcal{C}$ is given by $F_c(., o)$, then $E_{\mathcal{C}} F_c(., o) = F(., o),\ \forall o$.

An Example Scenario

A motivating example for such a scenario is a market with multiple sellers where the buyer is interested to learn an optimal negotiation strategy for buying an item, say a digital camera. Since such an item has a pretty much fixed set of selling points, e.g., megapixels, optical zoom factor, battery life etc., it is reasonable to assume that the negotiation strategies of the sellers will not vary much. In this example setup there is no intermediate reward throughout any sequence of online interactions, just one reward at the end of each sequence, viz. an inverse function of the price negotiated. Note however, that this reward does indeed depend on the entire sequence of negotiations. The buyer can go from one seller to another without committing to buy from any and learn offline how best to negotiate for the best deal from a number of online interactions each with a distinct seller.

The Algorithm

Our approach is to use the Monte Carlo based μ-PolicySearch algorithm from [15] to construct a T sequence of non-stationary deterministic policies, $\pi_T = \{\pi(., 0), \pi(., 1), \ldots, \pi(., T-1)\}$, for the AIM. We call this algorithm μ-PSAIM (μ-Policy Search in AIM), and is detailed in Table 2. First π_T is initialized randomly. To calculate $\pi(., t)$ at time t, using a baseline distribution μ like μ-PolicySearch, the learner generates a history and executes its actions in the w-window history. At the end, it enters a state in the AIM (step 2(a)ii in Table 2), executes a random action and thereafter executes $T-t-1$ steps of on-line interactions with the opponent, following the corresponding portions of the policy π_T. Then it switches the opponent and repeats the procedure m times. After observing m samples of $(T-t)$-step reward-sums (estimate of $Q_{\pi,t}(s, a)$ which represents the undiscounted sum of rewards obtained by playing a in s at t and then following π_T for remaining $T-t-1$ steps) it can construct an unbiased estimate of a quality function, $Q_{\pi,t}(\mu, p) = E_{s\sim\mu(.|t)} E_{a\sim p(.|s)} Q_{\pi,t}(s, a)$ (see [2, 15]), $p \in \Pi_1$, that is $\left(\frac{\epsilon}{r_{\max} T}\right)$-correct with probability at least $1-\delta$ [3], if

$$m = O\left(\frac{|A|^2 T^2}{\epsilon^2}(\log|\Pi_1| + \log\frac{1}{\delta})\right) \quad (1)$$

where Π_1 is the class of deterministic decision rules for the learner whose size can be doubly exponential in w in the worst case, but we consider a subset of this class of fixed size that is predisposed toward likelier states from μ. With the help of this quality function the learner can compute the optimal action rule $\pi(., t)$, and executing this entire procedure in a loop for $t = T-1, T-2, \ldots, 0$, it can construct the required T-sequence policy. Moreover, if μ is close to the distribution over most favorable states [2, 15] in the AIM, then the learner has a $T+w$ step optimal plan for interactions with the opponent class. Observe that only that portion of π_T, which has al-

[2]It can be observed that the algorithm explained later in Table 2, uses the observation of opponent's actions only to distinguish states and if this information is not available, it can still work since it has access to a μ-reset model.

[3]We refer to this as (ϵ, δ)-learning, $\epsilon, \delta > 0$.

ready been computed, is used at any t for computing $\pi(.,t)$; hence the random initialization of π_T is merely a technical step. The indicator function I (step 2(b)) gives 1 if argument is true, else 0. The opponent switch effectively provides a reset button. Note that reset models have been studied before in Multiagent Learning [10].

In context of the motivating example above, it is useful to observe that the price negotiated at the end of each interaction sequence would generate the estimate Q_i directly in step 2(a)iii; there is no intermediate reward to sum over to generate this estimate. Also note that each sequence starts at t and proceeds until T, the final step of negotiation, so that an estimate can indeed be generated at the end of every sequence.

1. Input Π_1, T. Randomly initialize $\pi_T = \{\pi(.,0), \pi(.,1), \ldots, \pi(.,T-1)\}$.
2. For $t = T-1, \ldots, 0$ do
   (a) For $i = 1, 2, \ldots, m$ do
      i. Select an opponent randomly without replacement from $\mathcal{C}$.
      ii. Generate a history h_w according to $\mu(.,t)$, ignore the opponent's actions and play the learner's sequence of w actions.
      iii. Reach a state (history) s_i, generate an action $a_i \sim$ Uniform, follow this action and then policy π_T for the remaining $T-t-1$ steps. Note the rewards and generate an estimate Q_i for $Q_{\pi,t}(s_i, a_i)$.
   (b) With the m samples of the form $\{(s_i, a_i, Q_i)\}$, define
   $$Q_{\pi,t}(\mu, p) = \frac{|A|}{m} \sum_i Q_i I(p(s_i) = a_i)$$
   (c)
   $$\pi(.,t) = \arg\max_{p \in \Pi_1} Q_{\pi,t}(\mu, p)$$
3. Return T-step policy $\pi_T = \{\pi(.,0)\pi(.,1)\ldots\pi(.,T-1)\}$.

Table 2: The μ-PSAIM algorithm

THEOREM 1. *The total number of transitions observed by the algorithm μ-PSAIM is given by*

$$\mathcal{P}(|A|, T, w, |\Pi_1|, 1/\epsilon, 1/\delta)$$

Proof : The dependence on w is easily established noting that step 2(a)ii is $O(w)$. The algorithm outputs a T step plan which together with the given baseline distribution μ implies a $T+w$ step plan. The m samples generated in Table 2 (step 2(a)) are not identically distributed since the transition distribution is varying but with mean identical to F. Therefore the quality estimates Q_i are still independent and unbiased (because $EF_c = F, c \in \mathcal{C}$) and $AQ_iI(p(s) = a)$ is an unbiased estimate of $Q_{\pi,t}(\mu, p)$. Thus Hoeffding inequality and union bounds are applicable. The rest now follows directly from Theorem 6.3.3 in [15]. □

THEOREM 2. *If p_T is the T step policy returned by μ-PSAIM, then for any T-step policy p'_T and any history h_w,*

$$U_{p_T}(h_w) \geq U_{p'_T}(h_w) - \epsilon - r_{\max} T \| d_{p'_T, h_w} - \mu \|$$

where $d_{p'_T,h_w}$ is the state distribution of the T-step policy p'_T starting from state h_w, $\|\|$ is the L_1-norm, and ϵ was introduced in context of equation 1 before.

Proof : This follows from Theorem 6.3.1 in [15]. □

This result establishes the quality of the policies learned by μ-PSAIM. The value of μ-PSAIM's output is close to that of any T-sequence policy whose future state distribution (d) is close to μ. Hence, if μ is a "good" distribution, μ-PSAIM's payoff will be close to optimal. Note however, that the notion of optimality is in context of the given class of deterministic decision rules, Π_1.

6. PAST SACRIFICE ADVERSARIES IN CONTRACT

Here we consider the example of a *past sacrifice adversary* or PSA [14] in the game of contract. This game allows two actions to each of the players, written as 0 and 1. The game payoffs of the learner are $M_1(1,1) = 1, M_1(0,0) = M_1(0,1) = M_1(1,0) = 0$. Unlike matching pennies, as discussed in [14], it is insufficient to learn a best response to the opponent since that does not guarantee the high payoff (i.e., 1) to the learner. The learner must also learn to influence the opponent to play action 1 frequently enough to score high *average* payoff. Whether this is possible (and in a sustainable manner) depends on the type of the opponent.

A past sacrifice adversary is defined [14] as a class of recent history adversaries whose strategy is given by a boolean formula $F_{\mathcal{I}}(h_w^t)$, where $\mathcal{I} \subseteq \{1, \ldots, w\}$ is an index set,

$$F_{\mathcal{I}}(h_w^t, 1) = \bigvee_{i \in \mathcal{I}, i \leq t-1} (\neg a_i \wedge o_i), t = 2, 3, \ldots$$

with $F_{\mathcal{I}}(h_w^1 = \emptyset, 1) = 1$. $F_{\mathcal{I}}(h_w^t, 0) = 1 - F_{\mathcal{I}}(h_w^t, 1)$, i.e., the opponent's strategy is deterministic at all times. The opponent essentially scans the history h_w^t to identify if the action pair $(0,1)$ was played at any time in the index set and plays $o_t = 1$ if so, else plays $o_t = 0$. The action pair $(0,1)$ is called a *sacrifice* since the learner passes the opportunity of playing 1 when the opponent plays 1 to score the high payoff, and plays 0 instead. Hence the opponent is predisposed to rewarding past sacrifices of the learner by playing 1 that allows the learner to score payoff 1 by playing 1. However, the player must be cautious since greedily playing 1 will deplete the history of any sacrifices making any future payoff of 1 impossible.

Freund et. al have shown [14] that after playing 0 for at most w^3 time steps, the learner can establish a pattern of sacrifices every g rounds, where $g = \text{GCD}(\mathcal{I})$ is the greatest common divisor of the indices in $\mathcal{I}$. So the optimal payoff (average) over infinite horizon that can be obtained against such an adversary by *any* player is at most $1/g$ (Lemma 3.2 [14]). Freund et. al have also presented a $1/2$-competitive (i.e., achieves at least half of the optimal payoff after some time) efficient algorithm to play contract (Theorem 3.1 [14]). Thus the lower bound of the payoff realized by their algorithm is at best $\frac{1}{2g}$. We show below that μ-PSAIM can learn to achieve higher bound on the average payoff than $\frac{1}{2g} - \epsilon$ in some cases for any given $\epsilon > 0$. But before that, in the next paragraph we present a short comparison of the two algorithms regarding their nature and assumptions in order to justify comparing their performances.

The main difference between the algorithm in [14] and μ-PSAIM is that the former is an online learner while the latter learns of-

fline from repeated online interactions. The former is designed to learn without using information about the opponent's state while μ-PSAIM infers the opponent's state from the observation of the last w joint actions [4]. Note that this is a parallel characterization of the opponent's actual state, e.g., the opponent could be using a small DFA model for its decisions while μ-PSAIM would use an exponential state-space for its opponent model. This is not wasteful, firstly because it does not need to sample the entire state space, and secondly since, for example, some PSA using DFA could actually require exponentially many states [14]. On the other hand, this uniformity in opponent-modeling allows μ-PSAIM to work with any RHA, not just a PSA, while the algorithm in [14] is specifically designed to learn against a PSA only. Note that [14] also mentions that if the learner knows $|\mathcal{I}|$ then it can achieve higher average payoff than $1/2g$, but μ-PSAIM can do that against some PSA *without* knowing $|\mathcal{I}|$ as we demonstrate below. Lastly, both algorithms assume that w is known.

THEOREM 3. *Against some PSA opponents, μ-PSAIM can (ϵ, δ)-learn to achieve higher average payoff than $\frac{1}{2g} - \epsilon$ in contract.*

Proof : The proof is by example. We first establish the bounds on the average payoff achievable by μ-PSAIM, u, through the following lemma and then show that in some cases this value can be greater than $\frac{1}{2g} - \epsilon$ (although always $< \frac{1}{g}$).

LEMMA 4. *If the baseline distribution μ is accurate and $T > \left(\frac{|\mathcal{I}|}{|\mathcal{I}|-1}\right)(w - \min\mathcal{I} + 1)$, μ-PSAIM will efficiently (ϵ, δ)-learn a policy that pays at least $u - \epsilon$ on the average after time w in contract, such that*

$$\frac{|\mathcal{I}| - 1}{w} \leq u \leq \frac{|\mathcal{I}| - 1}{|\mathcal{I}|}$$

Proof : Let $\mathcal{I} = \{i_1, i_2, \ldots, i_r\}, 1 \leq i_1 < i_2 < \ldots i_r \leq w$. The optimal policy in the given AIM can be seen as follows. The best baseline $\mu(., 0)$ for the learner is to play 0 for all of the first w steps. The learner first needs to insert a sacrifice at the most recent round in the history. This is trivial at step $w + 1$ since in the previous w plays, the opponent has played 1 at every step in $\mathcal{I}$ and the learner has played a 0 in all those steps (by μ), implying that $o_{w+1} = 1$. So if the learner chooses $a_{w+1} = 0$ then round $w+1$ is a sacrifice. Now the learner can start playing 1 and collect reward 1 as this sacrifice passes through the indices $i_r, i_{r-1} \ldots, i_2$. This gives it a maximum of $r - 1$ reward points over $w - i_2 + 1$ steps. Now when this sacrifice reaches index i_1 the learner must play 0 to insert a fresh sacrifice in the most recent round in history and repeat the above strategy. Thus the learner can score $r - 1$ reward in $w - i_1 + 1$ steps giving an average of

$$u = \frac{r - 1}{w - i_1 + 1}$$

This pattern of reward can be maintained indefinitely. An example with $w = 4$, $\mathcal{I} = \{2, 3\}$ is shown in Figure 1. Evidently, given the assumptions, Theorem 2 implies the learner will (ϵ, δ)-learn to attain payoff u. The result is established by noting that $1 \leq i_1 \leq w - r + 1$. □(Lemma 4).

Note that if $T \leq \left(\frac{|\mathcal{I}|}{|\mathcal{I}|-1}\right)(w - i_1 + 1)$, then it is not worthwhile to learn the optimal policy in AIM; instead the learner may just play 1 all the time making sure that it scores at least r reward in the initial w steps. Consequently, the best baseline distribution $\mu(.,.)$ for this scenario will concentrate the mass on playing all 1's.

[4]Their algorithm also needs to observe the opponent's actions at least in the initial rounds [14]

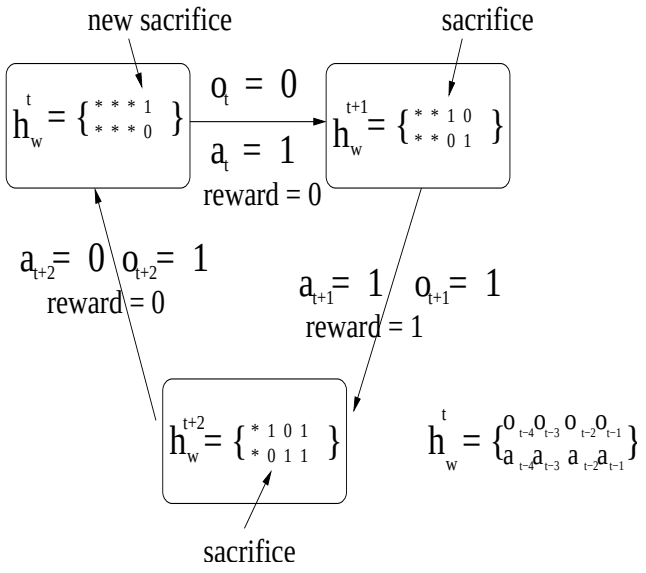

Figure 1: The sustainable payoff loop in Contract for $w = 4, \mathcal{I} = \{2, 3\}$.

According to the results in [14], the bound on u cannot exceed $1/g$. Observe that the maximum value of $u = \frac{r-1}{r} < 1$ is reached when $i_1 = w - r + 1$ or when $r = w$. It is easy to verify that in both extreme cases, $g = 1$. In general the average payoff that μ-PSAIM gets close to will always be less than $1/g$. However the bound can be higher than $1/2g - \epsilon$. For example, if $w = 5$ and $\mathcal{I} = \{3, 4, 5\}$, then $g = 1$, but $u = \frac{2}{3} > \frac{1}{2g}$. Thus μ-PSAIM can efficiently (ϵ, δ)-learn to achieve higher average payoff than that algorithm in some cases. □(Theorem 3).

7. EXPERIMENTS IN PRISONER'S DILEMMA

Several known algorithms fall into the class of RHA. For instance, IGA [19], WoLF-IGA [5] and ReDVaLeR [3] use stationary functions on only one step history of policy followed by the opponent as well as its own policy from last step, so a μ-PSAIM learner can learn a near-optimal T-step deterministic policy using w as low as 1. Another example is the *Tit-For-Tat* (TFT) strategy that is known to perform very well in the Prisoner's Dilemma game shown in Table 1 [1]. It also uses a window of size 1 and furthermore its decisions are independent of its own past choices. Therefore, the μ-PSAIM learner does not even need to observe the opponent's (TFT) actions to learn a T-step best response, because F is independent of TFT's policy.

We have experimented with a variant of the TFT algorithm, viz. *Tit-For-Two-Successive-Tats* (TFTST), which uses $w = 2$ and its policy is given by

$$F(h_2^t) = \begin{cases} \text{defect if the opponent defected in the last two rounds.} \\ \text{cooperate otherwise.} \end{cases}$$

The optimal strategy ($\pi_T = \{\pi(., 0), \pi(., 1), \ldots, \pi(., T-1)\}$ say) against TFTST in Prisoner's Dilemma would be to defect in the last two rounds and alternate between cooperate and defect in the previous rounds such that there are no more consecutive defects. The ideal $\mu(., t)$ distribution would be to distribute the mass uniformly

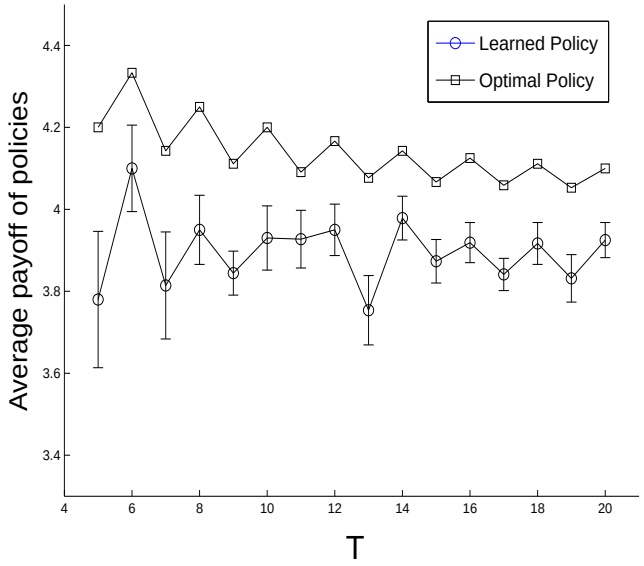

Figure 2: Average payoff plots of the optimal policy and the learned policy in Prisoner's Dilemma game against a TFTST opponent.

over all states of the form

$$\left\{ \begin{array}{cc} * & * \\ \pi_{t-2} & \pi_{t-1} \end{array} \right\}$$

and 0 for the rest of the states. However, this μ as a form of domain knowledge has a strong dependence on the knowledge of the optimal policy beforehand, making the endeavor useless. So we need to use a different $\mu(.,t)$ that does not assume knowledge of the optimal policy. On the other hand an inaccurate μ will contribute to the error in Lemma 2. Though this error when averaged over T is at most a constant, the value of this constant could be large. A "good" choice for μ is the stationary state distribution of a "good" policy, such as distributing the weight on alternating actions CD and DC and 0 on DD and CC. This μ too assumes some domain knowledge that alternating actions are "good" in this environment. Rather than using such μ distributions we wish to evaluate with one that assumes no domain knowledge. We have experimented with a suboptimal μ that effectively *does not assume any domain knowledge* and found that the error of the output policy is still reasonably small. In our experiments, we have used a stationary distribution (in the sense $\mu(.,t)$ is independent of t) that distributes mass uniformly over the *entire state space*. A little digression is in order here to discuss this choice. If μ contains no domain information (as in this case), Π_1 cannot be reduced in any useful way and recall that it can be doubly exponential in w. Though this can be tackled in small problems such as Prisoner's Dilemma with small w, some domain knowledge will be necessary in larger problems for tractability with regard to Π_1.

In the experiments, we varied T from 5 to 20, and noted the performance of the policy output from each of 10 runs of μ-PSAIM and plotted their average (over these 10 runs) and standard deviation in Figure 2. First observe that the optimal average payoffs are $4 + \frac{2}{T}$ and $4 + \frac{1}{T}$ for T being even and odd respectively, assuming that the learner always cooperates in the first w rounds to set up the opponent's memory [5]. Not only does the curve from the learned policies follow the optimal policy's pattern closely, the payoff is *always* higher than what mutual cooperation (note that cooperation is a safe strategy against the given opponent) would have yielded and way better than mutual defection (the equilibrium strategy). We used $m = 20T^2$ in these experiments which should have meant significant error (by equation 1) but in practice the error turns out to be quite low, especially given that our μ contained no information about the domain.

[5]This was also assumed in testing the output policies, but note that only the performance from last T rounds matter here.

8. CONCLUSION

We have provided a uniform framework (AIM) for learning against a *recent history adversary* in *arbitrary* repeated bimatrix games, by modeling such an agent as a Markov Decision Process. We have focused on learning an optimal policy in such an MDP over a finite horizon and adapted the existing efficient sampling-based μ-PolicySearch algorithm [2, 15] for learning optimal policies in such MDPs. We have shown that our new efficient algorithm, μ-PSAIM, can obtain higher average rewards than a previously known efficient algorithm [14] against some opponents in the repeated *contract* game. We have shown experimentally that even when μ incorporates no domain knowledge the output policy can perform close to optimal.

9. REFERENCES

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

[2] J. A. Bagnell, S. Kakade, A. Y. Ng, and J. Schneider. Policy search by dynamic programming. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2003.

[3] B. Banerjee and J. Peng. Performance bounded reinforcement learning in strategic interactions. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-04)*, pages 2–7, San Jose, CA, 2004. AAAI Press.

[4] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, Belmont, MA, 1995.

[5] M. Bowling and M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136:215 – 250, 2002.

[6] R. I. Brafman and M. Tennenholtz. R-max - A general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213 – 231, 2002.

[7] D. Carmel and S. Markovitch. Learning models of intelligent agents. In *Thirteenth National Conference on Artificial Intelligence*, pages 62–67, Menlo Park, CA, 1996. AAAI Press/MIT Press.

[8] D. Carmel and S. Markovitch. How to explore your opponent's stratey (almost) optimally. In *Proceedings of the Third International Conference on Multiagent Systems*, pages 64–71, Los Alamitos, CA, 1998. IEEE Computer Society.

[9] D. Carmel and S. Markovitch. Model-based learning of interaction strategies in multiagent systems. *Journal of Experimental and Theoretical Artificial Intelligence*, 10(3):309 – 332, 1998.

[10] V. Conitzer and T. Sandholm. AWESOME: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[11] C. Fiechter. Efficient reinforcement learning. In *Proceedings of the 7th Annual ACM Conference on Computational*

Learning Theory, pages 88 – 97, New Brunswick, NJ, 1994. ACM Press.

[12] C. Fiechter. Expected mistake bound model for on-line reinforcement learning. In *Proceedings of the 14th International Conference on Machine Learning*, pages 116 – 124, 1997.

[13] L. Fortnow and D. Whang. Optimality and domination in repeated games with bounded players. In *Proc. Symp. Theory of Computation*, Montreal, Canada, 1994.

[14] Y. Freund, M. Kearns, Y. Mansour, D. Ron, R. Rubinfeld, and R. Schapire. Efficient algorithms for learning to play repeated games against computationally bounded adversaries. In *Proceedings of the 36th IEEE Symposium on the Foundations of Computer Science*, pages 332 – 341. IEEE Press, 1995.

[15] S. Kakade. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, University College London, 2003.

[16] M. Kearns, Y. Mansour, and A. Ng. Approximate planning in large pomdps via reusable trajectories. In *Advances in Neural Information Processing Systems 12*. MIT Press, 2000.

[17] M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. In *Proceedings of the 15th International Conference on Machine Learning*, pages 260 – 268. Morgan Kaufmann, 1998.

[18] K. Pivazyan and Y. Shoham. Polynomial-time reinforcement learning of near-optimal policies. In *Proc. National Conf. on Artificial Intelligence*, 2002.

[19] S. Singh, M. Kearns, and Y. Mansour. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 541–548, 2000.

[20] G. Tesauro. Extending Q-learning to general adaptive multi-agent systems. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.