J. Algorithms ••• (••••) •••–•••

Contents lists available at ScienceDirect

Journal of Algorithms
Cognition, Informatics and Logic

www.elsevier.com/locate/jalgor

Effective learning in the presence of adaptive counterparts

Andriy Burkov *, Brahim Chaib-draa

DAMAS Laboratory, Laval University, G1K 7P4, Quebec, Canada

ARTICLE INFO

Article history:
Received 30 April 2009
Available online xxxx

Keywords:
Multiagent learning
Matrix games
Adaptive learning algorithms

ABSTRACT

Adaptive learning algorithms (ALAs) is an important class of agents that learn the utilities of their strategies jointly with the maintenance of the beliefs about their counterparts' future actions. In this paper, we propose an approach of learning in the presence of adaptive counterparts. Our Q-learning based algorithm, called Adaptive Dynamics Learner (ADL), assigns Q-values to the fixed-length interaction histories. This makes it capable of exploiting the strategy update dynamics of the adaptive learners. By so doing, ADL usually obtains higher utilities than those of equilibrium solutions. We tested our algorithm on a substantial representative set of the most known and demonstrative matrix games. We observed that ADL is highly effective in the presence of such ALAs as Adaptive Play Q-learning, Infinitesimal Gradient Ascent, Policy Hill-Climbing and Fictitious Play Q-learning. Further, in self-play ADL usually converges to a Pareto efficient average utility.

1. Introduction

Classically, an approach to solve the problem of decision making in the presence of multiple agents supposed that the agents would find an interdependent solution called "equilibrium". They could achieve that by means of interactions or by using an initial knowledge of their own reward function and those of the other players. One of the most known and widely used (especially in economics) concepts of equilibrium is the *Nash equilibrium* (NE). NE is a situation, in which each agent in a multiagent system acts so as to maximize its own utility given the other players' strategies (game theorists say that it plays its best response to those strategies). In that way, a unilateral deviation of a player from the equilibrium strategy does not increase its own utility. As soon as all rational (i.e., maximizing one's own utility) players simultaneously find themselves in this situation, NE is a stable point. Nash [1] showed that such point always exists in any n-player game in which every player can choose from finitely many strategies. This latter property makes this solution concept very attractive.

There are two basic approaches to finding a NE. The first one is a game theoretical approach. It assumes the complete knowledge by all agents of the reward structure of the game underlying the decision problem. According to this approach, each agent first computes an equilibrium strategy by solving an optimisation problem, and then all agents play on this strategy. One problem arises when the game has several equilibria, and the agents have computed the different ones. In this case, when the agents execute their strategies, belonging to the equilibria computed individually, their actual joint behavior does not form an equilibrium. This usually results in a lower utility for certain or all agents.

Another problem with the classical game theoretical approach is that the agents, by computing an equilibrium, implicitly assume that all other agents are rational and, therefore, will also behave in the way prescribed by the equilibrium. However, if certain agents are not rational (for example, broken, or have an insufficient performance to compute an equilibrium in a reasonable time, or follow a predefined fixed strategy, etc.) and if there is another agent that is aware of this irrationality (or is able to deduct it) this latter agent could and would want to exploit those weak counterparts in order to augment its

* Corresponding author.
E-mail addresses: burkov@damas.ift.ulaval.ca (A. Burkov), chaib@damas.ift.ulaval.ca (B. Chaib-draa).

0196-6774/$ – see front matter
doi:10.1016/j.jalgor.2009.04.003

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

own utility. Furthermore, following an equilibrium strategy in the presence of such weak counterparts can itself be irrational because this usually does not maximize the equilibrium player's utility given the counterparts' non-equilibrium behavior.

The second approach to finding an equilibrium is the adaptive one. It assumes that the agents learn their strategies by adapting to each other in self-play, i.e., using the same learning algorithm. The adaptive approach typically assumes that the agents do not know, or know only partially, the reward structure of the game. It is also commonly supposed that after making an action the adaptive learning agents are able to observe their own reward. Certain approaches also require the observability of the actions made by the other agents [2,3]. Finally, in the less realistic settings, the strategies (i.e., the probability distributions over actions) or the rewards obtained by the other agents are also assumed to be observable [4–6].

The adaptive learners learn the utilities of their strategies jointly with the maintenance of the beliefs about their counterparts' future actions. They base their decisions on the utilities of the strategies learned as yet and the current belief about the future action of their opponents. Their strategies can be pure (a single action) or mixed (a probability distribution over actions) depending on the algorithm in question.

As yet, among the adaptive learning algorithms (ALAs), the most known and representative ones are the following: Joint-Action Learning (JAL) [2], Infinitesimal Gradient Ascent (IGA) [4], Policy Hill-Climbing (PHC) [7], Fictitious Play Q-learning (FPQ) [8], and Adaptive Play Q-learning (APQ) [3].

For certain ALAs, it was formally shown that they possess the property of convergence to a NE, or, in the limit, to an equivalent utility. Another attractive property of virtually all ALAs is that they are capable of learning a best response to the observed empirical strategy of their counterparts. This means that the above problem with the game theoretical approach (i.e., the situation when there is a weaker counterpart that does not follow an equilibrium) is here solved. Such weaker counterparts are usually exploited by the adaptive learners. However, while certain ALAs have the above two properties, the latter does not guarantee optimality regardless the opponent. When faced with a more refined counterpart, adaptive learners can be exploited [9,10].

According to Shoham et al. [11], proposing a learning strategy, which is especially effective for a fixed class of counterparts, falls under the so-called "AI agenda". The proposed in this paper algorithm that we call Adaptive Dynamics Learner (ADL) is able to learn an effective behavior in the presence of adaptive counterparts. ADL takes as its input the fixed-length history of interactions with its counterparts. Then it learns a best long-term strategy for every observed history. By so doing, ADL can exploit the dynamics of strategy updates of a number of ALAs. This yields for ADL in a higher average utility than any equilibrium strategy can provide. We tested our algorithm on a set of well known and demonstrative repeated matrix games. Our results show that ADL agent is highly effective in both self-play, and, especially in adversarial games, against the PHC, FPQ, APQ and IGA players.

The rest of the paper is organized as follows. First, we describe the four ALAs (PHC, FPQ, APQ and IGA) and introduce our ADL algorithm. Then, by comparative testing, we demonstrate that ADL can exploit the ALAs in adversarial games by remaining rational and highly effective in other games, as well as versus the stationary opponents and in self-play. Finally, we analyze the exploitability of each of the above four ALAs on an example of the repeated Matching Pennies.

2. Adaptive learning algorithms

In this section, we describe four well-known ALAs: IGA [4], PHC [7], FPQ [8] and APQ [3]. For each of them, we specify why it belongs to the class of the adaptive learning algorithms. But first, we present the notation and some important game theoretical and multiagent learning notions.

A matrix game is a tuple $(n, \mathcal{A}^{1\ldots n}, R^{1\ldots n})$, where n is the number of players, $\mathcal{A}^j$ is the set of actions of player $j = 1 \ldots n$. During one play, all players simultaneously execute their actions $a^j \in \mathcal{A}^j$. The resulting vector $\mathbf{a} = (a^1, \ldots, a^n) \in \mathbf{A} \subseteq \times_j \mathcal{A}^j$ is called joint action. The reward function $R^j : \mathbf{A} \mapsto \mathbb{R}$ defines the immediate utility of a joint action $\mathbf{a}$ for player j. A *mixed* strategy of player j is a probability distribution $\pi^j \in \Delta(\mathcal{A}^j)$, with $\pi^j_{a^j}$ denoting the probability with which player j chooses the action a^j from the set $\mathcal{A}^j$, and $\Delta(\mathcal{A}^j)$ denoting the set of all possible probability distributions over player's actions. A strategy is called *pure* if $\pi^j_{a^j} = 1$ for a certain a^j. For simplicity and with no ambiguity, we can call pure strategy an action. A *strategy profile* is a vector $\mathbf{\Pi} = (\pi^1, \ldots, \pi^n)$ of all players' strategies.

It is commonly to use the notation $-j$ to denote all players except player j. A *reduced profile for player* j, $\mathbf{\Pi}^{-j} = (\pi^1, \ldots, \pi^{j-1}, \pi^{j+1}, \ldots, \pi^n)$, is a vector containing strategies of all players except j, and $\mathbf{\Pi}^{-j}_{\mathbf{a}^{-j}}$ is the probability for players $k \neq j$ to play a joint action $\mathbf{a}^{-j} \in \mathbf{A}^{-j} \subseteq \times_k \mathcal{A}^k$. Given a player j and a reduced profile $\mathbf{\Pi}^{-j}$, a strategy $\hat{\pi}^j$ is a *best response* (BR) of player j to $\mathbf{\Pi}^{-j}$ if the expected utility of the strategy profile $(\mathbf{\Pi}^{-j}, \hat{\pi}^j)$ is maximal for player j. A best response need not be unique. There can be a, possibly infinite, set of best responses of player j to a reduced profile $\mathbf{\Pi}^{-j}$. This set is denoted as $BR^j(\mathbf{\Pi}^{-j})$. More formally, the expected utility of a strategy profile $\mathbf{\Pi} = (\mathbf{\Pi}^{-j}, \pi^j)$ for a player j is given by:

$$U^j(\mathbf{\Pi}) = \sum_{a^j \in \mathcal{A}^j} \pi^j_{a^j} \sum_{\mathbf{a}^{-j} \in \mathbf{A}^{-j}} R\left(a^j, \mathbf{a}^{-j}\right) \mathbf{\Pi}^{-j}_{\mathbf{a}^{-j}}$$

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

where $R(a^j, \mathbf{a}^{-j})$ is the reward (or, the immediate utility) player j receives if the joint action $\mathbf{a} = (a^j, \mathbf{a}^{-j})$ is played. In this case, a best response of player j to the reduced profile $\boldsymbol{\Pi}^{-j}$ is a strategy $\hat{\pi}^j$ such that $\forall \pi^j \neq \hat{\pi}^j$:

$$U^j\left(\boldsymbol{\Pi}^{-j}, \hat{\pi}^j\right) \geqslant U^j\left(\boldsymbol{\Pi}^{-j}, \pi^j\right)$$

A repeated game is a process, which consists of a certain (may be unknown, or infinite) number of repetitions of the same matrix game. From the player j's perspective, the goal of the learning in a game is to compute a strategy π^{j*} maximizing its own expected utility. This can be done by repeatedly playing with an opponent. If the desired strategy π^{j*} is in a form of a mixed or a pure strategy defined above, it is often called a memoryless (or one shot game) strategy. This is due to the fact that this strategy is an unconditioned probability distribution over agent's actions. The majority of the existing learning algorithms for repeated games, including all ALAs mentioned above, only learn the strategies of this form.

However in a repeated game, a player's strategy can be memoryful. I.e., it can be defined as a function $\pi^{j*} : \mathcal{H} \mapsto \Delta(A^j)$, where $\mathcal{H}$ is a set of all interaction histories agent j is able to memorize. The including of even short recent interaction histories into the players' strategies has been observed to be a way to improve cooperation between independent learning agents [12] and to exploit a known memoryless opponent [9]. Furthermore, the Folk theorem in the Game Theory [13] states that the utility of any Pareto efficient outcome[1] in a matrix game can be the utility of a NE in the corresponding repeated game, if the players are *sufficiently patient*. The term "sufficiently patient" refers to how long a player can remember the past interactions and how these memories influence its future decisions. Informally speaking, the strategies that achieve this Pareto efficient utility as a utility of a NE usually have the pattern that they sufficiently punish the other player for any defection from the Pareto efficient joint behavior. This encourages the other rational player not to defect for a short term gain.

The reader can remark the similarity of the principles behind the Folk theorem with the above memoryful strategies. As we will show below, our new ADL algorithm learns memoryful strategies based on the fixed-length histories of recent interactions with the opponent player. Our experimental results demonstrate that in many games, this permits obtaining a higher utility in both self-play and versus ALAs, as compared to the utility of a one shot NE.

2.1. Adaptive Play Q-learning (APQ)

The first ALA we present is a Q-learning [14] based extension of the Adaptive Play algorithm for repeated games proposed by Young [15]. Formally, each player j playing Adaptive Play saves in memory a history $H_t^j = \{\mathbf{a}_{t-p}^{-j}, \ldots, \mathbf{a}_t^{-j}\}$ of the last p joint actions played by the other players. To select an action to execute at time $t+1$, each player randomly and irrevocably samples from H_t^j a set of length l, $\hat{H}_t^j = \{\mathbf{a}_{k_1}^{-j}, \ldots, \mathbf{a}_{k_l}^{-j}\}$, of examples. Then it computes the empirical distribution $\hat{\boldsymbol{\Pi}}^{-j}$ as an approximation of the real reduced profile of strategies played by the other players, as follows:

$$\hat{\boldsymbol{\Pi}}^{-j}_{\mathbf{a}^{-j}} = \frac{C(\mathbf{a}^{-j}, \hat{H}_t^j)}{l} \tag{1}$$

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

Given the probability distribution over the other players' actions, $\hat{\boldsymbol{\Pi}}^{-j}$, the Adaptive Player j plays its best response, $BR^j(\hat{\boldsymbol{\Pi}}^{-j})$. If there are several equivalent best responses, player j randomly chooses one of them. Young [15] proved the convergence of Adaptive Play to an equilibrium in self-play for a big class of games called "weakly acyclic games". For instance, such games as coordination and common interest games belong to this class.

The APQ algorithm (for Adaptive Play Q-learning) is an extension of Young's Adaptive Play to the multi-state (or, stochastic game) context [3]. In multi-state environments, such as the stochastic games, each system's state s, belonging to the set of states $\mathcal{S}$, is considered as a distinct matrix game [16]. So, with no ambiguity, we can call "games" the states of a multi-state environment. At each game iteration, a joint action is played, the rewards are generated and then the game is changed from s to s' according to a certain transition function. To handle multiple states and players, in APQ the usual single-agent Q-learning update rule has been modified as follows:

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

where s and s' are respectively the old and the new game, $\mathbf{a}$ is the joint action played in the game s, $Q^j(s, \mathbf{a})$ is the current value of the player j's utility for playing the joint action $\mathbf{a}$ in the game s, $R^j(s, \mathbf{a})$ is the reward the player j has received after playing the joint action $\mathbf{a}$ in the game s. $\gamma \in [0, 1]$ is the so called discount factor, and it usually measures in multi-state problems how the rewards that can be obtained in the future states affect the utilities of actions in the current state [17]. In repeated games, the discount factor γ is often understood as the probability that the repeated game will continue from one iteration to the next. Notice that for simplicity of exposition, in this paper we remain in the repeated game framework, i.e., when $|\mathcal{S}| = 1$.

[1] A game outcome, or a players' joint strategy, is said to be Pareto efficient if its utility is maximal for all players.

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

The APQ algorithm (as well as the original Adaptive Play) belongs to the class of ALAs due to the fact that it always maintains an estimate of the current strategy of the other players, and its current strategy is a best response to that estimate. If the other players' strategy changes, the strategy of the Adaptive Player adapts to this change. The sensibility of APQ to an opponent's strategy change depends on the history length p. The adaptation speed in turn depends on the sampling length l.

2.2. Infinitesimal Gradient Ascent (IGA)

Singh et al. [4] examined the dynamics of using the gradient ascent in two-player, two-action repeated games. The problem can be represented with two reward matrices for the row and column players, r and c, as follows:

$$R^r = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} \qquad R^c = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

The players r and c select simultaneously an action from the set $\mathcal{A}^{r,c} = \{1, 2\}$. The row player r selects an action $p \in \mathcal{A}^r$, the column player c selects an action $q \in \mathcal{A}^c$ and the rewards they obtain are respectively R^r_{pq} and R^c_{pq}.

So far as this is a two-action game, a mixed strategy can be represented as a single value. Let $\alpha \in [0, 1]$ be a probability with which player r selects the action 1 and let $1 - \alpha$ be the probability to play the other action. In a similar way, let $\beta \in [0, 1]$ and $1 - \beta$ be the probabilities to play respectively the actions 1 and 2 by the player c. The expected utilities of playing a strategy profile $\mathbf{\Pi} = (\alpha, \beta)$ are then computed by players r and c as follows:

$$U^r(\alpha, \beta) = r_{11}\alpha\beta + r_{22}(1-\alpha)(1-\beta) + r_{12}\alpha(1-\beta) + r_{21}(1-\alpha)\beta$$
$$U^c(\alpha, \beta) = c_{11}\alpha\beta + c_{22}(1-\alpha)(1-\beta) + c_{12}\alpha(1-\beta) + c_{21}(1-\alpha)\beta$$

To estimate the effect of changing its current strategy, a player can compute 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 iteration, the Infinitesimal Gradient Ascent (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 $0 < \eta \ll 1$. Obviously, the opponent's mixed strategy is supposed to be known by the players.

The IGA algorithm belongs to the class of ALAs because the player's strategy is always updated in the direction of the gradient of its expected utility. Since this expected utility is a function of the other player's actual strategy, similarly to the Adaptive Player, the IGA player will always adapt its strategy to the strategy changes of its counterpart. The sensibility of IGA to the opponent's strategy changes is immediate, while the adaptation speed depends on the value of the step size η.

2.3. Policy Hill-Climbing (PHC)

The PHC algorithm [7] for multiagent learning is a modification of the usual Q-learning. Following a transition from a game s to a game s', the utilities of the player j's actions are updated as usually:

$$Q^j(s, a^j) \leftarrow (1-\alpha) Q^j(s, a^j) + \alpha \left[R^j(s, a^j) + \gamma \max_{a^j} Q(s', a^j) \right]$$

where a^j is the action executed by player j in the game s.

However, unlike the Q-learning, which only learns pure strategies,[2] PHC can learn mixed strategies. The strategy is improved by increasing the probability of the highest valued action using the small step δ, called learning rate:

$$\pi^j_{a^j}(s) \leftarrow \pi^j_{a^j}(s) + \Delta_{sa^j}$$

where

[2] In the single-agent stationary environments, there always exists an optimal pure strategy.

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

$$\Delta_{sa^j} = \begin{cases} -\delta_{sa^j} & \text{if } a^j \neq \operatorname{argmax}_{a'^j} Q(s, a'^j) \\ \sum_{a'^j \neq a^j} \delta_{sa'^j} & \text{otherwise} \end{cases}$$

$$\delta_{sa^j} = \min\left(\pi^j_{a^j}(s), \frac{\delta}{|\mathcal{A}^j| - 1}\right)$$

while constrained to a legal probability distribution.

While PHC does not possess proved convergence properties, its learning dynamics has been empirically observed to be very similar to that of IGA [7]. To relate PHC to the class of ALAs, notice that PHC, albeit implicitly, still maintains the beliefs about its opponent strategy. These beliefs are "hidden" in the Q-values that are assigned to the pairs (*game*, *player's action*). In fact, the PHC player's Q-value of the game-action pair (s, a^j) reflects the expected utility of the action a^j given the counterparts' unobservable joint strategy in the game s. If the counterparts' strategies change, the strategy of the player j will change accordingly. The sensibility of PHC to the opponent strategy changes depends on the value of the learning rate α. The adaptation speed, in turn, is defined by the value of the second learning rate, δ. If the learning rate α is such that it decreases as $t \to \infty$,[3] PHC will gradually loose its sensibility to the counterparts' strategy changes. However, as we will demonstrate below, even when α becomes very small, our ADL algorithm is still able to play effectively against PHC.

2.4. Fictitious Plays Q-learning (FPQ)

Similarly to APQ, the FPQ algorithm maintains the empiric beliefs about the reduced profile of the other players' strategies. However, instead of keeping in memory separately a fixed length history of recent interactions and the Q-values for every joint action (as the APQ algorithm does) in FPQ these data are mixed in a way similar to PHC. I.e., in FPQ the Q-values of each game are assigned to the player's own actions and not to the joint actions. Let, after a transition from a previous game s to the next game s', $\mathbf{a}^{-j}$ be the joint action played by the opponents. The FPQ player j then simultaneously updates all its Q-values of the game s as follows,

$$Q^j(s, a^j) \leftarrow Q^j(s, a^j) + R^j(s, a^j, \mathbf{a}^{-j}) + \gamma \mathop{\mathrm{E}}_{s' \in \mathcal{S}}\left[\max_{a'^j \in \mathcal{A}^j} \frac{Q(s', a'^j)}{m(s')}\right], \quad \forall a^j \in \mathcal{A}^j$$

where $m(s)$ is the number of times the game s has been played. The expectation over the future games is computed using the probabilities of transition from the game s to all other games in $\mathcal{S}$. At each iteration, FPQ deterministically selects the action to play by searching in its table of Q-values and by picking the action corresponding to the highest Q-value. It randomizes between the actions with equal Q-values.

In its ALA properties, FPQ is similar to APQ, but with one important difference. The sensibility of FPQ to the opponent's strategy change is not a constant value. More precisely, it decreases as the number of games played grows. This is due to the fact that the Q-values of FPQ are constantly growing. In this case, when the absolute difference between the Q-values of different actions becomes large, the short-term changes of the opponents' strategy do not affect the strategy of the FPQ player. Therefore, at first glance it should become more and more difficult for ADL to exploit FPQ. However, we will show below that ADL remains capable of exploiting FPQ even after a significantly large number of games played.

3. Adaptive Dynamics Learner (ADL)

Although the ALAs usually demonstrate an effective behavior in self-play, Chang and Kaelbling [9] showed that these algorithms can be exploited by a more refined opponent. For instance, their algorithm, called PHC-Exploiter, outperforms PHC in adversarial games by observing its behavior.

As was later shown by Tesauro [10], it is possible to exploit the adaptive dynamics with a simple knowledge that the opponent belongs to the class of ALAs. His Hyper-Q learning algorithm explicitly learns the Q-values of mixed strategy profiles (to do that, he discretized the probability space with a certain discretization size). Tesauro [10] empirically demonstrated that Hyper-Q outperforms PHC and IGA players in the Rock, Paper, Scissors game. However, there are three major limitations making this algorithm impractical for real implementations. These are (1) discretization, which creates thousands of internal states in the games with merely two players and three actions, such as Rock, Paper, Scissors; (2) at each iteration, Hyper-Q agent uses a computationally hard Bayesian belief update operation; and (3) the game of total observability becomes partially observable because the beliefs about other player's strategies are represented in a form of probability distribution over all possible mixed strategies.

On the contrary, our Adaptive Dynamics Learner (ADL) algorithm is much simpler in terms of the amount of computation required per iteration. ADL assigns a distinct Q-value to each experienced game history H of a fixed length p and to each action from its action set. Then it learns these Q-values by using a form of Q-learning. This substantially reduces the space of internal states and possible strategies comparatively to the Hyper-Q approach.

More formally, the ADL player j maintains a table H^j of histories, considered as its internal states. To each history $h^j \in H^j$, it assigns a Q-value of the form $Q^j(h^j, a^j)$, $\forall a^j \in \mathcal{A}^j$. Being at iteration t in the state $h^j_t =$

[3] This is a necessary convergence condition of the Q-learning.

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

Require Maximum history length p, Maximum time t_{max}
1: Current time $t \leftarrow 0$
2: Current state $h_t^j \leftarrow EmptySequence$
3: $a_t^j \leftarrow RandomAction$
4: **while** $t \leqslant t_{max}$
5: Play a_t^j with some decreasing exploration.
6: Observe the reward R_t^j and the joint-action $\mathbf{a}_t^{-j}$.
7: Obtain new state h_{t+1}^j using (2) with h_t^j, a_t^j and $\mathbf{a}_t^{-j}$.
8: Find the action a_{t+1}^j maximizing Q-values in h_{t+1}^j and compute the utility $U^j(h_{t+1}^j)$.
9: Update $Q^j(h_t^j, a_t^j)$ using Eq. (3) with R_t^j and $U^j(h_{t+1}^j)$.
10: Update current state $h_t^j \leftarrow h_{t+1}^j$.
11: Save the action to play $a_t^j \leftarrow a_{t+1}^j$.
12: Increment the time $t \leftarrow t + 1$.
13: **end while**
14: **return**

Algorithm 1. Adaptive Dynamics Learner for player j.

$(a_{t-p}^j \mathbf{a}_{t-p}^{-j} a_{t-p+1}^j \mathbf{a}_{t-p+1}^{-j} \dots a_{t-1}^j \mathbf{a}_{t-1}^{-j})$, agent j searches in the table of Q-values the action a_t^j that maximizes the ADL's utility in h_t^j. After that, agent j plays a_t^j with some probability of exploration decreasing over time. Having observed the opponents' joint action and its own reward, it generates its new internal state at iteration $t+1$ by concatenating its own action a_t^j and the opponents' joint-action $\mathbf{a}_t^{-j}$ played at time t to h_t^j and by withdrawing two very first entries. That is,

$$h_{t+1}^j = \left(a_{t-p+1}^j \mathbf{a}_{t-p+1}^{-j} a_{t-p+2}^j \mathbf{a}_{t-p+2}^{-j} \dots a_t^j \mathbf{a}_t^{-j}\right) \tag{2}$$

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

$$Q^j\left(h_t^j, a_t^j\right) \leftarrow (1-\alpha) Q^j\left(h_t^j, a_t^j\right) + \alpha\left[R^j\left(h_t^j, a_t^j, \mathbf{a}_t^{-j}\right) + \gamma U^j\left(h_{t+1}^j\right)\right]$$

where $U^j(h_t^j) = \max_{a^j \in \mathcal{A}^j} Q^j(h_t^j, a^j)$ and $R^j(\cdot)$ is the reward obtained by j in the previous state. The ADL algorithm is formally defined in Algorithm 1.

4. Implementation and results

To validate our approach, we programmed four ALAs: IGA, PHC, FPQ and APQ. The following subsections provide the implementation details for all programmed algorithms, including ADL. Then we present the experimental results obtained in the games from GAMUT, a game theoretical test suite [18].

4.1. The APQ agent

The APQ agent we programmed has the following characteristics. The length of the history, p, is 16, the size of sampling, l, is 8, the discount factor, γ, is 0.99. The choice of particular values for p and l was dictated by the fact that the minimal suitable values chosen in non-adversarial games[4] made APQ too easily exploitable by ADL in adversarial games; furthermore, there is no formal way to choose the best values for these parameters in adversarial games. However, we observed that increasing the values of these parameters lowers the adaptation speed of APQ and, consequently, APQ becomes less exploitable. The learning rate α decreases gradually using the search-then-converge schedule [19] depending on the number of updates of the respective Q-value:

$$\alpha_t(s,a) = \frac{\alpha_0 \tau}{\tau + n_t(s,a)}$$

where t is the current iteration, α_0 is the initial value of the learning rate and $n_t(s,a)$ is the number of times that the Q-value for the action a has been updated in the state s to time t. We set $\alpha_0 = 0.5$ and $\tau = 10,000$.

4.2. The IGA agent

IGA supposes an omniscient knowledge by the agent of the mixed strategy currently followed by its opponent. However, ADL does not explicitly play mixed strategies. On the other hand, its strategies, being pure to it, appear to be mixed for the IGA player because the two algorithms do not act in the same internal state space. Indeed, the current internal states of these algorithms (i.e., concatenated joint actions for ADL, and opponent's current mixed strategy for IGA) are different,

[4] The choice of these values is game dependent, see [15] for details.

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

$$R^{(1,2)} = \begin{pmatrix} (-1,-1) & (1,-1) & (-1,1) \\ (-1,1) & (-1,-1) & (1,-1) \\ (1,-1) & (-1,1) & (-1,-1) \end{pmatrix}$$
Shapley's Game

$$R^{(1,2)} = \begin{pmatrix} (0,0) & (1,-1) & (-1,1) \\ (-1,1) & (0,0) & (1,-1) \\ (1,-1) & (-1,1) & (0,0) \end{pmatrix}$$
Rock, Paper, Scissors

$$R^{(1,2)} = \begin{pmatrix} (1,-1) & (-1,1) \\ (-1,1) & (1,-1) \end{pmatrix}$$
Matching Pennies

$$R^{(1,2)} = \begin{pmatrix} (3,3) & (0,5) \\ (5,0) & (2,2) \end{pmatrix}$$
Prisoner's Dilemma

Fig. 1. Four game matrices.

though from the standpoint of an external observer, the state of the system (i.e., the game played) is the same for both players. To make the IGA agent capable of estimating the strategy of its opponent, we implemented two different techniques: (i) Adaptive Play's technique and (ii) Exponential Moving Average (EMA). We described the opponent's strategy estimation technique, used by Adaptive Play, in Section 2.1. It consists of using Eq. (1) to estimate the probability distribution.

EMA is a family of statistical techniques used to analyze time series data in finance and technical analysis. Typically, EMA assumes that the recent observations are more informative than the older ones. As applied to our problem, given a new observed opponents' joint action $\mathbf{a}_t^{-j}$, an EMA based agent j updates the estimation $\hat{\mathbf{\Pi}}_t^{-j}$ as follows:

$$\hat{\mathbf{\Pi}}_{t+1}^{-j} \leftarrow (1-\mu)\hat{\mathbf{\Pi}}_t^{-j} + \mu \mathbf{u}\left(\mathbf{a}_t^{-j}\right)$$

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

In our experiments, we observed that the IGA agent using the Adaptive Play's estimation technique was usually more successful than the one using EMA. Therefore in our figures, we only present the results obtained by the IGA agent using the Adaptive Play's estimation technique. The strategy update step size of IGA, η, has been set to be as small as 0.05. As we observed in our experiments, lower values of this parameter (0.01, for example) decrease the adaptation speed of the IGA agent to the strategy changes of its opponent. However, IGA still remains exploitable by ADL in adversarial games (although after a longer initial learning period).

4.3. The FPQ and PHC agents

The only important input parameter for both FPQ and PHC algorithms is the discount factor γ. For uniformity with the other algorithms, we set it to be equal to 0.99. In PHC, the same search-then-converge schedule was used for decaying the learning rate α.

4.4. The ADL agent

The only important parameter of ADL is p, the length of the history defining its internal states. Is there a value of p that would permit ADL agent to learn well the dynamics of an arbitrary opponent? Is there a certain universal value or it should be individually adjusted to each opponent? Our experiments showed that most of the time the history of length 2 (i.e., the only most recent joint action) was sufficient to outperform the ALAs in adversarial games. On the other hand, 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 how to efficiently determine the best value for p remains an important open question.

In our preliminary experiments with ADL, we observed that the standard exploration techniques that are typically used in the reinforcement learning, such as GLIE [20], while being effective for ADL, were not efficient enough in terms of the learning speed. This can be explained by the fact that the internal states of ADL are not all interconnected. Indeed, by making a random exploratory action, the ADL agent only changes a small fraction of the variables defining its internal state. For example, consider a game in which each player has $\mathcal{A} = \{a, b\}$ as its action set, and $p = 6$. Let at a certain iteration t, the current internal state of ADL be (aa, aa, aa). If ADL decides to play an exploratory action at iteration t, it can only transit to one of the following four states: (aa, aa, aa), (aa, aa, ab), (aa, aa, ba) and (aa, aa, bb) while there exists 64 possible states. When the exploration probability becomes low, ADL risks staying in a limited subset of internal states for a long time. To overcome this, in our implementation we adopted the following modified exploration strategy. Instead of deciding whether to explore or to exploit at every iteration, our ADL agent decides once for p iterations in advance. Thus, if at iteration t ADL decides to make an exploratory action, it then makes random exploratory actions at each of the iterations $t+1, t+2, \ldots, t+p-1$. In practice, this considerably improves the learning speed.

4.5. Results

As mentioned above, we tested our algorithm in play versus the other four algorithms and in self-play on a set of games from GAMUT. First, we examined the case of ADL versus APQ in three adversarial games presented in Fig. 1: "Shapley's Game", "Rock, Paper, Scissors" and "Matching Pennies". In particular, we observed the evolution of average Bellman error, which is the absolute difference between two successive updates of Q-values, and the changes in the average reward of ADL per iteration (Fig. 2). During one repeated game run, the rewards were averaged after each 10,000 iterations. The final

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

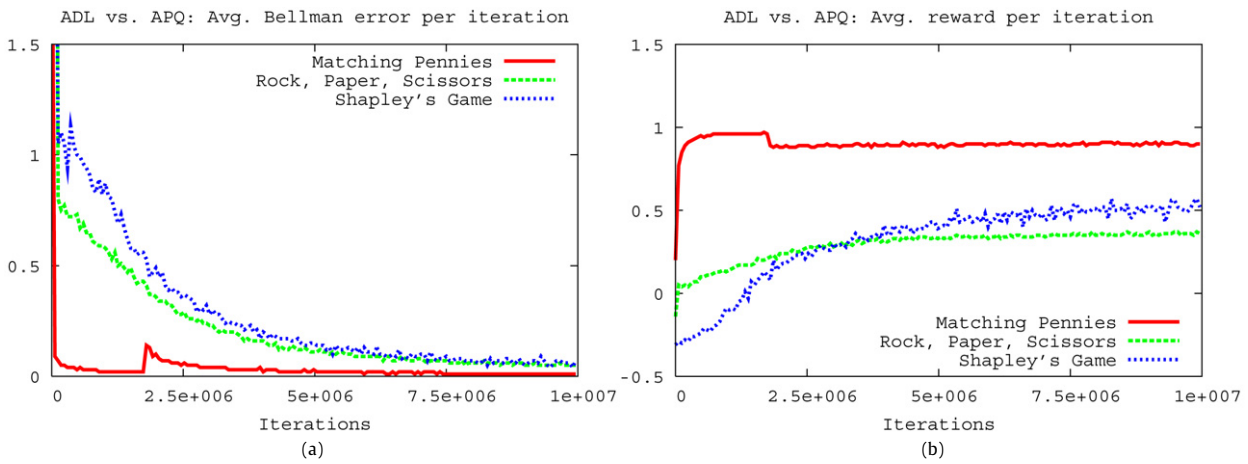

Fig. 2. ADL vs. APQ in the adversarial games.

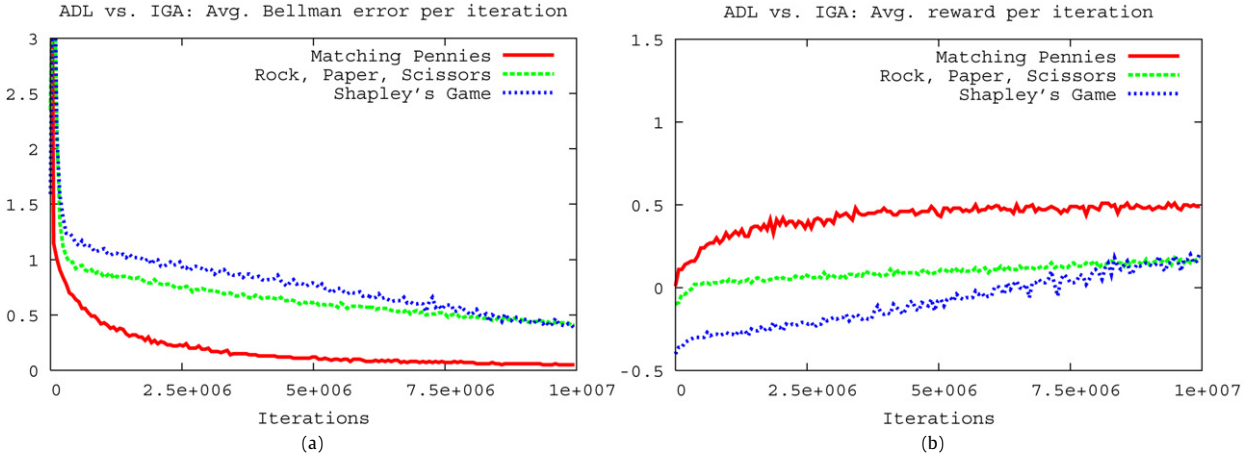

Fig. 3. ADL vs. IGA in the adversarial games.

curves represent the data averaged after ten independent runs. One can see that the process exhibits good progress towards the convergence, as suggested by a progressive reducing of the average Bellman error (Fig. 2a) and a substantial positive average reward of ADL per iteration (Fig. 2b).

Further, we examined ADL versus IGA in the same three games. ADL showed poorer performance against this opponent, as is seen from the average reward curves (Fig. 3b) where the converged values are lower than the analogical values obtained versus APQ. The average Bellman error decreased slower in that case (Fig. 3a). This can be explained by the fact that, unlike APQ, IGA is capable of playing mixed strategies.

The corresponding curves for FPQ and PHC algorithms are presented in Figs. 4 and 5. Surprisingly, PHC performed well enough despite its reduced knowledge about the world (recall that PHC does not perceive the opponent's actions). Similarly to APQ, playing only pure strategies, FPQ was observed to be more exploitable than PHC and IGA.

Finally, we verified whether the ADL's behavior remains rational versus other classes of opponents. In particular, on the Rock, Paper, Scissors example, we examined ADL's behavior in self-play and versus the opponents that do not evolve in time (e.g., follow a certain stationary mixed strategy).

Let $Random(x, y, z)$ denote a player playing a mixed strategy assigning the probabilities of x, y and z to the actions *Rock*, *Paper* and *Scissors*. Obviously, ADL player has a positive average reward close to zero against the $Random(0.33, 0.33, 0.34)$ opponent, which plays a strategy close to the NE $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. (In fact, any ADL's strategy would guarantee it a value close to zero if its opponent plays a NE.) However, ADL starts performing better as the random player's strategy becomes more distant from the equilibrium. In particular, it converged to the average reward of 0.02 against $Random(0.3, 0.3, 0.4)$, to the average reward of 0.25 against $Random(0.25, 0.25, 0.5)$ and to the average reward of 0.4 versus $Random(0.2, 0.2, 0.6)$. Thus, an important conclusion is that the ADL's behavior is rational in this case.

The result of ADL's play in "Matching Pennies" versus another ADL player with different values of the history-length p, is presented in Fig. 6. The curve reflects the average reward per iteration (after convergence) of player 1, having $p^1 = 6$, as

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

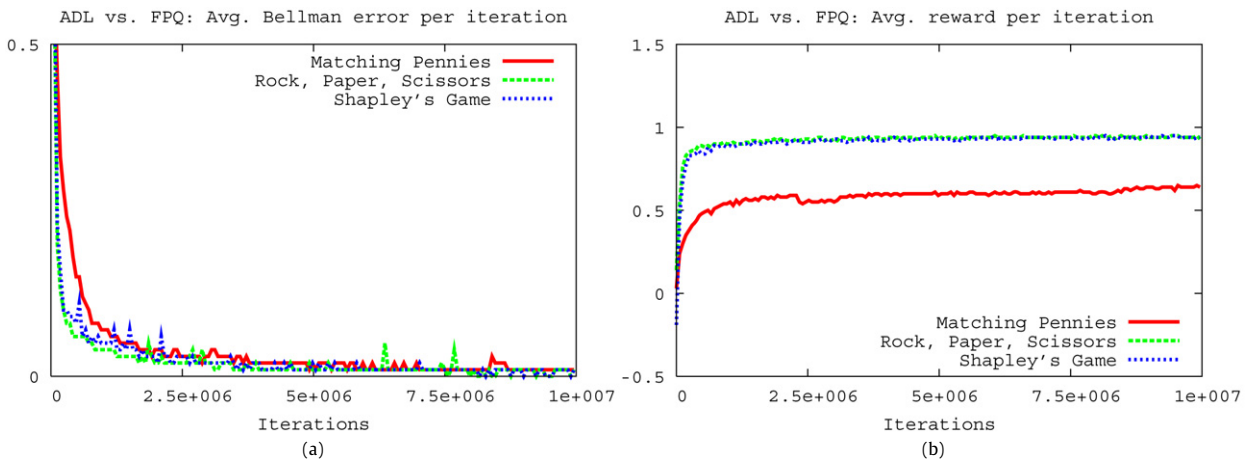

Fig. 4. ADL vs. FPQ in the adversarial games.

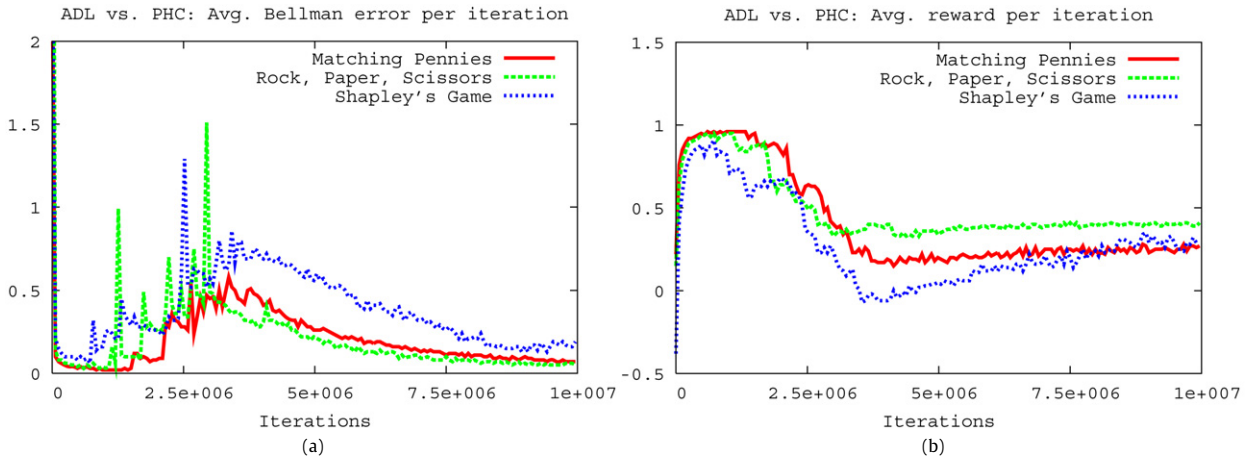

Fig. 5. ADL vs. PHC in the adversarial games.

a function of the history-length p^2 of player 2. As expected, when $p^1 = p^2$ both players have the same utility of a NE, i.e., a zero average reward per iteration. In the other cases, the player having a larger value of p performs better.

Another notable observation was made in the matrix games having a Pareto efficient outcome, which is not an equilibrium, such as the example of Prisoner's Dilemma given in Fig. 1. In such games, ADL in self-play often converges to an average reward close to the value of a Pareto efficient outcome (Fig. 7). On the contrary, the self-played ALAs always converge to an average utility of an equilibrium, which can not always be favorable for both players. Furthermore, in the games in which there are two outcomes, one of which is more favorable for the first player while the other one yields in a symmetrically higher utility for the second player, the average reward obtained by ADL in self-play is often a mean of these two outcomes. This means that the welfare of both players is maximized. The adaptive learners usually are only able to converge to either one of these outcomes.

In our comparative analysis, we only presented the dynamics of the learning process for the three adversarial games. This is due to the fact that to demonstrate the exploitability of ALAs, the adversarial case is the most interesting one. Indeed, the learning dynamics in most of the other games we programmed were trivial: almost from the beginning, the curves became 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 Table 1. The columns "ADL vs. IGA" and "ADL vs. APQ" show the reward of ADL in play versus these opponents. The "Max Nash" and "Min Nash" columns reflect respectively the maximal and minimal average rewards per iteration, which the row player can gain if a NE is played. Recall that in self-play, ALAs can only converge to the average utility of a NE.

5. Discussion

In this section, we present a first insight into the roots of such a high performance of ADL in the presence of the adaptive opponents. To do this, for each of the four ALAs, we discuss the possibilities to be exploited by a more refined opponent

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

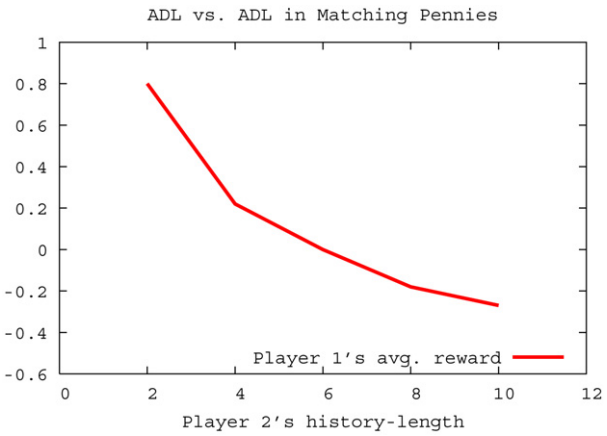

Fig. 6. Average reward of the ADL player 1, having $p = 6$, playing Matching Pennies versus the ADL player 2, having p respectively equal to 2, 4, 6, 8 and 10.

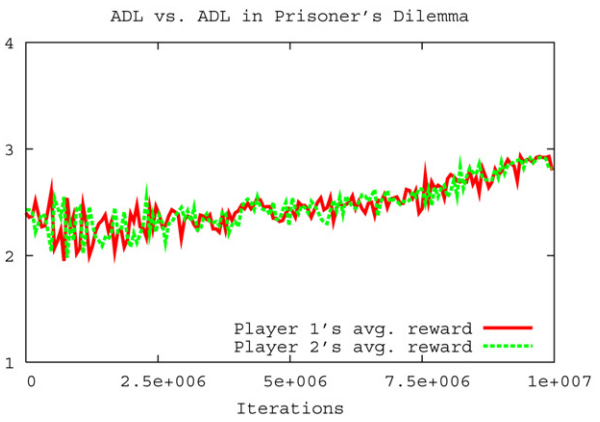

Fig. 7. ADL in the repeated Prisoner's Dilemma self-play.

Table 1
ADL, APQ and IGA players over the games from GAMUT.

	ADL vs. ADL	Max Nash	Min Nash	ADL vs. APQ	ADL vs. IGA
Battle of the Sexes	**0.81**	1.00	0.67	0.98	0.99
Chicken	**0.24**	1.00	−0.50	0.98	0.99
Collaboration game	**0.97**	1.00	1.00	0.99	0.99
Coordination game	**0.78**	0.80	0.70	0.79	0.79
Covariant game	**0.74**	1.00	0.24	0.96	0.99
Dispersion game	**0.97**	1.00	1.00	0.99	0.99
Grab the Dollar	**0.86**	1.00	0.78	0.76	0.81
Hawk and Dove	**0.81**	−0.22	−0.22	−0.20	−0.22
Majority Voting	**0.97**	1.00	−1.00	0.94	0.99
Matching Pennies	**0.00**	0.00	0.00	0.90	0.50
Minimum Effort	**0.24**	1.00	0.00	0.95	0.99
Prisoner's Dilemma	**2.87**	2.00	2.00	2.88	2.00
Rock, Paper, Scissors	**−0.01**	0.00	0.00	0.36	0.18
Shapley's game	**−0.02**	0.00	0.00	0.52	0.17
Traveler's Dilemma	**0.99**	1.00	−1.00	−1.00	−1.00
Two-By-Two game	**0.98**	1.00	1.00	0.99	0.99

on an example of the repeated Matching Pennies (RMP). An example of the RMP game matrix is presented in Fig. 1. In this game, the set of actions of both players is $\{\text{Heads}, \text{Tails}\} = \{\text{H}, \text{T}\}$.

5.1. On the exploitability of FPQ

The easiest way to demonstrate the exploitability of ALAs is to observe the behavior of FPQ in the presence of an exploiter. For simplicity, assume the discount factor γ of FPQ to be 0. Let suppose that at a certain game iteration t, the Q-values which FPQ assigns to Heads and Tails are the same and equal to Q. Let suppose that the exploiter executes the following periodic strategy: H, T, H, T, H, T, ... The corresponding Q-values of FPQ (after an update) will be $(Q-1, Q+1), (Q, Q), (Q-1, Q+1), (Q, Q), (Q-1, Q+1), (Q, Q), \ldots$. The actions executed by FPQ at the corresponding iterations will be[5] (H or T), T, (H or T), T, (H or T), T, ... The expected cumulative reward of the exploiter after three periods is therefore $0+1+0+1+0+1=3$. So, when the discount factor of FPQ is 0 and the exploiter executes the above periodic strategy, the latter has a positive expected average utility of $1/2$ per iteration. As we observed in our experiments, non-zero values of the discount factor make FPQ even more exploitable since higher values of the discount factor result in an additional inertia in the Q-value updates. For instance, the periodic strategy to which ADL converges in practice has longer periods of playing the same action. For example, a typical strategy to which ADL with $p = 6$ converges versus FPQ with $\gamma = 0.99$ is the following: H, H, H, H, H, H, T, T, T, T, T, T, H, H, H, H, H, H, T, T, T, T, T, T, ...

Notice that by only observing a recent history of joint actions, the exploiter can distinguish with confidence the situation when the Q-values of FPQ for both Heads and Tails become the same. If in the current recent history of joint actions the very recent opponent's action differs from the previous one, then its Q-values either have become equal (the opponent is

[5] Recall that the FPQ's strategy is a deterministic function of its Q-values.

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

(1) Choose an ϵ such that $0 < \epsilon \ll 1$.
(2) Play Heads with probability $\frac{1}{2} + \epsilon$ until the opponent starts playing Tails with probability 1; then play Tails with probability 1 until the opponent's strategy approaches a uniform distribution.
(3) Play Tails with probability $\frac{1}{2} + \epsilon$ until the opponent starts playing Heads with probability 1; then play Heads with probability 1 until the opponent's strategy approaches a uniform distribution.
(4) Go to step (2).

Algorithm 2. A procedure to exploit PHC.

randomizing), or one Q-value has just become higher than the other one. Obviously, in all other situations the exploiter has to play the best response to the observed opponent's play. This will yield in a non-negative average utility for the exploiter and will also shift the opponent's Q-values in the direction of equality. As we observed in our experiments, the ADL's Q-learning with a sufficient exploration and a sufficiently large discount factor, permits learning of a successful exploitive strategy.

5.2. On the exploitability of PHC

PHC players have already been shown to be exploitable by a more informed or a more refined opponent [9,10]. Without loss of generality, let suppose that ADL is the first player. One way to exploit PHC in the RMP is to follow a procedure similar to the one presented in Algorithm 2. The steps (2) and (3) of the above procedure assume that the PHC player will eventually change its strategy to deterministically play one action. This is assured by the fact that PHC updates its strategy so as to increase the probability of playing the action having the highest Q-value. For example, in the step (2), as long as the exploiter is playing the strategy, in which Heads has a slightly higher probability, the PHC's Q-value of Tails will eventually become slightly higher. Then it will adjust its strategy so as to give a higher probability to Tails. The speed with which PHC approaches a deterministic strategy is constant and depends on the value of the learning rate δ.

To determine whether the opponent follows a deterministic strategy, the exploiter can observe the opponent's recent history of play. For instance, if in all joint actions from the recent history the opponent repeats the same action, then the latter is more likely to be following a (close to) deterministic strategy. The complete recent history (including the actions of both players) and the Bellman principle would permit the exploiter to choose the right action in every history by implicitly determining whether the step (2) or (3) of the above procedure is being executed (assuming a sufficient exploration and a high value of the exploiter's discount factor).

5.3. On the exploitability of the other ALAs

The exploitability of APQ has the same roots as that of FPQ. In fact, the basic Fictitious Play and Adaptive Play algorithms are very similar. Their only difference that the latter bases its decisions on a history of a limited size while the former–on the whole interaction history. However, from the standpoint of an exploiter they do not differ. In fact, in practice we observed that APQ behaves similarly to FPQ in the presence of the exploiter playing the above periodic strategy.

In turn, IGA has the same exploitability property as PHC. Indeed, the satisfaction of conditions of the steps (2) and (3) of the procedure of Algorithm 2 (eventual convergence of the ALA to a deterministic strategy) is assured by the fact that after every repeated game play, IGA updates its strategy in the direction of the gradient of its expected utility. This gradient, in turn, as well as the expected utility, depends on the opponent's actual strategy. As long as the exploiter follows as prescribed by the steps (2) or (3) of the above procedure, the IGA's utility gradient will direct the strategy updates so as to eventually play either Tails or Heads with probability 1. Recall, that the speed, with which IGA adapts its strategy, remains constant all along the learning. This speed is completely defined by the value of the step size η.

6. Related work

As it was already said, there have been several successful attempts of exploiting a weak learning agent in repeated adversarial games. Chang and Kaelbling [9] proposed an approach for exploiting a PHC player. Their PHC-Exploiter algorithm is capable of estimating the learning rate δ of its PHC adversary. Then, using this estimate, it is capable of computing a strategy that can "fool" the PHC player in adversarial games by playing in a way similar to the procedure of Algorithm 2.

Tesauro [10] proposed a Q-learning based approach for exploiting weak gradient ascent players. His Hyper-Q agent learns the Q-values of the pairs (*player's own strategy, opponent's estimated strategy*). This approach was successfully tested against IGA and PHC in Rock, Paper, Scissors. However, the scalability of Hyper-Q is an important issue because, in general case, the table of Q-values of Hyper-Q should contain an infinity of elements.

In a recent work, Chakraborty and Sen [21] proposed modeling the learning environments induced by gradient ascent learners (WoLF-IGA, WoLF-PHC [7] and ReDVaLeR [6]) as MDPs. In the presence of a gradient ascent adversary, the learning algorithm, called MB-AIM-FSI, first creates a set of hypotheses about the model of the learning environment that can be induced by the learning adversary. Then the algorithm performs a sequence of exploratory actions and collects the observations of its own rewards and the opponent's actions. Finally, using the maximum likelihood principle, MB-AIM-FSI chooses

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003

the best model that fits the observed data, and solves its corresponding MDP. This process is iteratively repeated so that the model is gradually improved.

Powers and Shoham [22,23] proposed heuristically composed algorithms capable of effectively playing repeated games against the game playing algorithms from a target class of opponents. In short, their algorithms first test whether the opponent player's algorithm belongs to the target class. If yes, it chooses the best predefined behavior. Otherwise, it either plays a best response to the observed opponent's behavior or the minimax strategy.

7. Conclusions

In this paper, we presented an approach to the learning in adaptive dynamics systems. Such a system may be viewed as a two-player repeated matrix game where the learner's goal is to maximize its own long-term average utility given that the other agents (considered as one whole agent) may be following an adaptive learning algorithm (ALA). Our new Adaptive Dynamics Learner (ADL) algorithm, by interacting with the opponent player, learns the Q-values of its own internal states. These states are obtained as a concatenation of joint actions played in the recent limited-length history.

We empirically demonstrated that in adversarial games, our algorithm outperforms such ALAs as IGA [4], APQ [3], FPQ [8] and PHC [7] even if a very short history length is used to form the ADL's internal states. In repeated Prisoner's Dilemma self-play, ADL demonstrates a steady convergence to the average utility of a cooperative outcome. In other non-adversarial game self-plays, when possible, ADL converges to an average utility maximizing the welfare of both players. In the same games, ALAs typically converge to a less attractive average utility of a Nash equilibrium.

Finally, on an example of the repeated Matching Pennies, we gave a first theoretical insight into the reasons of exploitability of each of the four ALAs. We explained how each algorithm can be exploited by a more refined opponent and gave several examples of a successful exploitation strategy. Our experiments showed that our ADL algorithm is capable of learning the strategies that resemble those example strategies.

Acknowledgments

This research was supported by the Natural Sciences and Engineering Council of Canada and the Fonds Québécois de la Recherche sur la Nature et les Technologies. We would also like to thank the anonymous reviewers for their helpful comments and suggestions.

References

[1] J. Nash, Equilibrium points in n-person games, Proc. Natl. Acad. Sci. USA 36 (1) (1950).
[2] C. Claus, C. Boutilier, The dynamics of reinforcement learning in cooperative multiagent systems, in: Proceedings of the Fifteenth National Conference on Artificial Intelligence, AAAI'98, AAAI Press, Menlo Park, CA, 1998, pp. 746–752.
[3] O. Gies, B. Chaib-draa, Apprentissage de la coordination multiagent: une méthode basée sur le Q-learning par jeu adaptatif, Rev. Intelligence Artif. 20 (2–3) (2006) 385–412.
[4] S. Singh, M. Kearns, Y. Mansour, Nash convergence of gradient dynamics in general-sum games, in: Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence, UAI'00, Morgan Kaufmann, San Francisco, CA, 2000, pp. 541–548.
[5] J. Hu, M. Wellman, Nash Q-learning for general-sum stochastic games, J. Mach. Learn. Res. 4 (2003) 1039–1069.
[6] B. Banerjee, J. Peng, Generalized multiagent learning with performance bound, Autonomous Agents Multi-Agent Syst. (ISSN 1387-2532) 15 (3) (2007) 281–312.
[7] M. Bowling, M. Veloso, Multiagent learning using a variable learning rate, Artificial Intelligence 136 (2) (2002) 215–250.
[8] M. Bowling, Multiagent learning in the presence of agents with limitations, PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2003.
[9] Y. Chang, L.P. Kaelbling, Playing is believing: The role of beliefs in multi-agent learning, in: NIPS'01, Vancouver, Canada, 2001, in: Adv. Neural Inf. Process. Syst., vol. 14, MIT Press, Cambridge, MA, 2002, pp. 1483–1490.
[10] G. Tesauro, Extending Q-learning to general adaptive multi-agent systems, in: S. Thrun, L. Saul, B. Scholkopf (Eds.), NIPS'04, in: Adv. Neural Inf. Process. Syst., vol. 16, MIT Press, Cambridge, MA, 2004.
[11] Y. Shoham, R. Powers, T. Grenager, If multi-agent learning is the answer, what is the question? Artificial Intelligence 171 (7) (2007) 365–377.
[12] J.W. Crandall, M.A. Goodrich, Learning to compete, compromise, and cooperate in repeated general-sum games, in: Proceedings of the Twenty Second International Conference on Machine Learning, ICML'05, ACM Press, 2005, pp. 161–168.
[13] D. Fudenberg, D. Levine, The Theory of Learning in Games, MIT Press, Cambridge, MA, 1998.
[14] C. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (3) (1992) 279–292.
[15] H.P. Young, Individual Strategy and Social Structure: An Evolutionary Theory of Institutions, Princeton University Press, Princeton, NJ, 1998.
[16] L. Shapley, Stochastic games, Proc. Natl. Acad. Sci. 39 (1953) 1095–1100.
[17] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA, 1998.
[18] E. Nudelman, J. Wortman, Y. Shoham, K. Leyton-brown, Run the gamut: A comprehensive approach to evaluating game-theoretic algorithms, in: Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS'04, 2004, pp. 880–887.
[19] C. Darken, J. Moody, Note on learning rate schedule for stochastic optimisation, in: NIPS'91, in: Adv. Neural Inf. Process. Syst., vol. 3, Morgan Kaufmann, San Mateo, CA, 1991, pp. 832–838.
[20] S. Thrun, Efficient exploration in reinforcement learning, PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1992.
[21] D. Chakraborty, S. Sen, MB-AIM-FSI: A model based framework for exploiting gradient ascent multiagent learners in strategic interactions, in: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS'08, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2008, pp. 371–378.
[22] R. Powers, Y. Shoham, Learning against opponents with bounded memory, in: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, IJCAI'05, Edinburgh, UK, 2005.
[23] R. Powers, Y. Shoham, New criteria and a new algorithm for learning in multi-agent systems, in: Lawrence K. Saul, Yair Weiss, Leon Bottou (Eds.), NIPS'05, in: Adv. Neural Inf. Process. Syst., vol. 17, MIT Press, Cambridge, MA, 2005.

Please cite this article in press as: A. Burkov, B. Chaib-draa, Effective learning in the presence of adaptive counterparts, J. Algorithms (2009), doi:10.1016/j.jalgor.2009.04.003