The Role of Reactivity in Multiagent Learning

Bikramjit Banerjee and Jing Peng
Department of Electrical Engineering & Computer Science
Tulane University
New Orleans, LA 70118, USA
{banerjee, jp}@eecs.tulane.edu

Abstract

In this paper we take a closer look at a recently proposed classification scheme for multiagent learning algorithms. Based on this scheme an exploitation mechanism (we call it the Exploiter) was developed that could beat various Policy Hill Climbers (PHC) and other fair opponents in some repeated matrix games. We show on the contrary that some fair opponents may actually beat the Exploiter in repeated games. This clearly indicates a deficiency in the original classification scheme which we address. Specifically, we introduce a new measure called Reactivity *that measures how fast a learner can adapt to an unexpected hypothetical change in the opponent's policy. We show that in some games, this new measure can approximately predict the performance of a player, and based on this measure we explain the behaviors of various algorithms in the Matching Pennies game, which was inexplicable by the original scheme. Finally we show that under certain restrictions, a player that consciously tries to avoid exploitation may be unable to do so.*

1. Introduction

Multiagent Learning (MAL) is an intersection of Distributed Artificial Intelligence (DAI) and Machine Learning (ML). Broad and well-developed as Machine Learning is, it is still incomplete without adequately addressing learning in Multiagent Systems (MAS). Agents in a MAS typically operate in large, complex, open, dynamic and unpredictable environments. Therefore it is necessary to endow such agents with capabilities to adapt to such environments. MAL is also an area where ML and Game Theory [16] meet. The latter has been extensively used for modeling concurrent reinforcement learning problems. Several algorithms for multiagent reinforcement learning have been proposed [6, 9, 11, 10], mostly guaranteed to converge to an equilibrium in the limit. It has been argued [2] that one of the key requirements for a MAL algorithm is convergence to a stationary policy, usually conditioned on the other agents' play.

Recently Chang and Kaelbling [5] have argued that most of the existing MAL algorithms assume that the opponents are playing stationary policies. While some others like Bully and Godfather [12] do assume that the opponents are also learning, they make a limited usage of history. Multiplicative weight adaptation [7] assumes the opponent can play non-stationary strategies but is limited by the assumption of complete knowledge of the opponents' strategies. Chang and Kaelbling then suggested a classification scheme that identifies a learner's usage of history ($\mathcal{H}$) and its belief ($\mathcal{B}$) about the opponent using their histories to play possibly non-stationary policies. An agent in class $\mathcal{H}_s \times \mathcal{B}_t$ uses its memory of s previous time periods to decide its current strategy and believes that the opponent does the same for t previous time periods. Having an opponent model has been shown to present advantage to a learner before [4]. Chang and Kaelbling have shown that the ability to model an opponent's learning algorithm may produce a learner in the league $\mathcal{H}_\infty \times \mathcal{B}_\infty$ that often achieves more than the equilibrium payoff against several existing learning algorithms. This learner, called PHC-Exploiter (we shall call it Exploiter henceforth), was found to score at least the expected equilibrium payoff against many PHC variants, Q-learners (Q_0 and Q_1), Minimax and Nash-Qs[10, 9] and other "fair opponents", i.e. learners in $\mathcal{H} \times \mathcal{B}$ class equal or less capable than the Exploiter. In this paper we show that though the Exploiter has been explicitly designed to beat PHC and other fair opponents, some of these opponents can actually beat the Exploiter. Based on this observation we argue that the $\mathcal{H} \times \mathcal{B}$ based classification scheme is deficient. We then present a novel criterion called *reactivity* for rating various learning algorithms and quantify an *effective reactivity* for policy iterators. We show that this criterion can effectively explain the behaviors of the PHC variants against the Exploiter in the Matching Pennies game. We also show that under some restrictions and a specific sense of the term *ex-*

ploitation, a player that consciously tries to avoid exploitation may be unable to do so.

2. Background & Definitions

Here we provide definitions of key concepts for our work. We refer to A_i as the set of possible actions available to the ith agent. A *mixed policy*, π_i for agent i, is a probability distribution over A_i. If the entire probability mass is concentrated on a single action, it is also called a *pure policy*.

Definition 1 *A bimatrix game is given by a pair of matrices, (M_1, M_2), (each of size $|A_1| \times |A_2|$ for a two-agent game) where the payoff of the kth agent for the joint action (a_1, a_2) is given by the entry $M_k(a_1, a_2),\ \forall(a_1, a_2) \in A_1 \times A_2,\ k = 1, 2$.*

A *constant-sum game* (also called competitive games) is a special bimatrix game where

$$M_1(a_1, a_2) + M_2(a_1, a_2) = c,\ \forall(a_1, a_2) \in A_1 \times A_2$$

where c is a constant. If $c = 0$, then it is also called a zero-sum game.

We consider the problem of concurrent learning in the context of repeated play of a bimatrix game by two agents. The policy of the opponent of agent i is written as π_{-i}. The expected payoff of agent i is

$$V_i(\pi_i, \pi_{-i}) = \sum_{(a_1,a_2)\in A_1\times A_2} \pi_1(a_1)\pi_2(a_2)M_i(a_1, a_2),\ i = 1, 2$$

In a repeated game, the goal of the learner (agent i) is to deduce a policy that maximises $V_i(\pi_i, \pi_{-i})$ against the opponent's policy.

Definition 2 *A* best response *of agent i to its opponent's policy, π_{-i}, is a set of probability vectors $BR_i(\pi_{-i})$ defined as $BR_i(\pi_{-i}) = \{\pi_i^* \in PD(A_i)|V_i(\pi_i^*, \pi_{-i}) \geq V_i(\pi_i, \pi_{-i}), \forall\pi_i \in PD(A_i)\}$, where $PD(A_i)$ is the set of probability distributions over A_i.*

It is the set of optimal policies that an agent can play to maximize its expected payoff given the opponent is playing π_{-i}. If both of the players are playing mutual best responses, then they are said to be in a *Nash equilibrium*. It is a joint policy point from where no player has any incentive for unilateral deviation, given the other's strategy. Such an equilibrium is a characteristic of the game (M_1, M_2) but no such equilibrium in pure policies may exist for some games. However, there always exists at least one such equilibrium in *mixed policies* for an arbitrary finite bimatrix game [14].

3. Policy Gradient Learners

Recent convergence results in single-agent policy gradient learning [17, 18] encouraged the rise of variants of Policy Hill Climbers (PHC) like WoLF (Win or Learn Fast) [2, 3] which has been shown to learn stationary best responses to the opponents in some small games. The simple PHC algorithm updates a value function (Q values) according to the usual Q-learning rule [1] and then increases the probability of the maximising action by a fixed $\delta \in (0, 1]$, while decreasing the probabilities of the other actions uniformly to keep π a legal probability distribution. It does not converge to Nash equilibrium policies in self play in all games, but WoLF-PHC which uses a variable rate δ according to the scheme

$$\delta = \begin{cases} \delta_w & \text{if } \sum_a \pi(s,a)Q(s,a) > \sum_a \bar{\pi}(s,a)Q(s,a) \\ \delta_l > \delta_w & \text{otherwise} \end{cases} \tag{1}$$

where $\bar{\pi}(s, a)$ is an average policy calculated over time, has been empirically shown to converge in many interesting games. The situation where $\sum_a \pi(s, a)Q(s, a) > \sum_a \bar{\pi}(s, a)Q(s, a)$, i.e., where the expected payoff of the learner is greater than that of an average policy against the opponent's current policy, is called *winning*. We shall refer to the ratio $\frac{\delta_w}{\delta_l}$ as λ. All these variants of PHC believe that the opponent is playing a stationary policy and hence try to learn a stationary best response to that policy. Chang and Kaelbling [5] classified these as $\mathcal{H}_\infty \times \mathcal{B}_0$ learners as they have full access to their history ($\mathcal{H}$) but believe ($\mathcal{B}$) that the opponent plays a stationary strategy i.e., that the opponent does not use any memory. They demonstrated that such deficiency in beliefs could be exploited to their disadvantage while eluding convergence. The Exploiter (adapted from [5] in Table 1 for clarity) essentially belongs to the league $\mathcal{H}_\infty \times \mathcal{B}_\infty$ since it believes that the opponent is playing a non-stationary strategy and learns when losing (at the same rate that it estimates the opponent to be learning, this estimate being $\hat{\delta}_2$) but exploits when winning, by playing the maximising action deterministically (step e). The Exploiter also uses a slightly different method to compute the WoLF condition than equation 1. It explicitly estimates the opponent's policy (step c) and uses either estimation or apriori knowledge (we have endowed the Exploiter with this latter option) for its equilibrium policy, and accordingly computes the right hand side of equation 1. Moreover, in all experimental comparisons among algorithms, all common parameters, viz. α, δ will be assumed identical unless otherwise noted.

---

1 The rule is usually given by $Q_{t+1}(a) = (1 - \alpha)Q_t(a) + \alpha r_t$ where α is the learning rate, r_t is the payoff at time t and a is the action whose value is updated.

1. Assume Expoliter is agent 1 and opponent is agent 2 and a single state problem (repeated game). Input $\alpha, \delta \in (0, 1]$, the learning rates. Initialize $Q(a) \leftarrow 0$, $\pi_1(a) \leftarrow \frac{1}{|A_1|}$.
2. Repeat
   (a) Select action a according to π_1 with suitable exploration.
   (b) Observe reward r and update $Q(a) \leftarrow (1 - \alpha)Q(a) + \alpha r$.
   (c) Observe action a_2^t of opponent (time t), update history h and calculate an estimate of opponent's policy, $\hat{\pi}_2^t(b) = \frac{\sum_{\tau=t-W}^{t} \#(h[\tau]=b)}{W}, \forall b \in A_2$, where W is size of estimation window, $\#(h[\tau] = b) = 1$ if $a_2^\tau = b$, else $\#(h[\tau] = b) = 0$.
   (d) Calculate estimate of opponent's learning rate, $\hat{\delta}_2 \leftarrow \frac{|\hat{\pi}_2^t - \hat{\pi}_2^{t-W}|}{W}$.
   (e) If $\sum_a \pi_1(a)Q(a) > V_1(\pi_1^*, \hat{\pi}_2^t)$
   $$\pi_1(a) \leftarrow \begin{cases} 1 & \text{if } a = \arg\max_{a'} Q(a') \\ 0 & \text{otherwise} \end{cases}$$
   else
   $$\pi_1(a) \leftarrow \pi_1(a) + \begin{cases} \hat{\delta}_2 & \text{if } a = \arg\max_{a'} Q(a') \\ \frac{-\hat{\delta}_2}{|A_1|-1} & \text{otherwise} \end{cases}$$

Table 1. The PHC-Exploiter Algorithm

4. Exploitation in Matching Pennies

A frequent strategic situation is one where one player tries to match the other's action while the other wants to select an action distinct from the opponent's. E.g. a goalie wants to dive in the same direction as the shot whereas the shooter's intention is exactly the opposite. When a magician asks the spectators to guess which of his hands holds the coin, he intends them to choose the wrong hand while the spectators want to embarrass him. Such situations are represented by the simple bimatrix zero-sum game $M_1(H, H) = M_1(T, T) = -1$, $M_1(H, T) = M_1(T, H) = 1$, $M_2(.,.) = -M_1(.,.)$ called *matching pennies*. Here $A_1 = A_2 = \{H, T\}$, and the goal of agent 1 is to mismatch action with agent 2, while the latter desires a match. This game has no pure equilibrium and the only equilibrium is mixed, with each player's actions being equally likely. This game is suited for studying cumulative gains since in each iteration the payoffs can either increase or decrease by 1. The Exploiter designed by Chang and Kaelbling [5] was shown to beat WoLF in cumulative gain in this game. We shall concentrate on this game in this paper though our results are applicable in other zero-sum bimatrix games where reactivity (defined in section 6) is a key factor affecting performance.

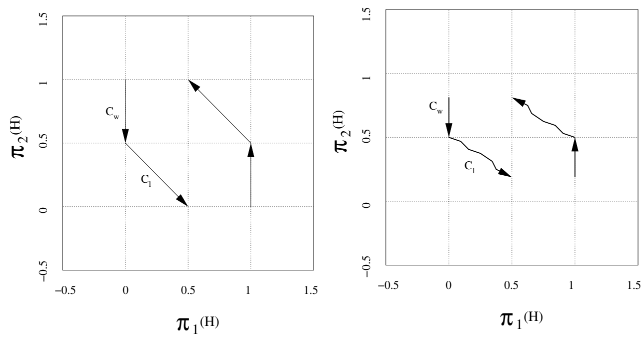

Figure 1. The Exploiter's joint policy trajectory against WoLF-PHC. Left: Trajectory same as against PHC when $\hat{\delta}_2$ is accurate. Right: Typical trajectory when $\hat{\delta}_2$ is inaccurate.

The Exploiter's joint trajectory in the policy space with a PHC player is (shown in Figure 1 left) as follows [5]: the Exploiter (player 1 in game (M_1, M_2)) selects heads with probability 0 ($\pi_1(H) = 0$) as long as the opponent's probability of selecting heads is $\pi_2(H) \geq 0.5$, while the PHC reduces $\pi_2(H)$ from 1 to 0.5. In this phase the exploiter is winning and hence exploiting the fact that in majority cases he is likely to "mismatch" the opponent by playing T deterministically. As in [5], this part of the cycle is called C_w. After this the exploiter starts losing and increases $\pi_1(H)$ linearly (at the same estimated rate as the opponent is decreasing $\pi_2(H)$ from 0.5 to 0, i.e., $\hat{\delta}_2$) till the PHC's $\pi_2(H)$ is down to 0. This loss phase of the cycle is referred to as C_l.Then the Exploiter instantly shifts $\pi_1(H)$ to 1 and continues in a symmetric fashion. The analysis for the behavior of the Exploiter against PHC was presented in [5]. In this paper we present the analysis of the Exploiter's behavior against a WoLF-PHC player and show that though the latter belongs to the class $\mathcal{H}_\infty \times \mathcal{B}_0$, it can still beat the Exploiter in cumulative payoffs under certain conditions.

4.1. Exploiter versus WoLF-PHC

We note that when $\hat{\delta}_2$ is accurate at all times, the joint trajectory against a WoLF-PHC will be identical to that against a PHC player, as shown in Figure 1 left. If the estimate $\hat{\delta}_2$ is inaccurate (described below) then the joint trajectory will be something like Figure 1 right. However, the key difference between PHC and WoLF-PHC as opponents, is that the opponent WoLF will be able to get out of its loss phase (win phase for Exploiter, C_w) at a faster rate (δ_l) than PHC can. We show below that if δ_l is fast enough, WoLF can actually beat Exploiter in cumulative payoffs.

Lemma 1 *Assume that the Exploiter knows the opponent WoLF-PHC's policy at all times, but calculates a time varying (inaccurate) estimate of the opponent's learning rate, i.e., $\hat{\delta}_2(t)$ such that*

$$\theta = \frac{1}{\tau^2}\int_0^\tau \hat{\delta}_2(t)(\tau^2 - t^2)dt$$

assuming the above integral exists. τ is the length of a loss phase for the Exploiter. Then the WoLF-PHC can beat the Exploiter in cumulative payoffs, i.e., $\int_{C_l+C_w} V_1(\pi_1^t, \pi_2^t)dt < 0$, if the former uses

$$\lambda < 1 - 2\theta.$$

Proof: If the Exploiter knows the opponent's policy at all times t, then its computation of the winning criterion becomes more accurate. In case of Matching Pennies the equilibrium payoff of the Exploiter is 0 against any opponent policy, so the Exploiter compares its current expected payoff to 0 to identify winning situations. Writing $\pi_1^t(H)$ and $\pi_2^t(H)$ simply as π_1 and π_2 respectively, we get the Exploiter's expected payoff at time t is given by

$$\begin{aligned} V_1(\pi_1, \pi_2) &= -\pi_1\pi_2 - (1-\pi_1)(1-\pi_2) + \pi_1(1-\pi_2) \\ &\quad +(1-\pi_1)\pi_2 \\ &= -4\pi_1\pi_2 + 2\pi_1 + 2\pi_2 - 1 \end{aligned}$$

During C_l, the opponent WoLF tries to move π_2 from 0.5 to 0 linearly at the rate of δ_w (WoLF is winning in this phase) and the Exploiter changes π_1 from 0 to 0.5 at the variable rate $\hat{\delta}_2(t)$. However, given the way the Exploiter computes $\hat{\delta}_2(t)$, it is easy to see that $\hat{\delta}_2(t) \geq \delta_w$ at all times in C_l. Thus the Exploiter will reach 0.5 *before* WoLF reaches 0, and let this time (when C_l ends) be τ. This is visualized in Figure 1 right. Now the instantaneous policies can be expressed as functions of time t as $\pi_1 = \int_0^t \hat{\delta}_2(t)dt$ and $\pi_2 = \frac{1}{2} - \delta_w t$ where C_l lasts from $t = 0$ to $t = \tau$. Consequently the expected payoff of the Exploiter in C_l is

$$V_1^L = \int_{C_l} V_1(\pi_1, \pi_2)dt = (2\theta - 1)\delta_w\tau^2$$

where θ is as given in the statement of the Lemma. During C_w, unlike against PHC, the joint trajectory starts at $(0, 0.5 + \delta_w\tau)$ instead of $(0, 1)$. In this case, $\pi_1 = 0$ and $\pi_2 = 0.5 + \delta_w\tau - \delta_l t$ and C_w lasts from $t = 0$ to $t = \frac{\delta_w\tau}{\delta_l} = \lambda\tau$, and the expected payoff of the Exploiter in C_w is

$$V_1^W = \int_{C_w} V_1(\pi_1, \pi_2)dt = \frac{\delta_w^2\tau^2}{\delta_l}$$

Thus the total reward over a complete cycle, $V_1 = 2 \times (V_1^W + V_1^L)$ can be negative when $(V_1^W + V_1^L) < 0$, i.e., when $\lambda < 1 - 2\theta$. ∎

Corollary 1 *If the Exploiter can estimate the opponent's learning rate accurately at all times, then it will be beaten in cumulative payoff by a WoLF-PHC player if the latter uses $\lambda < \frac{1}{3}$.*

Proof: As stated earlier, in this case the joint trajectory will be similar to that against a PHC player (Figure 1 left) since during C_l, both Exploiter and WoLF use the fixed learning rate of δ_w. That means in C_l, $\hat{\delta}_2(t) = \delta_w$ at all times t and the length of C_l becomes $\tau = \frac{1}{2\delta_w}$. Using the computed values we get $\theta = \frac{2}{3}\delta_w\tau$ and the result follows from Lemma 1. ∎

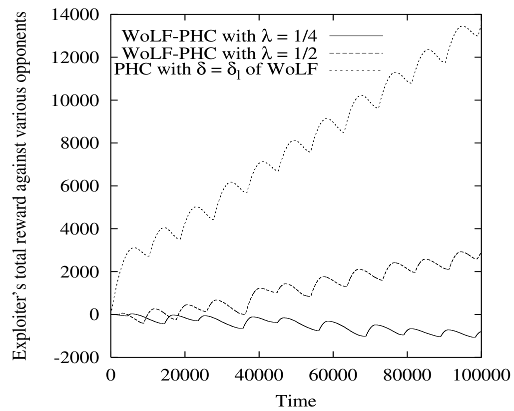

Figure 2. Exploiter's cumulative gains against a WoLF-PHC using $\delta_l = 8 \times 10^{-5}, \delta_w = 2 \times 10^{-5}$ (hence $\lambda = \frac{1}{4}$), against another WoLF-PHC using $\delta_l = 8 \times 10^{-5}, \delta_w = 4 \times 10^{-5}$ (hence $\lambda = \frac{1}{2}$), and against a PHC player using $\delta = 8 \times 10^{-5}$ (=δ_l of WoLF-PHC).

The above results are summarized in the experimental payoff curves of the Exploiter in Figure 2.

Lemma 2 *If the Exploiter knows accurately the opponent WoLF-PHC's policy and learning rate at all time, and the WoLF-PHC agent uses $\delta_w = 0$ (i.e., $\lambda = 0$, we call such an agent WoLF-0), then instead of any repeating cycle, their policy trajectories always come to a halt at*

1. *the initial joint policy point (π_1^0, π_2^0), if $0.5 \leq \pi_1^0 \leq 1$ and $0.5 \leq \pi_2^0 \leq 1$ or if $0 \leq \pi_1^0 \leq 0.5$ and $0 \leq \pi_2^0 \leq 0.5$.*
2. *$(0, 0.5)$ if $0 \leq \pi_1^0 < 0.5$ and $0.5 < \pi_2^0 \leq 1$.*
3. *$(1, 0.5)$ if $0.5 < \pi_1^0 \leq 1$ and $0 \leq \pi_2^0 < 0.5$.*

Proof: The proof is straightforward noting that (i) C_w cannot last indefinitely and (ii) in any C_l phase, since the WoLF is winning, π_2 will stop changing and the Exploiter trying to move at the *same* pace as the opponent at that time, will also effectively stop changing π_1. ∎

It is worthwhile to note that this version of WoLF has been studied before under the title of *Win Stay, Lose Shift*[2] [15]. Interestingly, the Exploiter is never able to beat WoLF-0 in expected payoffs in the steady state, as seen from the next corollary.

Corollary 2 *If the Exploiter knows accurately the opponent WoLF-0's policy and learning rate at all time, then the former's expected payoff at any time after the policy trajectories have come to rest is* ≤ 0.

4.2. Exploiter versus Q-learner

We consider a specific version of a Q-learner for a stage game, named $\mathcal{Q}$, and defined as follows.

Definition 3 *$\mathcal{Q}$ uses the usual value function update rule $Q_{t+1}(a) = (1-\alpha)Q_t(a) + \alpha r_t$ where α is its learning rate and r_t is the payoff at time t, selects action a_t according to rule $a_t = \arg\max_b Q_t(b)$ while randomizing uniformly among actions with equal (and maximum) Q-value, and apart from this uses no other exploration scheme.*

Clearly $\mathcal{Q}$ can adapt to the opponent's policy quickly depending on how large α is. If $\alpha = 1$ it will exhibit a knee-jerk reaction to the opponent's last action and will generally not let the opponent to hold it back in the vicinity of a particular strategy in its policy space, that the Exploiter can exploit. This is precisely what the Exploiter does to a PHC (or variant) and succeeds whenever the PHC (or variant) is slow to *react*. Hence, by construction we expect the $\mathcal{Q}$ to be generally unexploitable by the Exploiter (i.e., $V_1 \not> 0$). Figure 3 (left) presents the experimental expected payoff curves of the Exploiter against $\mathcal{Q}$, and suggests that $\mathcal{Q}$ is actually capable of beating the Exploiter. Note, though, that in some cases the Exploiter may be able to exploit the $\mathcal{Q}$, e.g., when the Exploiter knows that $\mathcal{Q}$'s $\alpha = 1$ i.e. it always plays best response to the Exploiter's last action, the Exploiter can play the actions H, T alternately always ensuring $+1$ reward at every step. This will be discussed again in light of our reactivity measure.

5. The Classification Scheme

We have identified PHC variants and $\mathcal{Q}$ belonging to the group $\mathcal{H}_\infty \times \mathcal{B}_0$ (a class that [5] refers to as *fair* opponents for the Exploiter) that can defeat the Exploiter in cumulative expected payoffs under certain conditions. This is contrary to the purpose of the $\mathcal{H} \times \mathcal{B}$ classification scheme, since the scheme predicts that the Exploiter should score at least the minimax payoff in Matching Pennies against any fair opponent. Moreover, $\mathcal{Q}$ belongs to $\mathcal{H}_\infty \times \mathcal{B}_0$ when $\alpha < 1$

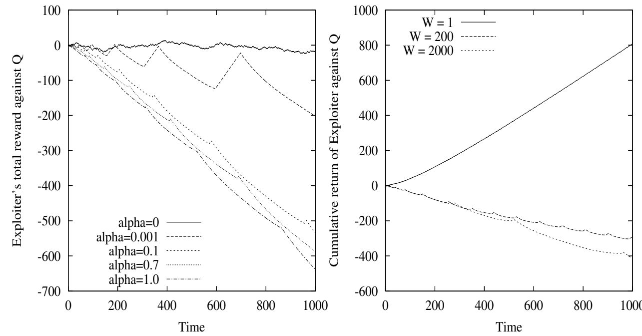

Figure 3. Exploiter's cumulative gain against $\mathcal{Q}$ for various values of $\mathcal{Q}$'s α (left) and Exploiter's W (right). Left: Exploiter's α is fixed at 0.9, and its $W = 3000$. Right: Exploiter's $\alpha = 0.9$ and $\mathcal{Q}$'s $\alpha = 0.8$.

and $\mathcal{H}_1 \times \mathcal{B}_0$ when $\alpha = 1$. The classification scheme requires that $\mathcal{H}_1 \times \mathcal{B}_0$ be a weaker class than $\mathcal{H}_\infty \times \mathcal{B}_0$ but the former version of $\mathcal{Q}$ attains a higher payoff (less exploitation) than the latter version against the Exploiter in Figure 3 (left). These are behaviors that the $\mathcal{H} \times \mathcal{B}$ scheme clearly fails to explain.

Our take at this apparent contradiction is that the $\mathcal{H} \times \mathcal{B}$ scheme is incomplete, even though it is undoubtedly important in delineating learner capabilities. What it essentially states is that if an agent *uses* more resources (i.e., belongs to a higher league of $\mathcal{H} \times \mathcal{B}$) then it should be more capable than the ones using less resources. However, the possibility of an agent using less resource but in a *more effective* manner being still able to beat a higher league opponent has been overlooked. We reconcile this latter observation with the essence of the $\mathcal{H} \times \mathcal{B}$ classification scheme by the following hypothesis:

Hypothesis 1 *The $\mathcal{H}_s \times \mathcal{B}_t$ scheme, rather than representing the capability of all agents in a group [5], actually represents the* maximal *capability of any agent in that group. An agent with the maximal capability of its group cannot be beaten by the corresponding (or any) agent in a lower group. However, it cannot be guaranteed that the former will beat the latter.*

This allows for agents in higher groups to be beaten by those of lower groups simply because the former makes suboptimal use of its available resources. An important example of an agent attaining maximal capability in its class is a *universally consistent* learner [8, 7]. Such a learner is *safe*, i.e., attains its minimax payoff at the very least, irrespective of the opponent, e.g., smooth fictitious play [8] but not fictitious play. Multiplicative weight [7] which has this property, belongs to the league of $\mathcal{H}_\infty \times \mathcal{B}_\infty$ as observed in [5]. But Exp3 [1] which also has this property belongs to the

2 The nomenclature is borrowed from Behavioral Psychology

class $\mathcal{H}_\infty \times \mathcal{B}_0$, and it cannot be beaten by any opponent in a lower or even a higher group. Therefore, Exp3 is an algorithm with the maximal capability in group $\mathcal{H}_\infty \times \mathcal{B}_0$. The above hypothesis is consistent with these observations.

Having noted the limitation of the $\mathcal{H} \times \mathcal{B}$ classification scheme, we now move to identify the key factor affecting the performances of various algorithms in the Matching Pennies game. To develop the intuition for such a measure, we note that a recurring theme in section 4 was that a learner capable of getting out of its loss phase faster usually performed well. It is also worthwhile to note that the Exploiter behaves like $\mathcal{Q}$ when winning but like a PHC when losing. Hence at least when losing, the Exploiter is slow to update its policy and adapt to the opponent. This indicates that the capability of $\mathcal{Q}$ to react faster to the opponent's policies all the time, may be the reason for its superior performance against the Exploiter. In the following section we formalize this idea of *reactivity* and show that it can indeed explain the curves in Figures 2,3. We do not tout reactivity as a sufficient criterion for rating learner capabilities. We only claim that it is a *necessary* criterion in case of some specific games like Matching Pennies, Rock-Scissors-Paper etc., and in conjunction with the $\mathcal{H}\times\mathcal{B}$ scheme, *might* be sufficient. However, we do not verify this latter part of the claim in this paper.

6. Reactivity

The key intuition behind the Exploiter's success against PHC variants is that when winning, the Exploiter *adapts immediately* to the opponent's policy, but adapts more cautiously (no slower than its estimate of the opponent's rate) when losing. We call this speed of adaptibility, *reactivity* and denote it by R_i^t for agent i at time t. We propose a generic definition of reactivity as below.

Definition 4 *The reactivity of agent i at time t is given by*

$$R_i^t = f(\mathcal{T}) \text{ where } \mathcal{T} = \min_{\tau>0} \max_{\pi_{-i}} \tau,\ s.t.\ \pi_i^{t+\tau} \in BR_i(\pi_{-i})$$

where π_{-i} is a hypothetical policy that the opponent might turn to at exactly time t and continue to play it thereafter. The exact functional form f (an appropriate inverse function) needs to be defined in context of the specific player and the game being played.

In other words, imaginably, if the opponent switches its policy (to π_{-i}) at time t and maintains it for sufficiently long, then a function of the shortest time after which the player i can play a best response to the worst case switch of the opponent is its reactivity at that time. Obviously this would also depend on the initial configuration at time t and the target configuration arising due to the opponent's switch (imaginary), which will be nullified through the choice of f. Now some of the algorithms under study exhibit different reactivities at win times ($t \in W$) than at loss times ($t \in L$), so we generally distinguish between reactivities at loss times $R_i^L = \max_{t\in L}\{R_i^t\}$ and that at win times, R_i^W defined similarly.

Analysis in the Matching Pennies Game

In the following analysis we assume the algorithms normally using Q-update rule actually use the following *informed Q-update rule*. This makes the analysis easier.

Definition 5 *An informed Q-update rule for agent i is one that allows update of all action values by their expected payoffs at every iteration t,*

$$Q_i^{t+1}(a) = Q_i^t(a) + \alpha[V_i(a, \pi_{-i}^t) - Q_i^t(a)],\ \forall a \in A_i$$

We write $\tilde{\pi}_i$ as the target policy for the learner after the opponent switches its policy to stationary π_{-i} as stated in Definition 4, and let $\tilde{Q}_i$ be the resulting target Q-value function for the learner. Let $\Delta_Q^t = \|Q_i^t - \tilde{Q}_i\|$ where $\|.\|$ is a max norm over the action space. Then the following lemma establishes the value of $\mathcal{T}$ (Definition 4) for PHC in the Matching Pennies game.

Lemma 3 *For a PHC player (player i) using an informed Q-update rule and $\delta \geq \delta_0$,*

$$\mathcal{T} = \frac{\log\left(\frac{\Delta_Q^t}{\|\tilde{Q}_i\|}\right)}{\log\left(\frac{1}{1-\alpha}\right)} + \frac{1}{\delta} \tag{2}$$

at any time t, where $\delta_0 = \|\pi_i^t\| \log\left(\frac{1}{1-\alpha}\right) / \log\left(\frac{\Delta_Q^t}{\|\tilde{Q}_i\|}\right)$.

Proof: Without loss of generality assume that the target policy before time t (i.e., before the opponent's switch as in Definition 4) is $[1, 0]$. That means $Q_i^t = [a, -a]$ for some $a > 0$. Then, given that δ is high enough (i.e. $\geq \delta_0$), the worst case switch by the opponent will make $\tilde{\pi}_i = [0, 1]$, i.e., $\tilde{Q}_i = [-b, b]$ for some $b > 0$. The result is established noting that action values have to switch signs before the policy can approach $\tilde{\pi}_i$. ∎

If $\delta < \delta_0$, there are some distinct cases leading to different values of $\mathcal{T}$ but it can be shown that identical functional forms of $\mathcal{T}$ applies to WoLF and Exploiter in all of those cases, though for Exploiter when winning and for $\mathcal{Q}$, the second term in the sum (equation 2) is 1 (also in all cases). Specifically, for the algorithms under study and all admissible values of δ (i.e., $1 \geq \delta \geq 0$), we have a generic form of $\mathcal{T}$ given by $\mathcal{T} = d_1 g(\alpha) + d_2 h(\delta)$ where d_1, d_2 are dependent on the initial situation, and $g(\alpha) = 1/\log(\frac{1}{1-\alpha})$, $h(\delta) = \frac{1}{\delta}$. For $\mathcal{Q}$ and winning Exploiter, $h = d_2 = 1$. For the purpose of comparison of the algorithms, we eliminate the initial conditions in the form of d_1, d_2, and choose the inverse form of f (Definition 4) as $f = \frac{1}{g.h}$ when $0 \leq \alpha < 1$ and $1 \geq \delta \geq 0$. The following analysis also holds for the same ranges of α, δ.

Thus we have, $R^t_{PHC} = \log\left(\frac{1}{1-\alpha}\right)\delta, \quad \forall t$, $R^L_{WoLF} = \log\left(\frac{1}{1-\alpha}\right)\delta_l > \log\left(\frac{1}{1-\alpha}\right)\delta_w = R^W_{WoLF}$, and $R^L_{Exploiter} = \log\left(\frac{1}{1-\alpha}\right)\hat{\delta}_2 \leq \log\left(\frac{1}{1-\alpha}\right) = R^W_{Exploiter}$. We also define the *overall reactivity* of agent i as $R_i = \max\{R^W_i, R^L_i\}$. However, there is another crucial question that has not been addressed, in addition to reactivity: *when* does the agent display higher reactivity? In general, when an agent is losing it should adapt faster but should adapt slowly when winning in order to prolong the winning phase to maximise cumulative return[3]. Thus this question is critical in predicting the performance of the learning algorithms under study against an opponent like the Exploiter in games like Matching Pennies. Hence we define a new criterion for rating different PHC variants called *effective reactivity* and denoted by $\mathcal{R}_i$, which tempers the overall reactivity R_i value by a fraction that specifies how high the loss time reactivity (R^L_i) is relative to the sum of loss and win time (R^W_i) reactivities, i.e., $\mathcal{R}_i = \left(\frac{R^L_i}{R^L_i + R^W_i}\right) R_i$.

According to this definition, $\mathcal{R}_{PHC} = \frac{1}{2}\log\left(\frac{1}{1-\alpha}\right)\delta$, and $\mathcal{R}_{WoLF} = \left(\frac{1}{1+\frac{\delta_w}{\delta_l}}\right)\log\left(\frac{1}{1-\alpha}\right)\delta_l$. Since $\delta_l > \delta_w$, $\mathcal{R}_{WoLF} > \frac{1}{2}\log\left(\frac{1}{1-\alpha}\right)\delta_l$, which means WoLF has a higher effective reactivity than PHC even when the latter uses $\delta = \delta_l$. Therefore the Exploiter scores higher against such a PHC than against WoLF as seen in figure 2. Also, $\mathcal{R}_{Exploiter} = \left(\frac{1}{1+\hat{\delta}_2}\right)\log\left(\frac{1}{1-\alpha}\right)\hat{\delta}_2$. In practice usually $\hat{\delta}_2 \ll \frac{\delta_w}{\delta_l}$ which means usually Exploiter has a higher effective reactivity than WoLF, but if $\lambda = \frac{\delta_w}{\delta_l}$ is small enough, the performances may be reversed. Note that this predicts the result of Corollary 1 only approximately. However, accurate prediction would necessitate a more intricate measure but we find the simplicity of $\mathcal{R}$ more appealing.

Now in case of WoLF-0, the effective reactivity is $\mathcal{R}_{WoLF-0} = \log\left(\frac{1}{1-\alpha}\right)\delta_l > \mathcal{R}_{Exploiter}$, when the α values are equal even if the Exploiter's estimate $\hat{\delta}_2$ is sufficiently close to δ_l used by WoLF-0, since $\delta_l \geq \hat{\delta}_2 \geq 0$. Hence the Exploiter can never be expected to beat WoLF-0 (in steady state), which is in accordance with Corollary 2.

Finally we note that the effective reactivity of $\mathcal{Q}$ is $\frac{1}{2}\log\left(\frac{1}{1-\alpha}\right)$. So in order to achieve $\mathcal{R}_{Exploiter} > \mathcal{R}_{\mathcal{Q}}$, we need $\hat{\delta}_2 > 1$ if α values are equal. But is this possible? The Exploiter estimates the opponent's policy (see Table 1) by noting the relative frequencies of his different actions over a window of size W, and calculating the distance between the two distributions resulting from two successive windows, *normalized* by the window size. If we assume this distance measure to be Euclidean, then the maximum value possible for $\hat{\delta}_2$ is $\frac{\sqrt{2}}{W}$. Since usually $W \gg \sqrt{2}$, $\mathcal{R}_{Exploiter} < \mathcal{R}_{\mathcal{Q}}$, when α values of the two agents are equal. In addition, $\mathcal{R}_{\mathcal{Q}}$ can only increase with higher α since $\log\left(\frac{1}{1-\alpha}\right)$ is an increasing function of α in the range $\alpha \in [0, 1]$. These observations fit accurately with the experimental results in Figure 3 (left). Moreover, if any of the algorithms use $\alpha = 0$, its effective reactivity becomes 0 which is expected since such an algorithm effectively loses any capacity of learning or adapting to an opponent. But note from Definition 3 that if $\mathcal{Q}$ uses $\alpha = 0$, it will always play the equilibrium policy in Matching Pennies, if its initial Q-values of all actions were identical. This was the case in our experiments, and the corresponding curve in Figure 3 (left) verifies that it does indeed roughly score the minimax payoff.

It is interesting to note that the more accurately the Exploiter tries to estimate $\hat{\delta}_2$ (larger window), less is its effective reactivity, and the poorer it should perform against $\mathcal{Q}$. However, if it uses a very short window (say size 1, equivalent to saying that the Exploiter believes $\mathcal{Q}$'s $\alpha = 1$, the case mentioned in Section 4.2) then it is likely to have a $\hat{\delta}_2$ as high as 1 sometimes, and with a slightly higher α than $\mathcal{Q}$, it may have a high enough reactivity to beat $\mathcal{Q}$. For instance, if Exploiter uses $\alpha = 0.9$ and $\mathcal{Q}$ uses $\alpha = 0.8$, then $\log\left(\frac{1}{1-\alpha}\right)$ of the former will be 1.43 times higher than that of the latter. In this case the Exploiter is quite likely (but not guaranteed) to have a higher effective reactivity than $\mathcal{Q}$. Figure 3 (right) verifies these claims, and reinforces the necessity of $\mathcal{R}$ in rating learner capabilities in games like Matching Pennies, at least for the types considered in this paper.

7. Can Exploitation be Prevented Consciously?

We have seen thus far, how the Exploiter consciously tries to exploit its victims, and how the latter can sometimes reverse such exploitation. We now ask a related question - if an algorithm tries to avoid exploitation *consciously* can it guarantee to prevent the same irrespective of the opponent? We focus on a specific sense of the term *exploitation* and define it to mean an opponent playing a "bluff" policy to lead the learner into a region of its policy space where the opponent can achieve significant rewards (enough to more than make up for its losses during the bluff) by the means of a "bash" policy, as long as the learner remains in the *vicinity* of this unfavorable position, and be able to repeat this "bluff and bash" cycle indefinitely. Since this is not neces-

3 Incidentally, this is close to the philosophy of WoLF

sarily bad for the learner in *general-sum* games, we focus on zero-sum games only.

Definition 6 *A TAEC (trying to avoid exploitation consciously) agent in class $\mathcal{H}_s \times \mathcal{B}_t$ is one that uses $\mathcal{H}_s$ (and rewards in the s-window) to explicitly identify any exploitation and uses this knowledge in conjunction with $\mathcal{B}_t$ to prevent exploitation.*

We argue (informally) that a TAEC agent in $\mathcal{H}_s \times \mathcal{B}_t$ for any finite s, t that tries to *consciously avoid exploitation* [4] may not be able to do so.

Lemma 4 *Any TAEC agent cannot be absolutely exploitation free in the sense defined above.*

Proof: (Sketch) If an agent assumes that its opponent is using $t < \infty$ units of its history to formulate its policies, then it is possible, in principle, that there is an opponent actually using more than t units of history and being able to exploit the agent because of the deficiency in its belief. Again, if $s < \infty$, then it is possible, in principle, that the opponent spreads its "bluff and bash" cycle over a window significantly larger than s units of time, so that an s-window estimate simply fails to identify any exploitation. Hence it is necessary (but may not be sufficient) that both s, t be infinite for the agent to entertain any hope of being guaranteed exploitation-free. However, we know from [13] that an agent in $\mathcal{H}_\infty \times \mathcal{B}_\infty$ is impossible to design, since though the set of possible strategies is uncountably infinite, there are only countably infinite histories and beliefs. ∎

8. Conclusions and Future Work

In this paper we addressed the apparent deficiency of $\mathcal{H} \times \mathcal{B}$ based classification of MALs by showing that fair opponents can beat a higher league Exploiter in a zero-sum game. Then we presented a new general criterion called *reactivity* that measures how fast an agent can learn a best response to an unexpected worst case switch in the opponent's policy, and showed that it approximately predicts the performance of a learner as a function of the parameters of its learning algorithm, in the Matching Pennies game. Emphasis was placed on the ease of computing this measure, hence it is sufficiently general but still capable of approximately explaining the behaviors of PHC-variants, $\mathcal{Q}$ and the Exploiter in the Matching Pennies game, as seen in experiments. In future we would like to investigate how exactly the new criterion fits into the $\mathcal{H} \times \mathcal{B}$ picture. We also want to generalize our notion of reactivity and extend our experimental results to other zero-sum games and complex stochastic games.

References

[1] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-arm bandit problem. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 322 – 331, Milwaukee, WI, 1995. IEEE Computer Society Press.

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

[3] M. Bowling and M. Veloso. Scalable learning in stochastic games. In *AAAI Workshop Proceedings on Game Theoretic and Decision Theoretic Agents*, Edmonton, Canada, 2002.

[4] D. Carmel and S. Markovitch. Incorporating opponent models into adversary search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 120–125, Menlo Park, CA, 1996. AAAI Press/MIT Press.

[5] Y.-H. Chang and L. P. Kaelbling. Playing is believing: The role of beliefs in multi-agent learning. In *Neural Information Processing Systems*, Vancouver, Canada, 2001.

[6] C. Claus and C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 746–752, Menlo Park, CA, 1998. AAAI Press/MIT Press.

[7] Y. Freund and R. E. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29:79 – 103, 1999.

[8] D. Fudenberg and D. Levine. Consistency and cautious fictitious play. *Journal of Economic Dynamics and Control*, 19:1065 – 1089, 1995.

[9] J. Hu and M. P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proc. of the 15th Int. Conf. on Machine Learning (ML'98)*, pages 242–250, San Francisco, CA, 1998. Morgan Kaufmann.

[10] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proc. of the 11th Int. Conf. on Machine Learning*, pages 157–163, San Mateo, CA, 1994. Morgan Kaufmann.

[11] M. L. Littman. Friend-or-foe Q-learning in general-sum games. In *Proceedings of the Eighteenth International Conference on Machine Learnig*, MA, USA, 2001.

[12] M. L. Littman and P. Stone. Leading best response strategies in repeated games. In *17th International Joint Conference on AI (IJCAI) workshop on Economic Agents, Models and Mechanisms*, 2001.

[13] J. H. Nachbar. Prediction, optimization, and learning in repeated games. *Econometrica*, 65:275 – 309, 1997.

[14] J. F. Nash. Non-cooperative games. *Annals of Mathematics*, 54:286 – 295, 1951.

[15] M. Nowak and K. Sigmund. A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game. *Nature*, 364:56 – 58, 1993.

[16] G. Owen. *Game Theory*. Academic Press, UK, 1995.

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

[18] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pages 1057 – 1063. MIT Press, 2000.

4 Note that a universally consistent player that is absolutely exploitation free is not an example of this type of *learner*.