Exponential Moving Average Q-Learning Algorithm

Mostafa D. Awheda
Department of Systems and Computer Engineering
Carleton University
Ottawa, Canada K1S 5B6
Email: mawheda@sce.carleton.ca

Howard M. Schwartz
Department of Systems and Computer Engineering
Carleton University
Ottawa, Canada K1S 5B6
Email: Howard.Schwartz@sce.carleton.ca

***Abstract*—A multi-agent policy iteration learning algorithm is proposed in this work. The Exponential Moving Average (EMA) mechanism is used to update the policy for a Q-learning agent so that it converges to an optimal policy against the policies of the other agents. The proposed EMA Q-learning algorithm is examined on a variety of matrix and stochastic games. Simulation results show that the proposed algorithm converges in a wider variety of situations than state-of-the-art multi-agent reinforcement learning (MARL) algorithms.**

I. Introduction

In reinforcement learning, an agent learns from the experience of interacting with its environment. After perceiving the state of its environment at each time step, the agent takes an action so that its environment transitions from a state to a new state. A scalar reward signal is used to evaluate the transition. The objective for the agent is to maximize its cumulative reward [12], [13], [15]. Reinforcement learning is also well-suited for multi-agent learning because of its simplicity and generality [4], [5], [10]. Learning is a key element of multi-agent systems (MAS) as it allows an agent to improve its performance by adapting the dynamics of the other agents and environment [17]. Learning encounters some challenges when it is used in multi-agent learning. One of these challenges is that the other learning agents have to be explicitly considered by each learning agent and therefore the environment is nonstationary. The environment of a multi-agent system is no longer stationary as the Markov property is violated by the other learning agents. As a result of non-stationary environment, single agent reinforcement learning techniques are not guaranteed to converge in multi-agent settings.

Several multi-agent reinforcement learning (MARL) algorithms have recently been proposed and studied [2], [3], [6], [8], [11], [14], [17]. All these algorithms assume that each agent knows its own immediate rewards. Some of these algorithms have theoretical results of convergence in general-sum games. The Infinitesimal Gradient Ascent (IGA) algorithm proposed in [14] fails to converge in games with mixed Nash equilibrium. The Win-or-Learn-Fast Infinitesimal Gradient Ascent (WoLF-IGA) and Win-or-Learn-Fast Generalized Infinitesimal Gradient Ascent (GIGA-WoLF) algorithms proposed in [2], [3] fail to converge in some challenging games such as in the Shapley's game [6]. The Nash-Q and the Adapt When Everybody is Stationary Otherwise Move to Equilibrium (AWESOME) algorithms proposed in [8], [11] require the knowledge of the other agents' actions and their immediate rewards as well in order to guarantee convergence to Nash equilibrium [6], [17]. On the other hand, the Weighted Policy Learner (WPL) and Policy Gradient Ascent with Approximate Policy Prediction (PGA-APP) algorithms proposed in [6], [17] converge to Nash eqilibrium in a wider variety of situations without requiring the knowledge of the other agents' immediate rewards.

In this paper, a multi-agent policy iteration learning algorithm is proposed. The proposed algorithm enables agents to converge to a Nash equilibrium. The main contribution of this paper is exploring the idea of using the exponential moving average (EMA) mechanism as a basis to update the agent's strategy. This mechanism has been used before in literature to only estimate the opponents' strategy such as in [16]. In this work, we have evaluated EMA Q-learning, WoLF-PHC [3], GIGA-WoLF [2], WPL [6], and PGA-APP [17] algorithms on a variety of matrix and stochastic games. Due to page limitations, we only show the EMA Q-learning, PGA-APP and WPL algorithms. The WoLF-PHC and GIGA-WoLF algorithms have been shown and discussed in [2], [6]. Compared to state-of-the-art MARL algorithms (WoLF-PHC [3], GIGA-WoLF [2], WPL [6], and PGA-APP [17]), the EMA Q-learning algorithm empirically converges in a wider variety of situations.

This paper is organized as follows. Preliminary Concepts and Notation are provided in Section 2. A new multi-agent policy iteration learning algorithm called the Exponential Moving Average Q-learning (EMA Q-learning) algorithm is proposed in section 3. The simulation and results are presented in Section 4.

II. Preliminary Concepts and Notation

A. Markov Decision Processes

A Markov decision process (MDP) [1], [9] can be described as a tuple, (S, A, R, T), where S is the discrete space of states, A is the discrete space of actions, R is the reward function and T is the transition function. The reward function defines the reward that an agent i receives when choosing an action from the available actions at the given state. The transition function describes how a probability distribution is defined over the next states as a function of the given state and the agent's action [3]. In a Markov decision process (MDP), the objective

978-1-4673-5925-2/13/$31.00 ©2013 IEEE

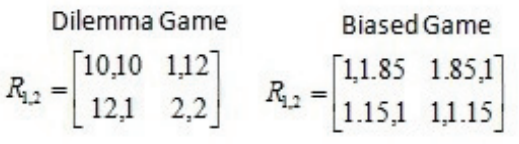

Fig. 1. Matrix games

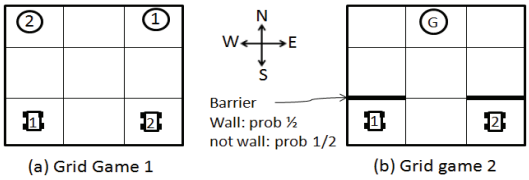

Fig. 2. Two stochastic games [11]

of the agent is to find a policy $\pi : S \rightarrow A$ that maps states to actions so that the discounted future reward is maximized [3], [10].

B. Matrix Games

A matrix game (strategic game) can be described as a tuple $(n, A_{1,\ldots,n}, R_{1,\ldots,n})$, where n is the agents' number, A_i is the discrete space of agent i's available actions, and R_i is the payoff function that agent i receives. In matrix games, the objective of agents is to find pure or mixed strategies that maximize their payoffs. A pure strategy is the strategy that chooses actions deterministically, whereas a mixed strategy is the strategy that chooses actions based on a probability distribution over the agent's available actions. Figure 1 shows some examples of matrix games. The dilemma game and the biased game are shown. In these games, one player chooses a row and the other chooses a column in the matrix. In these games, each cell (i, j) in the payoff matrix represents the payoff recieved by the row and column players, repectively. The dilemma game has a pure Nash equilibrium strategy that executes the second action of each player with a probability of one. On the other hand, the biased game has two mixed Nash equilibrium strategies with probabilities not uniform across actions, (0.15,0.85) and (0.85,0.15).

C. Stochastic Games

A stochastic game can be described as a tuple $(n, S, A_{1,\ldots,n}, R_{1,\ldots,n}, T)$, where n is the agents' number, S is the discrete space of states, A_i is the discrete space of agent i's available actions, R_i is the reward function, and T is the transition function [3]. Stochastic games can be described as an extension of Markov decision processes (MDP). They can also be described as an extension of matrix games as each state in a stochastic game can be dealt with as a matrix game [3]. Figure 2 shows two stochastic games introduced by Hu and Wellman [11]. The players in both games are located in lower corners and are allowed to move one cell in the four compass directions (North, East, South and West). The transition is ignored if both players move to the same cell (excluding a goal cell).The players' goals in both grid games are located as shown in Figure 2. The transition in grid game 1 is deterministic; grid game 2, on the other hand, has deterministic and probabilistic transitions. At the lower corners in grid game 2, the probability of transition to the next cell is 0.5 when the player takes the action North. In both grid games, the player that reaches its goal is rewarded 100 points, it receives -1 points when either it hits the wall or moves into the same cell the other player moves into (excluding a goal cell), and it receives 0 points otherwise. Reaching its goal with a minimum number of steps is therefore the aim of each player in both games. As soon as an agent reaches its goal, the game ends [11]. Grid game 1 has ten different Nash equilibria; whereas grid game 2 has two different Nash equilibria [11]. Figure 3 shows one Nash equilibrium for each grid game.

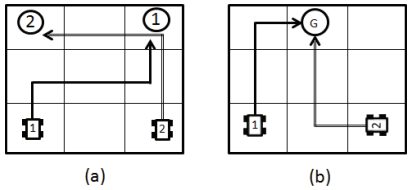

Fig. 3. (a) A Nash equilibrium of grid game 1. (b) A Nash equilibrium of grid game 2 [11].

III. The Exponential Moving Average Q-Learning (EMA Q-Learning) Algorithm

The exponential moving average (EMA) approach is a model-free strategy estimation approach. It is a family of statistical approaches used to analyze time series data in finance and technical analysis. Typically, EMA gives the recent observations more weight than the older ones [7]. The EMA estimation approach is used in [16] by the hyper Q-learning algorithm to estimate the opponent's strategy. It is also used in [7] by the Infinitesimal Gradient Ascent (IGA) agent to estimate its opponent's strategy. The EMA estimator used to estimate the strategy of the agent's opponent(s) can be described by the following equation [7], [16]:

$$\pi^{-j}_{t+1}(s) = (1-\eta)\pi^{-j}_{t}(s) + \eta\vec{u}(a^{-j}) \quad (1)$$

Where $\pi^{-j}(s)$ is the opponent's strategy, η is a small constant step size and $0 < \eta << 1$, and $\vec{u}(a^{-j})$ is a unit vector representation of the action a^{-j} chosen by the opponent $(-j)$ at the state s. The unit vector $\vec{u}(a^{-j})$ contains the same number of elements the π^{-j} does. The elements in the unit vector $\vec{u}(a^{-j})$ are all equal to zero except for the element corresponding to the action a^{-j} which is equal to 1.

In this work, we are proposing a simple algorithm that uses the EMA approach. This algorithm is called the EMA Q-learning algorithm. The proposed algorithm uses the exponential moving average (EMA) mechanism as a basis to update the strategy of the agent itself. Furthermore, it uses two different variable learning rates η_w and η_l when updating

Algorithm 1 The exponential moving average (EMA) Q-learning algorithm for agent j:

Initialize:
learning rates $\theta \in (0,1]$, η_l and $\eta_w \in (0,1)$
constant gain k
exploration rate ϵ
discount factor ζ
$Q^j(s,a) \leftarrow 0$ and $\pi^j(s) \leftarrow$ ICs (arbitrarily)
Repeat
(a) At the state s, select an action a according to the strategy $\pi^j_t(s)$ with some exploration.
(b) Observe the immediate reward r^j and the new state s'.
(c) Update $Q^j_{t+1}(s,a)$ using Equation (2).
(d) Update the strategy $\pi^j_{t+1}(s)$ by using Equation (3).

the agent's strategy instead of only one constant learning rate η used in [7], [16]. The values of these variable learning rates are inversly proportional to the number of iterations. The recursive Q-learning algorithm for a learning agent j is given by the following equation:

$$Q^j_{t+1}(s,a) = (1-\theta)Q^j_t(s,a) + \theta(r^j + \zeta \max_{a'} Q^j_t(s',a')) \quad (2)$$

The EMA Q-learning algorithm updates the strategy of the agent j by Eq (3); whereas Algorithm 1 lists the whole formal procedure of the EMA Q-learning algorithm for a learning agent j.

$$\pi^j_{t+1}(s) = (1-k\eta)\pi^j_t(s) + k\eta\vec{u}(a) \quad (3)$$

where,
k is a constant gain and $k\eta \in (0,1)$.

$$\eta = \begin{cases} \eta_w & \text{if } a = \text{argmax}_{a'} Q^j_t(s,a') \\ \eta_l & \text{otherwise} \end{cases}$$

$$\vec{u}(a) = \begin{cases} \vec{u}_1(a) & \text{if } a = \text{argmax}_{a'} Q^j_t(s,a') \\ \vec{u}_2(a) & \text{otherwise} \end{cases}$$

$\vec{u}_1(a)$ is a unit vector representation of the action a with zero elements except for the element corresponding to the action a which is equal to one. This is to make the EMA Q-learning learn fast when the action a chosen by the agent j at the state s is equal to the greedy action obtained from the agent's Q-table at the state s. On the other hand, $\vec{u}_2(a) = \frac{1}{|A_j|-1}[\vec{1} - \vec{u}_1(a)]$. This is to make the EMA Q-learning learn cautiously and increase the opportunity of exploring the other agent's actions when the chosen action of agent j and the greedy action obtained from the agent's Q-table at the state s are different. Assume that, for example, the agent j has four possible actions at each state and the action a chosen by the agent j at the state s is the third action. The vector $\vec{u}(a)$ will be defined in this case as $\vec{u}(a) = \vec{u}_1(a) = [0,0,1,0]$ if the greedy action obtained from the agent's Q-table at the state s is also the third action. On the other hand, the vector $\vec{u}(a)$ will be defined as $\vec{u}(a) = \vec{u}_2(a) = [1/3, 1/3, 0, 1/3]$ if the greedy action obtained from the agent's Q-table at the state s is not the third action.

IV. Simulation and Results

We have evaluated the EMA Q-learning, WoLF-PHC [3], GIGA-WoLF [2], WPL [6], and PGA-APP [17] algorithms on a variety of matrix and stochastic games. Due to page limitations, we only show the EMA Q-learning, the PGA-APP and the WPL algorithms. The WoLF-PHC and GIGA-WoLF algorithms have been shown and discussed in [2], [6]. The results of applying the WPL, the PGA-APP and the EMA Q-learning algorithms to different matrix and stochastic games are presented in this section. A comparison among the three algorithms in terms of the convergence to Nash equilibrium is provided. We use the same learning and exploration rates for all algorithms when they are applied to the same game. In some cases, these rates are chosen to be close to those rates used in [17]. In other cases, on the other hand, the values of these rates are chosen based on trial and error basis to achieve the best performance of all algorithms.

A. Matrix Games

The EMA Q-learning, PGA-APP and WPL algorithms are applied to the matrix games depicted in Figure 1. Figure 4 shows the probability distributions of the second actions for both players in the dilemma game. The EMA Q-learning, the PGA-APP, and the WPL algorithms are shown. In this game, the parameters of the EMA Q-learning algorithm are set as follows: $\eta_w = \frac{1}{20+i/25}$, $\eta_l = 0.01\eta_w$, $k = 1$, $\zeta = 0.2$, and $\theta = 0.05$ with an exploration rate $\epsilon = 0.05$. The parameter γ is set as $\gamma = 0.5$ in the PGA-APP algorithm and the learning rate η is decayed with a slower rate in the WPL algorithm and is set as $\eta = \frac{1}{20+i/600}$. Figure 5 shows the probability distributions of the first actions for both players in the biased game while learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. In this game, the parameters of the EMA Q-learning algorithm are set as follows: $\eta_w = \frac{1}{10+i/5}$, $\eta_l = 0.01\eta_w$, $k = 1$, $\zeta = 0.95$, and $\theta = 0.8$ with an exploration rate $\epsilon = 0.05$. In the PGA-APP algorithm, the values of ζ and γ are set as follows: $\zeta = 0$ and $\gamma = 3$. In the WPL algorithm, the parameter ζ is set as $\zeta = 0$ and the learning rate η is decayed with a slower rate and is set as $\eta = \frac{1}{10+i/350}$. As shown in Figure 4, the players' strategies successfully converge to the Nash equilibrium in the dilemma game when learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. On the other hand, Figure 5 shows that both of the PGA-APP and the WPL algorithms fail to converge to a Nash equilibrium in the biased game; only the EMA Q-learning algorithm succeeds to converge to a Nash equilibrium in the biased game.

B. Stochastic Games

It is important to mention here that we only concern about the first movement of both players from the initial state. Therefore, the figures that will be shown in this section will represent the probabilities of players' actions at the initial state.

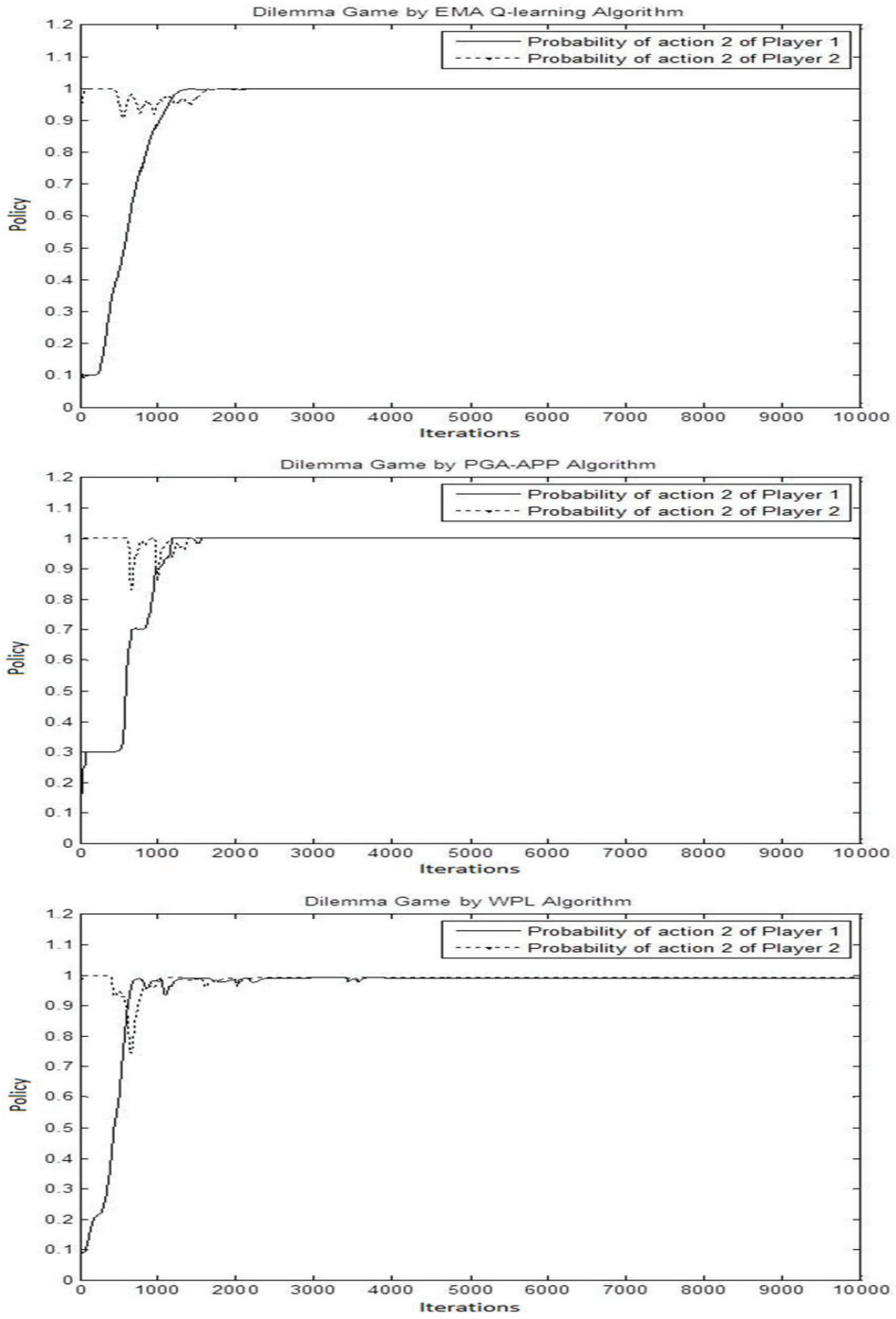

Fig. 4. Probability distributions of the second actions for both players in the dilemma game. The EMA Q-learning, the PGA-APP, and the WPL algorithms are shown. (Averaged over 10 runs)

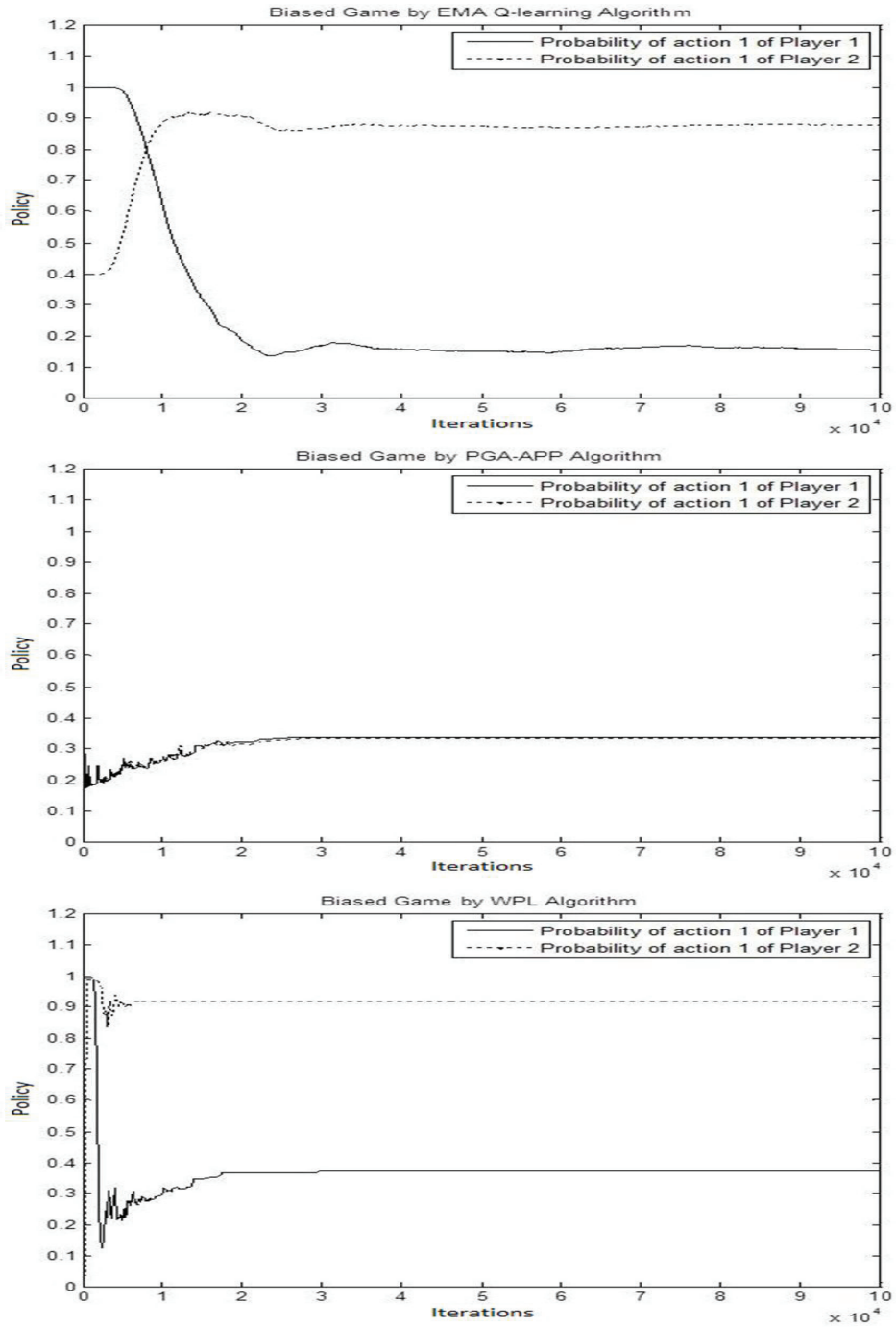

Fig. 5. Probability distributions of the first actions for both players in the biased game. The EMA Q-learning, the PGA-APP, and the WPL algorithms are shown. (Averaged over 10 runs)

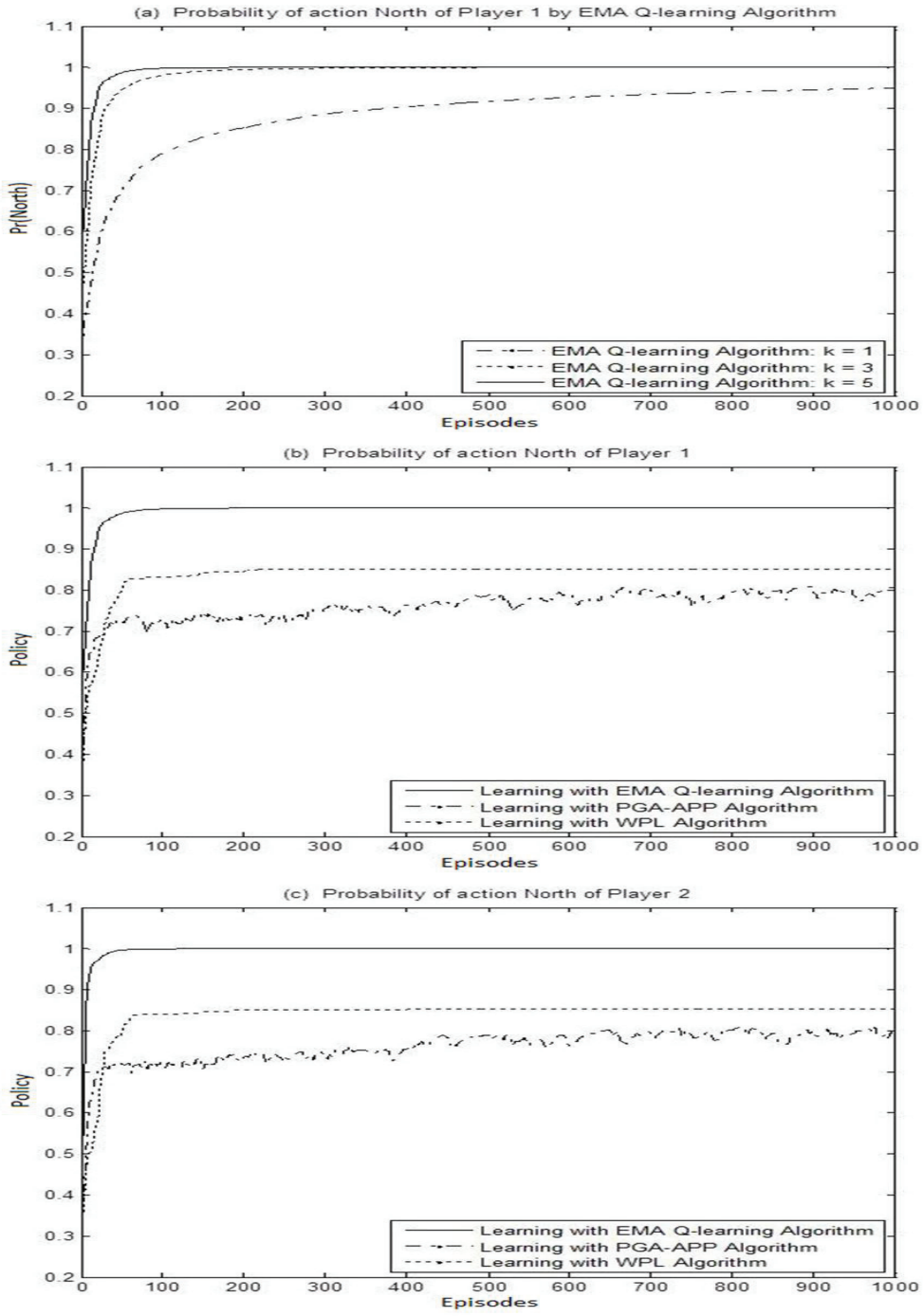

Fig. 6. Grid game 1: (a) Probability of action North of Player 1 when learning with the EMA Q-learning algorithm with different values of the constant gain k. Plots (b) and (c) illustrate the probability of action North of Player 1 and Player 2, respectively, when learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. (Averaged over 10 runs)

1) Grid Game 1: The EMA Q-learning, PGA-APP and WPL algorithms are used to learn the grid game 1 depicted in Figure 2(a). Grid game 1 has ten different Nash equilibria [11]. One of these Nash equilibria (optimal policy paths) is shown in Figure 3(a). Figure 3(a) shows that the action North is the optimal action for both players when they are at the initial state. The learning and exploration rates used by all algorithms are the same. The parameters of the EMA Q-learning algorithm are set as follows: $\eta_w = \frac{1}{10+i}$, $\eta_l = 0.001\eta_w$, $k = 5$, $\zeta = 0$, and $\theta = 0.8$ with an exploration rate $\epsilon = \frac{1}{1+0.001i}$, where i is the current number of episodes. The values of the parameters of the PGA-APP algorithm are the same as those of the EMA Q-learning algorithm except that $\gamma = 3$ and η has a very slow decaying rate, $\eta = \frac{1}{10+i/5000}$. The WPL algorithm also has the same parameters as the EMA Q-learning algorithm except that the learning rate η has a very slow decaying rate of $\eta = \frac{1}{10+i/5000}$.

Figure 6(a) shows the probability of selecting action North by Player 1 at the initial state when learning with the EMA Q-learning algorithm with different values of the constant gain k. Player 2 has similar probability distributions when learning with the EMA Q-learning algorithm with different values of the constant gain k. Figure 6(a) shows that Player 1's speed of convergence to the optimal action (North) increases as the value of the constant gain k increases. Figure 6(a) shows that the probability of selecting the optimal action North by Player 1 requires almost 80 episodes to converge to one when k=5 and 320 episodes when k=3. However, when $k = 1$, many more episodes are still required for the probability of selecting the action North to converge to one. Figure 6(b) and Figure 6(c) show the probabilities of taking action North at the initial state by both players when learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. Figure 6(b) and Figure 6(c) show that the probabilities of taking action North by both players converge to the Nash equilibrium (converge to one) when learning with the EMA Q-learning algorithm. However, the PGA-APP and the WPL algorithms fail to make the players' strategies converge to the Nash equilibria. Figure 6 shows that the EMA Q-learning algorithm outperforms the PGA-APP and the WPL algorithms in terms of the convergence to Nash equilibria. It also shows that the EMA Q-learning algorithm can converge to Nash equilibria with a small number of episodes by adjusting the value of the constant gain k. This will give the EMA Q-learning algorithm an empirical advantage over the PGA-APP and the WPL algorithms.

2) Grid Game 2: The EMA Q-learning, the PGA-APP and the WPL algorithms are also used to learn the grid game 2 depicted in Figure 2(b). Grid game 2 has two Nash equilibria [11]. Figure 3(b) shows one of these Nash equilibria. It is apparent from this particular Nash equilibrium that the action North is the optimal action for Player 1 at the initial state; whereas the action West is the optimal action for Player 2. Thus, for the algorithms to converge to this particular Nash equilibrium, the probability of selecting the action North by Player 1 should converge to one. The probability of selecting the action West by Player 2, on the other hand, should also converge to one. The learning and exploration rates used by all algorithms are the same. The parameters of the EMA Q-learning algorithm are set as follows: $\eta_w = \frac{1}{10+i}$, $\eta_l = 0.001\eta_w$, $k = 10$, $\zeta = 0.1$, and $\theta = \frac{1}{1+0.001i}$ with an exploration rate $\epsilon = \frac{1}{1+0.001i}$, where i is the current number of episodes. The values of the parameters of the PGA-APP algorithm are the same as those of the EMA Q-learning algorithm except that $\gamma = 3$, $\zeta = 0$ and η has a very slow decaying rate of $\eta = \frac{1}{10+i/5000}$. The WPL algorithm also has the same parameters as the EMA Q-learning algorithm except that $\zeta = 0$ and η has a very slow decaying rate of $\eta = \frac{1}{10+i/5000}$.

Figure 7(a) shows the probability of selecting action North by Player 1 when learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. Figure 7(a) illustrates that the probability of selecting the action North by Player 1 successfully converges to one (Nash equilibrium) when Player 1 learns with the EMA Q-learning algorithm. However, the PGA-APP and the WPL algorithms fail to make Player 1 choose the action North with a probability of one. Figure 7(b) shows the probability of selecting action West by Player 2 when learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. As can be seen from Figure 7(b), the probability of selecting action West by Player 2 successfully converges to one (Nash equilibrium) when Player 2 learns with the EMA Q-learning algorithm. The PGA-APP and the WPL algorithms, on the other hand, fail to make Player 2 choose action West with a probability of one. Figure 7 shows that the EMA Q-learning algorithm outperforms the PGA-APP and the WPL algorithms in terms of the convergence to Nash equilibria. This will give the EMA Q-learning algorithm an empirical advantage over the PGA-APP and the WPL algorithms.

V. Conclusion

A new multi-agent policy iteration learning algorithm is proposed in this work. It uses the exponential moving average (EMA) estimation technique to update the players' strategies. This policy iteration learning algorithm is called the EMA Q-learning algorithm. This algorithm is applied to a variety of matrix and stochastic games. The results show that the value of the constant gain k affects the speed of convergence of players' strategies to Nash equilibria. The results also show that the EMA Q-learning algorithm outperforms the PGA-APP and the WPL algorithms in terms of the convergence to Nash equilibria. The results also show that the EMA Q-learning algorithm can converge to Nash equilibria with a small number of episodes by adjusting the value of the constant gain k. This will give the EMA Q-learning algorithm an empirical advantage over the PGA-APP and the WPL algorithms.

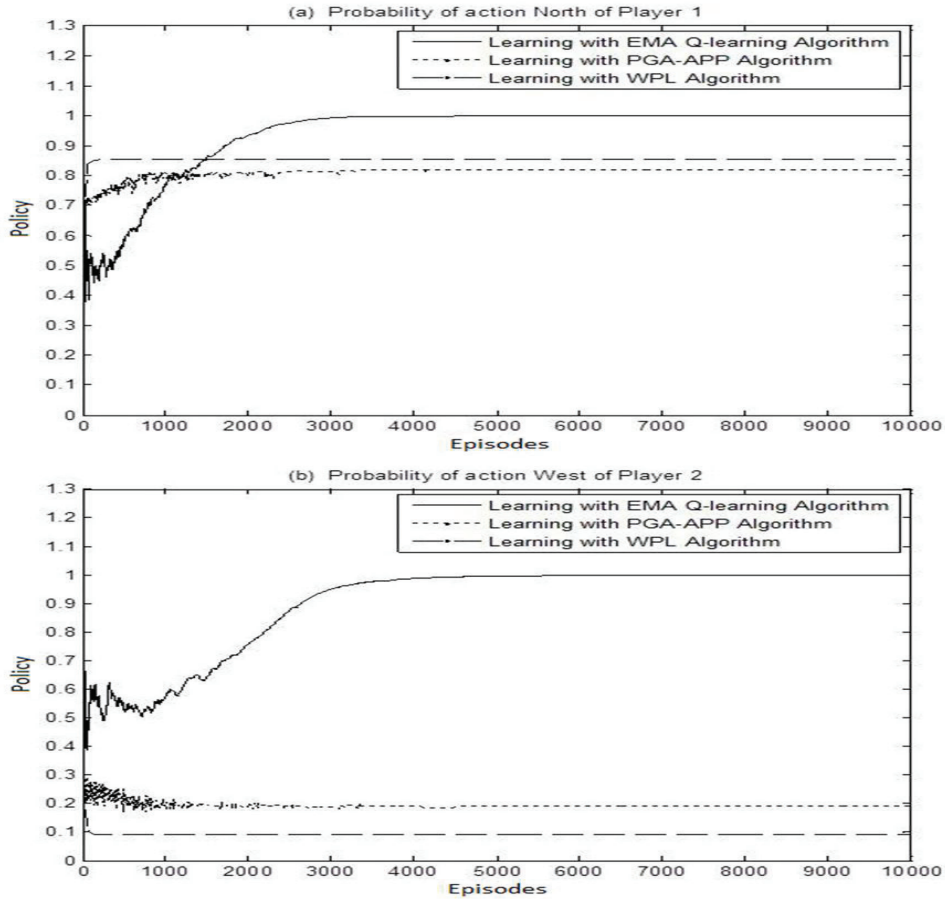

Fig. 7. Grid game 2: (a) The probability of selecting action North by Player 1 when learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. (b) The probability of selecting action West by Player 2 when learning with the EMA Q-learning, the PGA-APP, and the WPL algorithms. (Averaged over 10 runs)

REFERENCES

[1] R. Bellman, *Dynamic Programming*, Princeton University Press, 1957.

[2] M. Bowling, *Convergence and no-regret in multiagent learning*, In Advances in Neural Information Processing Systems 17. MIT Press, 2005.

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

[4] L. Buşoniu and R. Babuška and B. D. Schutter, *Multiagent reinforcement learning: A survey*, 9th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 1-6, 2006.

[5] L. Buşoniu and R. Babuška and B. D. Schutter, *A comprehensive survey of multiagent reinforcement learning*, IEEE Trans Syst Man Cybern C, 38(2):156-172, 2008.

[6] S. Abdallah and V. Lesser, *A multiagent reinforcement learning algorithm with non-linear dynamics*, Journal of Artificial Intelligence Research 33, pp. 521-549, 2008.

[7] A. Burkov and B. Chaib-draa, *Effective learning in the presence of adaptive counterparts*, Journal of Algorithms, 64(4):127-138, 2009.

[8] V. Conitzer and T. Sandholm, *Awesome: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents*, Machine Learning, 67(1-2), pp. 23-43, 2007.

[9] R. A. Howard, *Dynamic Programming and Markov Processes*, MIT Press, 1960.

[10] J. Hu and M. P. Wellman, *Multiagent reinforcement learning: Theoretical framework and an algorithm*, In proceedings fifteenth international conference on machine learning, pp. 242-250, 1998.

[11] J. Hu and M. P. Wellman, *Nash q-learning for general-sum stochastic games*, Journal of Machine Learning Research, 4:1039-1069, 2003.

[12] L. P. Kaelbling and M. L. Littman and A. W. Moore, *Reinforcement learning: A survey*, Journal of Artificial Intelligence Research, 4:237-285, 1996.

[13] S. Sen and G. Weiss, *Learning in multiagent systems*, In: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. The MIT Press, Cambridge, 1999.

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

[15] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, The MIT Press, Cambridge, Massachusetts, 1998.

[16] G. Tesauro, *Extending q-learning to general adaptive multi-agent systems*, In Advances in Neural Information Processing Systems 16, pp. 215-250, 2004.

[17] C. Zhang and V. Lesser, *Multi-agent learning with policy prediction*, In Proceedings of the 24th National Conference on Artificial Intelligence (AAAI10), Atlanta, GA, USA, pp. 746-752, 2010.