Effective Learning in Adaptive Dynamic Systems

Andriy Burkov and **Brahim Chaib-draa**
DAMAS Laboratory
Laval University
G1K 7P4, Quebec, Canada
{burkov,chaib}@damas.ift.ulaval.ca

Abstract

Classically, an approach to the policy learning in multiagent systems supposed that the agents, via interactions and/or by using preliminary knowledge about the reward functions of all players, would find an interdependent solution called "equilibrium". Recently, however, certain researchers question the necessity and the validity of the concept of equilibrium as the most important multiagent solution concept. They argue that a "good" learning algorithm is one that is efficient with respect to a certain class of counterparts. Adaptive players is an important class of agents that learn their policies separately from the maintenance of the beliefs about their counterparts' future actions and make their decisions based on that policy and the current belief. In this paper we propose an efficient learning algorithm in presence of the adaptive counterparts called Adaptive Dynamics Learner (ADL) which is able to learn an efficient policy over the opponents' adaptive dynamics rather than over the simple actions and beliefs and, by so doing, to exploit this dynamics to obtain a higher utility than any equilibrium strategy can provide. We tested our algorithm on a big set of the most known and demonstrative matrix games and observed that ADL agent is highly efficient against Adaptive Play Q-learning (APQ) agent and Infinitesimal Gradient Ascent (IGA) agent. In self-play, when possible, ADL is able to converge to a Pareto optimal strategy that maximizes the welfare of all players instead of an equilibrium strategy.

Introduction

Classically, an approach to the policy learning in multiagent systems (MAS) supposed that the agents, by means of interactions and/or by using preliminary knowledge about the reward functions of all players, would find an interdependent solution called "equilibrium". One of the most used concepts of equilibrium is the Nash equilibrium where each agent in a MAS plays its best response to the other players' strategies and a unilateral deviation of a player from the equilibrium strategy decreases its own utility. There are two basic approaches to finding a Nash equilibrium. The first one is a game theoretic approach which supposes the complete knowledge of the reward structure of the underlying game by all the agents. In such an approach, each agent calculates an equilibrium, by using mathematical programming, and all the agents play on it. But a problem arises when there are several tantamount equilibria in a game and the agents have calculated the different ones. Another problem is that the agents, by calculating an equilibrium, suppose that the other agents are rational and, thus, they will also follow this solution. But what if certain agents are not rational, or play a fixed strategy, or evolve according to some fixed rules, and what if some agents know (or are able to deduct) this and could exploit this knowledge to augment their utilities? As yet, there is no equilibrium concept which can answer this question. The second approach to finding an equilibrium is the adaptive one, which supposes that the agents learn by adapting to each other in self-play (i.e. all the agents use the same learning algorithm) and do not know the reward structure of the game and are only able to make actions and observe their own rewards and, in some approaches, the actions made by the others. It was shown that certain algorithms of this class converge to a Nash equilibrium (or to a utility that is equivalent to the utility of a Nash equilibrium). Among these algorithms the most outstanding are Joint-Action Learning (Claus & Boutilier 1998), Infinitesimal Gradient Ascent (IGA)[1] (Singh, Kearns, & Mansour 1994), Policy Hill-Climbing (Bowling & Veloso 2002) and Adaptive Play Q-learning (Gies & Chaib-draa 2006) (a Q-learning based extension of the Adaptive Play algorithm (Young 1993)). The adaptive players[2] learn their policies separately from the maintenance of the beliefs about their counterparts' future actions and make their decisions based on that policy and the current belief. These decisions can be in pure or in mixed strategies depending on the algorithm in question.

Recently, certain researchers (Shoham, Powers, & Grenager 2003) question the necessity and the validity of the concept of equilibrium as the most important multiagent solution concept. They rather point out the efficiency of a particular learning algorithm versus a certain class of coun-

[1] IGA is the only algorithm among those listed which requires a complete knowledge of the opponent's current strategy and reward structure of the game to calculate the gradient.

[2] To discriminate between adaptive player as a member of a class of learning agents and Young's Adaptive Player, we will write "adaptive player" or "adaptive algorithm" (in lower case) to denote the former and Adaptive Player (with a capital letter) for the latter.

terparts. The *adaptive players* is such a class and in this paper we propose an efficient algorithm of learning in presence of the adaptive counterparts called Adaptive Dynamics Learner (ADL). This algorithm is able to learn an efficient policy over the opponents' adaptive dynamics rather than over the simple actions and beliefs and, by doing so, to exploit these dynamics to obtain a higher utility than any equilibrium strategy can provide. We tested our algorithm on a set of the most known and demonstrative repeated matrix games from the GAMUT test suite (Nudelman *et al.* 2004) and the results show that ADL agent is highly efficient in self-play and against APQ and IGA agents[3].

The rest of the paper is organized as follows. First, we will describe IGA and APQ learning algorithms. Then, we will present our novel approach to the learning of the adaptive dynamics in MAS. Then, by comparative testing we will demonstrate that our algorithm exploits IGA and APQ players in adversarial games by remaining rational and highly efficient in the other games, versus the stationary opponents and versus itself as well.

Adaptive Dynamics

In this section we describe two algorithms which represent two basic subclasses of adaptive learning algorithms: those that are able to learn a pure strategy and those that are able to learn a mixed one. These are APQ and IGA algorithms respectively. But first we present the notation we use. A (normal form) stage game is a tuple $(n, \mathcal{A}^{1\dots n}, R^{1\dots n})$, where n is the number of players, $\mathcal{A}^j$ is the strategy space of player $j, j = 1 \dots n$, and the value function $R^j : \times\mathcal{A}^j \mapsto \mathbb{R}$ which defines the utility for player j of a joint action $\vec{a} \in \mathbf{A} = \mathcal{A}^1 \times \dots \times \mathcal{A}^n$.

A mixed strategy for player j is a distribution $\pi \in \Delta(\mathcal{A}^j)$, where $\Delta(\mathcal{A}^j)$ is a set of distributions over the player's action set $\mathcal{A}^j$, i.e., $\pi(a^j)$ is the probability for player j to select some action a^j. A strategy is *pure* if $\pi(a^j) = 1$ for some a^j. A *strategy profile* is a collection $\Pi = \{\pi^j : j = 1 \dots n\}$ of all players' strategies. A *reduced profile for player* j, $\Pi^{-j} = \Pi \backslash \{\pi^j\}$, is a strategy profile of all players except j, and $\Pi(a^{-j})$ is the probability for players $k \neq j$ to play a joint action $\vec{a}^{-j}$, where $\vec{a}^{-j}$ is $\langle a^k, k = 1 \dots n, k \neq j \rangle$. Given a player j and a reduced profile Π^{-j}, a strategy $\hat{\pi}^j$ is a *best reply* (BR) to Π^{-j} if the expected utility of the strategy profile $\Pi^{-j} \cup \{\hat{\pi}^j\}$ is maximal for player j. Since a best reply may not be unique, there is a set of best replies of player j to a reduced profile Π^{-j} which is denoted as $BR^j(\Pi^{-j})$. More formally, the expected utility of a strategy profile Π for a player j is given by:

$$U^j(\Pi) = \sum_{a^j \in \mathcal{A}^j} \pi_i(a^j) \sum_{a^{-j}} R(\langle a^j, \vec{a}^{-j} \rangle) \Pi^{-j}(\vec{a}^{-j}) \quad (1)$$

where Π is $\Pi^{-j} \cup \{\pi^j\}$ and $R(\langle a^j, \vec{a}^{-j} \rangle)$ is the value that player j receives if the joint action $\vec{a} = \langle a^j, \vec{a}^{-j} \rangle$ is played by all players. In this case, a best reply of player j to the reduced profile Π^{-j} is a strategy $\hat{\pi}^j$ such that:

$$U^j(\Pi) \geq U^j(\Pi') \quad \forall \pi^j \neq \hat{\pi}^j$$

where Π is $\Pi^{-j} \cup \{\hat{\pi}^j\}$ and Π' is $\Pi^{-j} \cup \{\pi^j\}$ respectively. A repeated game (or iterated game) is a game which consists of a certain number of repetitions of some stage game.

Adaptive Play *Q*-learning

Formally, each player j playing Adaptive Play saves in memory a history $H_t^j = \{\vec{a}_{t-p}^{-j}, \dots, \vec{a}_t^{-j}\}$ of last p joint actions played by the other players. To select a strategy to play at time $t+1$ each player randomly and irrevocably samples from H_t^j a set, $\hat{H}_t^j = \{\vec{a}_{k_1}^{-j}, \dots, \vec{a}_{k_l}^{-j}\}$, of examples of length l and calculates the empiric distribution $\hat{\Pi}^{-j}$ as an approximation of the real reduced profile, Π^{-j}, of strategies played by the other players, using the following:

$$\hat{\Pi}^{-j}(\vec{a}^{-j}) = \frac{C(\vec{a}^{-j}, \hat{H}_t^j)}{l} \quad (2)$$

where $C(\vec{a}^{-j}, \hat{H}_t^j))$ is the number of times that the joint action $\vec{a}^{-j}$ was played by other players according to the set $\hat{H}_t^j$.

Given the probability distribution over other players' actions, $\hat{\Pi}^{-j}$, the player j randomly plays its best reply to this distribution, $BR^j(\hat{\Pi}^{-j})$, with some exploration. If there are several equivalent best replies, the player j randomly choose one of them. Thus, Adaptive Play agent plays the pure strategies only.

Young (Young 1993) proved the convergence (with some restrictions) of Adaptive Play to an equilibrium when played in self-play for a big class of games. APQ is a simple extension of Young's Adaptive Play to the multi-state stochastic game context. For that the usual monoagent Q-learning update rule is modified to consider multiple agents as follows:

$$Q^j(s, \vec{a}) \leftarrow (1-\alpha) Q^j(s, \vec{a}) + \alpha [R^j(s, \vec{a}) + \gamma \max_{a^j \in \mathcal{A}^j} U^j(\hat{\Pi}(s'))]$$

where j is an agent, $\Pi(s')$ is $\hat{\Pi}^{-j}(s') \cup \{\hat{\pi}^j(s')\}$, $\vec{a} = \langle a^1, \dots, a^n \rangle$ is a vector of actions played by the agents in state $s \in \mathcal{S}$, $Q^j(s, \vec{a})$ is the current value for player j of playing the joint action $\vec{a}$ in state s, $R^j(s, \vec{a})$ is the immediate reward the player j receives if the joint action $\vec{a}$ is played in the state s, and $\hat{\Pi}(s')$ is a strategy profile where the player j plays its best reply, $\hat{\pi}^j(s')$, to the reduced profile of other players' strategies, $\hat{\Pi}^{-j}(s')$, calculated empirically by the player j over the past. In the repeated game context $|\mathcal{S}| = 1$.

Infinitesimal Gradient Ascent

Singh et al. (Singh, Kearns, & Mansour 1994) examined the dynamics of using the gradient ascent in two-player, two-action repeated games. The problem can be represented with two payoff matrices for the row and column players as follows:

$$R = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}, \; C = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

[3]This comparison is correct as soon as APQ and IGA agents have the same or a greater quantity of available information about the world as compared to ADL, and, thus, if they behave poorer in a game, this is not because they can access less information.

The players r and c select simultaneously an action from the set $\mathcal{A} = \{1, 2\}$, the row player r selects an action i, the column player c selects an action j and the payoffs they obtain are R_{ij} and C_{ij} respectively.

As long as this is two-action game, a mixed strategy can be represented as a single value. Let $\alpha \in [0, 1]$ be a probability the player r selects the action 1 and $1 - \alpha$ the probability to play the action 2. Let, similarly, $\beta \in [0, 1]$ and $1 - \beta$ be the probability to play the actions 1 and 2 respectively by the player c. The expected utility of playing a strategy profile Π is then calculated as follows:

$$\begin{aligned} U^r(\{\alpha, \beta\}) &= r_{11}\alpha\beta + r_{22}(1-\alpha)(1-\beta) \\ &\quad + r_{12}\alpha(1-\beta) + r_{21}(1-\alpha)\beta \end{aligned}$$

$$\begin{aligned} U^c(\{\alpha, \beta\}) &= c_{11}\alpha\beta + c_{22}(1-\alpha)(1-\beta) \\ &\quad + c_{12}\alpha(1-\beta) + c_{21}(1-\alpha)\beta \end{aligned}$$

To estimate the effect of changing its current strategy, a player can calculate a partial derivative of its expected utility with respect to its mixed strategy:

$$\frac{\partial U^r(\{\alpha, \beta\})}{\partial \alpha} = \beta u - (r_{22} - r_{12})$$

$$\frac{\partial U^c(\{\alpha, \beta\})}{\partial \beta} = \alpha u' - (c_{22} - c_{21})$$

where $u = (r_{11} + r_{22}) - (r_{21} + r_{12})$ and $u' = (c_{11} + c_{22}) - (c_{21} + c_{12})$.

At each time step IGA player adjusts its current strategy in the direction of the gradient so as to maximize its utility:

$$\alpha_{t+1} = \alpha_t + \eta \frac{\partial U^r(\{\alpha_t, \beta_t\})}{\partial \alpha}$$

$$\beta_{t+1} = \beta_t + \eta \frac{\partial U^c(\{\alpha_t, \beta_t\})}{\partial \beta}$$

where η is a step size, usually $\eta \in [0, 1]$. Obviously, the opponent's mixed strategy is supposed to be known by the players.

Singh and colleagues (Singh, Kearns, & Mansour 1994) proved the convergence of IGA to a Nash equilibrium (or to the equivalent average reward), when played in self-play, in the case of the infinitesimal step size ($\lim_{\eta \to 0}$), whence the name of the algorithm.

Adaptive Dynamics Learning

Although the adaptive dynamics show an efficient behavior in self-play, Chang and Kaelbling (Chang & Kaelbling 2001) showed that it can be exploited on the example of exploiting of PHC player (Bowling & Veloso 2002). Their PHC-Exploiter agent was able to outperform the PHC player using the knowledge of the structure of PHC adaptation algorithm. As was later shown by Tesauro (Tesauro 2004), it is possible to exploit the adaptive dynamics with a simple knowledge that the opponent is an adaptive player. His Hyper-Q learning algorithm learned explicitly the Q-values of the mixed strategy profiles. To do that, he discretized the probability space with a certain discretization size and empirically showed that Hyper-Q outperforms PHC and IGA players in RockPaperScissors game. But there are three major shortcomings that make this algorithm intractable in the real implementations. These are (1) the discretization which creates about 100 thousands of the virtual states for a game with merely two players and three actions, such as RockPaperScissors, (2) Hyper-Q agent uses a computationally hard Bayesian belief update operation at each time step and (3) the game of total observability becomes partially observable because the beliefs about other player's strategies are represented in the form of probability distribution over all possible mixed strategies.

On the contrary, we propose a much simpler adaptive dynamics learning algorithm called Adaptive Dynamics Learner (ADL) which associates a Q-value to each experienced game history H of fixed length p and a simple action $a^j \in \mathcal{A}^j$, and then learns these Q-values using a form of Q-learning. This substantially reduces the space of virtual states and actions comparatively with Hyper-Q approach. More formally, ADL player j maintains a table H of histories, considered by it as the system's states. To each history $h \in H$ it associates a Q-value of the form $Q(h, a^j) \quad \forall a^j \in \mathcal{A}^j$. Being at time step t in the state $h_t = \langle a^j_{t-p} a^{-j}_{t-p} a^j_{t-p+1} a^{-j}_{t-p+1} \dots a^j_{t-1} a^{-j}_{t-1} \rangle$ the agent j searches in H the action a^j_t which maximizes the Q-values for h_t. After that the agent j plays a^j_t with some exploration decreasing over time. Having observed the opponents' joint action and its own reward it updates its state at time $t + 1$ by concatenating its own action a^j_t and the opponents' joint-action a^{-j}_t played at time t to h_t and by eliminating two very first entries, that is,

$$h_{t+1} = \langle a^j_{t-p+1} a^{-j}_{t-p+1} a^j_{t-p+2} a^{-j}_{t-p+2} \dots a^j_t a^{-j}_t \rangle \qquad (3)$$

Finally, the player j updates the Q-value in h_t corresponding to the action a^j_t by using following Q-learning update rule:

$$\begin{aligned} Q^j(h_t, a^j_t) &\leftarrow (1-\alpha) Q^j(h_t, a^j_t) \\ &\quad + \alpha [R^j(h_t, \langle a^j_t, a^{-j}_t \rangle) + \gamma U^j(h_{t+1})] \end{aligned} \qquad (4)$$

where $U^j(h_{t+1}) = \max_{a^j \in \mathcal{A}^j} Q^j(h_{t+1}, a^j)$ and $R^j(h_t, \langle a^j_t, a^{-j}_t \rangle)$ is the reward obtained by j after playing a^j_t in the previous state h_t.

The complete formal definition of ADL algorithm is presented in Algorithm 1.

Implementation and Results

To test our algorithm we programmed two adaptive learning algorithms, IGA and APQ. In the following subsections we will provide the implementation details for each of the programmed algorithms, including ADL. Then we will present the testing results obtained in the games from GAMUT.

Adaptive Play Q-learning Agent

The AQL agent we used has the following characteristics. The length of the history, p, is 16, the size of sampling, l, is 8, the discount factor, γ, is 0.9, the learning rate, α, is proper for each state-action pair and decreases gradually using the *search-then-converge* schedule suggested by Darken

function ADL(p, t_{max})
  returns: a policy
  inputs: p, maximum history length
    t_{max}, maximum time

  Current time $t \leftarrow 0$
  Current state $h_t \leftarrow EmptySequence$
  $a_t^j \leftarrow RandomAction$
  while $t \leq t_{max}$ **do**
    Play a_t^j with some decreasing exploration
    Observe the reward R_t^j and the joint-action a_t^{-j}
    Obtain new state h_{t+1} using (3) with h_t, a_t^j and a_t^{-j}
    Find the action a_{t+1}^j maximizing Q-values in h_{t+1} and calculate the utility $U^j(h_{t+1})$
    Update $Q(h_t, a_t^j)$ using equation (4) with R_t^j and $U^j(h_{t+1})$
    Update current state $h_t \leftarrow h_{t+1}$
    Save the action to play $a_t^j \leftarrow a_{t+1}^j$
    Increment the time $t \leftarrow t + 1$
  return current policy

Algorithm 1: Adaptive Dynamics Learner (ADL) algorithm for player j.

and Moody (Darken & Moody 1991) depending on the number of updates of the respective Q-value:

$$\alpha_t(h, a^j) = \frac{\alpha_0 \tau}{\tau + n_t(h, a^j)}$$

where t is the current time step, α_0 is the initial value of the learning rate and $n_t(h, a^j)$ is the number of times that Q-value for the action a^j was updated in state h to time t. We set $\alpha_0 = 0.5$ and $\tau = 10000$. It can be shown that this schedule satisfies the conditions of convergence of Q-learning.

Infinitesimal Gradient Ascent Agent

IGA algorithm supposes omniscient knowledge by the agents of the mixed strategies of their opponents. However, neither ADL nor APQ agent explicitly play a mixed strategy, but their strategies, being pure for them are mixed for IGA player as soon as they do not act in the same internal state space, i.e., the current internal states of each agent (counterparts' actions history of APQ, concatenated joint actions of ADL and opponents' current mixed strategy of IGA) are different, though the current external state (the game played) is the same. To estimate the strategy of the opponents by IGA agent we implemented two most used techniques: (i) Adaptive Play's technique and (ii) Exponential Moving Average (EMA). We described Adaptive Play's technique in Subsection when we presented APQ algorithm. It consists in using equation (2) to estimate the probability distribution. EMA is a family of similar statistical techniques used to analyze time series data in finance and especially in technical analysis. Typically, EMA assumes that the recent observations are more informative than older ones, and, thus, as applied to our problem, given a new observed joint action $\vec{a}_t^{-j}$, IGA agent j updates a probability distribution $\vec{\Pi}_t^{-j}$, represented as a vector, as follows:

$$\vec{\Pi}_{t+1}^{-j} \leftarrow (1 - \mu)\vec{\Pi}_t^{-j} + \mu\vec{u}(\vec{a}_t^{-j})$$

where $\vec{u}(\vec{a}_t^{-j})$ is a unit vector representation of the action $\vec{a}_t^{-j}$ observed at time t and μ is a small constant, $\mu \in [0, 1]$.

We observed that in almost all runs the IGA agent that used Adaptive Play's estimation technique of probability estimation was more efficient than one that used EMA, therefore in our figures we will only present the results for Adaptive Play's technique based IGA agent.

Adaptive Dynamics Learner

The only interesting parameter of ADL is p, that is the maximal history length. What is the length that will permit to the ADL agent to learn well the dynamics of an opponent? Is there a universal value or it should be adjusted for each opponent individually? Our experiment showed that in most cases the history of the length 2 (that is, the only most recent joint action!) was sufficient to outperform the adaptive agents in adversarial games, but the value of 6 (three most recent actions) was sufficient to perform well regardless of the game played. Thus, in our experiments we set $p = 6$, but the question of how to determine the best value of p, if such exists, is still open.

Results

As mentioned above, we tested our algorithm in play versus the other two algorithms and in self-play on a set of games from GAMUT test suite (Nudelman *et al.* 2004). All the games we used were generated by GAMUT with normalization of the extreme reward values to the modulus of 1. Figure 1 represents game matrices for the three adversarial games that are of the most interest.

First, we examined the behavior of ADL against APQ player (Figure 2). To do that we observed the evolution of average Bellman error, that is the difference between two successive updates of Q-values, and the changes in the average reward per play[4] (the rewards were averaged over each $10,000$ plays). It is easy to see that the process exhibited good progress toward the convergence, as suggested by progressive reducing of average Bellman error (Figure 2(a)) and substantial positive average reward per time step (Figure 2(b)).

Further, we examined ADL versus IGA player (Figure 3). ADL showed better performance against this opponent, as is seen from the average reward diagram where the converged values are higher than the analogical values obtained in play vs. APQ agent. The average Bellman error decreased slower in that case, which is explained by the stochastic strategies used by IGA unlike APQ player, which converged directly to a policy in pure strategies.

Finally, we verified whether ADL, by being efficient against adaptive opponents, remains rational in play against

[4]Here plays and time steps are equivalent.

"MatchingPennies"

$$R, C = \begin{bmatrix} 1,-1 & -1,1 \\ -1,1 & 1,-1 \end{bmatrix}$$

"RockPaperScissors"

$$R, C = \begin{bmatrix} 0,0 & 1,-1 & -1,1 \\ -1,1 & 0,0 & 1,-1 \\ 1,-1 & -1,1 & 0,0 \end{bmatrix}$$

"ShapleysGame"

$$R, C = \begin{bmatrix} -1,-1 & 1,-1 & -1,1 \\ -1,1 & -1,-1 & 1,-1 \\ 1,-1 & -1,1 & -1,-1 \end{bmatrix}$$

Figure 1: Three adversarial games from GAMUT. R, C is the payoff matrix each entry of which contains the payoffs for the row and column players respectively.

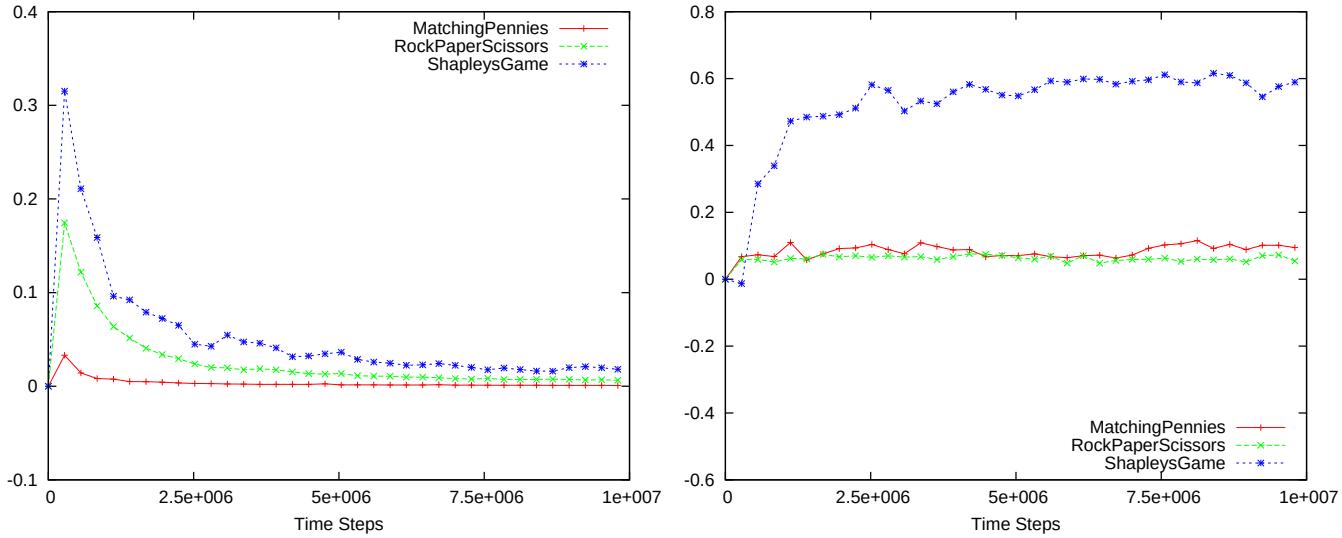

Figure 2: ADL vs. APQ in the adversarial games. Left plot shows average Bellman error of ADL, while right one reflects its average reward per time step.

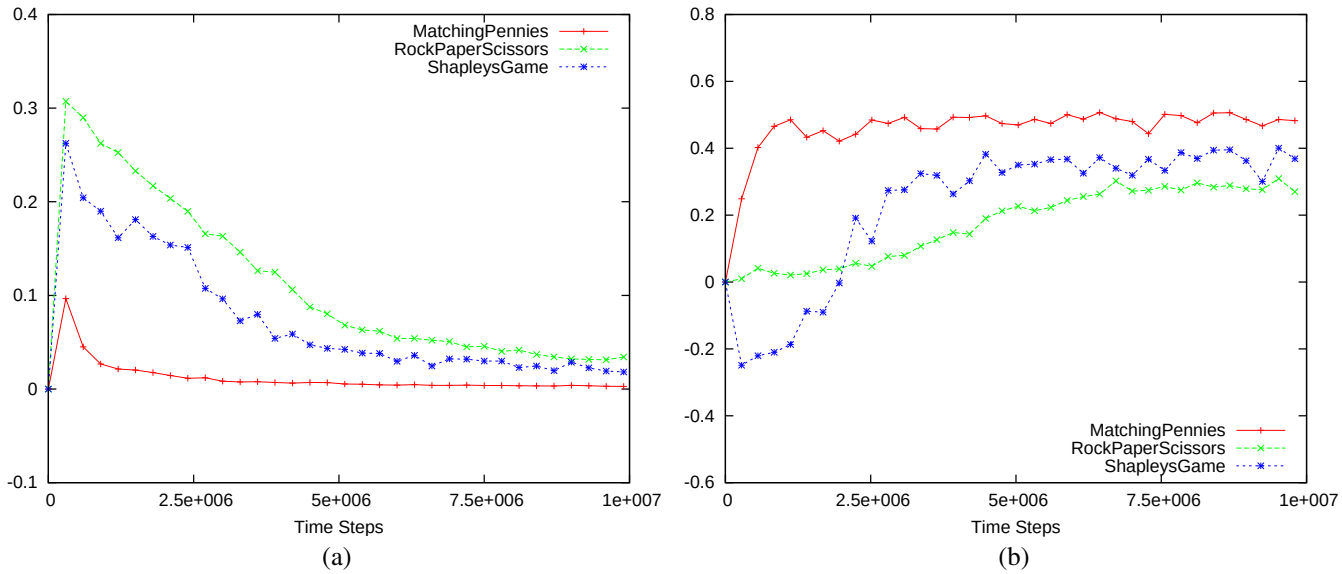

Figure 3: ADL vs. IGA in the adversarial games. Left plot represents average Bellman error of ADL, while right one reflects its average reward per time step.

itself (so called self-play) and versus other types of opponents which do not evolve in time and follow a stationary policy, such as a mixed strategy (5). As for the stationary opponents, we tested the behavior of ALD against a players which played a random (mixed) strategies on the example of RockPaperScissors game. Let the notation

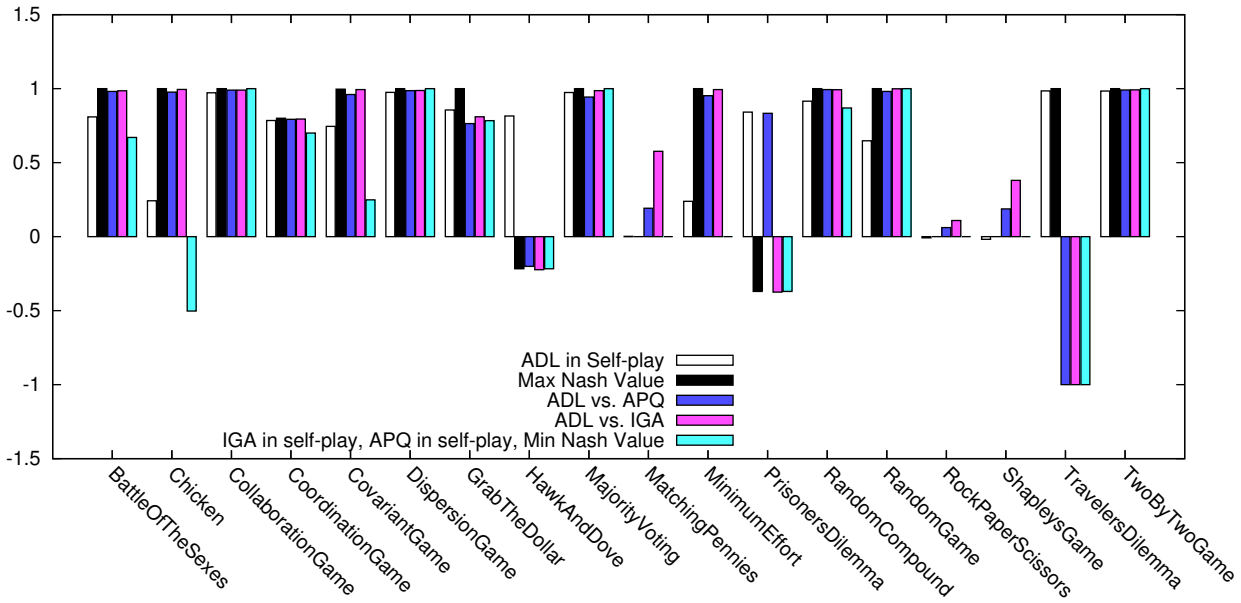

Figure 4: ADL, APQ and IGA players over the games from GAMUT.

$Random(x, y, z)$ denote a player playing a mixed strategy where there are a nonnegative probabilities of x, y and z, $x + y + z = 1$, that the actions 1, 2 and 3 respectively will be played by that player. As expected, ADL player had a positive average reward close to 0 against the $Random(0.33, 0.33, 0.34)$ opponent which played a strategy close to the Nash equilibrium $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, but it performed better as the random players were more distant from the equilibrium, thus, it converged to average reward of 0.02 against $Random(0.3, 0.4, 0.4)$, to the reward of 0.25 against $Random(0.25, 0.25, 0.5)$ and to the reward of 0.4 versus $Random(0.2, 0.2, 0.6)$. Thus, an important conclusion is that ADL algorithm remained rational in that case as well. As we already noticed above, we presented the dynamics of the learning process for three games only (RockPaperScissors, MatchingPennies and ShapleysGame) because in our opinion the adversarial case is the most interesting one. In fact, the curves of the learning dynamics in the other games are trivial: in most cases, almost from the beginning they become straight lines with some minor fluctuations. The *minimal* of the converged values of average reward of the row player over all runs for all games are presented in Figure 4. The bar graphs "ADL vs. IGA" and "ADL vs. APQ" show the reward of ADL, as the row player, in play versus these opponents. The "Max Nash" and "Min Nash" bar graphs reflect respectively the maximal and minimal average rewards per play, which the row player can gain if a Nash equilibrium is played. It is important to note that in games with a Pareto optimal strategy that is not a an equilibrium (such as PrisonersDilemma) the solution to which ADL converges in self-play is Pareto optimal, while the adaptive algorithms converge to an equilibrium which is not favorable for both players. Recall that a strategy is said to be Pareto optimal if its utility is maximal for all players. Furthermore, if there are two equilibrium outcomes, one of which is more favorable to one agent than to another one, and vice versa (such as in GrabTheDollar), then the average rewards obtained by both ADL players in self-play are a mean of these two equilibrium rewards, i.e., the welfare of both is maximized. The adaptive players are only able to find one of these equilibria, thus, as soon as an equilibrium is found, one of the agents will always have a lower utility. The dynamics of the self-play in adversarial games are not interesting enough to include in the paper; as expected, the agents, by being rational, converged to a Nash equilibrium, which brings zero average reward to both players.

Related Work

We studied how our approach to learning in MAS is situated among the other recent algorithms. There have been many new algorithms proposed in the last five years and there is still no common idea to which direction multiagent learning should progress. Three years later after the publication of critical survey of the state-of-the-art multiagent learning trend by Shoham and colleagues (Shoham, Powers, & Grenager 2003) where the authors asked about the *question* the researchers should aim at when speaking about multiagent learning, there is still no such question. We would emphasize two common directions of modern research: (1) heuristic agent composition (Powers & Shoham 2005b; 2005a) and (2) direct dynamics learning and exploiting (Chang & Kaelbling 2001; Tesauro 2004). The main and common traits of modern research are (i) a classification of algorithms relatively to a class of counterparts against which they would perform well and (ii) a definition of a lower bound of the utility which is guaranteed to be reached by

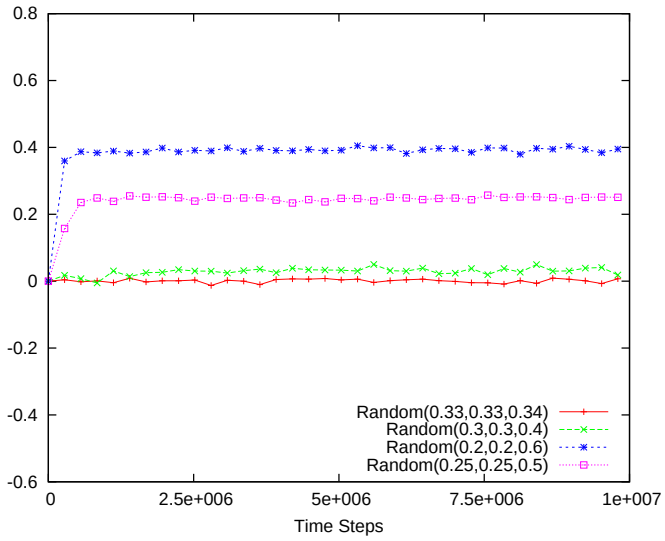

Figure 5: Average reward of ADL vs. stationary opponents playing different mixed strategies in RockPaperScissors.

that algorithm against an unknown class of opponents. Our algorithm relates to the second direction and possesses the first trait. In our opinion, however, there should be a more general approach which is an issue for further investigation.

Conclusions and Future Works

In this paper we presented an approach to learning in adaptive dynamic systems. An adaptive dynamic system may be viewed as a two-player game where a goal of a player is to maximize its own long-term utility given that the other agents (considered as a one whole agent) may follow an adaptive learning strategy. Our algorithm, called ADL (for Adaptive Dynamics Learner), by interacting with the opponent player, learns the Q-values of the states that are formed as an ordered set of joint-actions of the past plays of limited-length history. We empirically showed that our algorithm outperforms IGA (Singh, Kearns, & Mansour 1994) and APQ (Gies & Chaib-draa 2006) algorithms even if a very short history length was used to form the states. While being more general then a heuristically composed algorithms, such as Manipulator (Powers & Shoham 2005a) or MetaStrategy (Powers & Shoham 2005b), our approach is much simpler then the analogical Hyper-Q algorithm (Tesauro 2004) and, thus, possibly better scalable. It is also able to maximize the welfare of both players in self-play. Several untested adaptive dynamics opponents, such as No-Regret (Jafari *et al.* 2001) and PHC (Bowling & Veloso 2002), and certain stationary and non-stationary opponents still remain to be compared with ADL and we will give attention to that, however considerably more research is needed to develop a theoretical analysis of our approach.

References

Bowling, M., and Veloso, M. 2002. Multiagent learning using a variable learning rate. *Artificial Intelligence* 136(2):215–250.

Chang, Y., and Kaelbling, L. 2001. Playing is believing: The role of beliefs in multi-agent learning. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS'01)*.

Claus, C., and Boutilier, C. 1998. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI'98)*. Menlo Park, CA: AAAI Press.

Darken, C., and Moody, J. 1991. *Note On Learning Rate Schedule For Stochastic Optimisation*, volume 3. San Mateo, CA: Morgan Kaufmann. 832–838.

Gies, O., and Chaib-draa, B. 2006. Apprentissage de la coordination multiagent : une méthode basée sur le Q-learning par jeu adaptatif. *Revue d'Intelligence Artificielle* 20(2-3):385–412.

Jafari, A.; Greenwald, A.; Gondek, D.; and Ercal, G. 2001. On no-regret learning, fictitious play, and Nash equilibrium. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML'2001)*, 226–223. San Francisco, CA: Morgan Kaufmann.

Nudelman, E.; Wortman, J.; Leyton-Brown, K.; and Shoham, Y. 2004. Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *Proceedings of Autonomous Agents and Multiagent Systems (AAMAS'04)*.

Powers, R., and Shoham, Y. 2005a. Learning against opponents with bounded memory. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI'05)*.

Powers, R., and Shoham, Y. 2005b. New criteria and a new algorithm for learning in multi-agent systems. In Saul, L. K.; Weiss, Y.; and Bottou, L., eds., *Advances in Neural Information Processing Systems*, volume 17. MIT Press.

Shoham, Y.; Powers, R.; and Grenager, T. 2003. Multi-agent reinforcement learning: a critical survey. Technical report, Stanford University.

Singh, S.; Kearns, M.; and Mansour, Y. 1994. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI'94)*, 541–548. San Francisco, CA: Morgan Kaufman.

Tesauro, G. 2004. Extending Q-learning to general adaptive multi-agent systems. In Thrun, S.; Saul, L.; and Scholkopf, B., eds., *Advances in Neural Information Processing Systems*, volume 16. Cambridge, MA: MIT Press.

Young, H. 1993. The evolution of conventions. *Econometrica* 61(1):57–84.