An investigation of guarding a territory problem in a grid world

Xiaosong Lu and Howard M. Schwartz

Abstract—A game of guarding a territory in a grid world is proposed in this paper. A defender tries to intercept an invader before he reaches the territory. Two reinforcement learning algorithms are applied to make two players learn their optimal policies simultaneously. Minimax-Q learning algorithm and Win-or-Learn-Fast Policy Hill-Climbing learning algorithm are introduced and compared. Simulation results of two reinforcement learning algorithms are analyzed.

I. INTRODUCTION

The game of guarding a territory was first introduced by Isaacs [1]. In the game, the invader tries to move to the territory as close as possible while the defender tries to intercept and keep the invader away from the territory as far as possible. The practical application of this game can be found in surveillance and security missions for autonomous mobile robots. There are few published works in this field since the game was introduced [2], [3]. In these published works, the defender tries to use a fuzzy controller to locate the invader's position [2] or applies a fuzzy reasoning strategy to capture the invader [3]. However, in these works, the defender is assumed to know his optimal policy and the invader’s policy. There is no learning technique applied to the players in their works. In our research, we assume the defender or the invader has no prior knowledge of his optimal policy and the opponent’s policy. We will apply learning algorithms to the players and let the defender or the invader obtain his own optimal behavior after learning.

The problem of guarding a territory in [1] is a differential game problem where the dynamic equations of the players are typically differential equations. In our research, we will investigate how the players learn to behave with no knowledge of the optimal policies. Therefore, the above problem becomes a multi-agent learning problem in a multiagent system. In the literature, there are large amount of published papers on multiagent systems [4], [5]. Among the multiagent learning applications, the predator-prey or the pursuit problem in a grid world has been well studied [6], [5]. To better understand the learning process of the two players in the game, we will create a grid game of guarding a territory which has never been studied so far.

The main contributions of this paper include establishing a grid game of guarding a territory and applying two multiagent learning algorithms to the game. Most of multi-agent learning algorithms are based on multi-agent reinforcement learning (MARL) methods [5]. According to the definition of the game in [1], the grid game we established will be a two-player zero-sum stochastic game. The conventional minimax-Q learning algorithm [7] will be well suited to solving our problem. However, if the player does not always take the action that is most damaging the opponent, the opponent might have better performance using a learning method than the minimax-Q learning [4]. This learning method is called Win-or-Learn-Fast Policy Hill-Climbing (WoLF-PHC) learning algorithm [8]. In this paper, we will discuss both of the MARL algorithms and compare their learning performance.

The paper is organized as follows. Section II introduces the game of guarding a territory. In this section, we will build the game in a grid world and make it a test bed for the aforementioned learning algorithms. Section III introduces the background of stochastic games. In section IV, we introduce the minimax-Q learning algorithm. We will apply this algorithm to both the defender and the invader and let the two players learn their optimal policies simultaneously. To compare with the minimax-Q learning method, another MARL algorithm called WoLF-PHC is shown in section V. Simulation results and the comparison of these two learning algorithms are presented in section VI. Section VII is our conclusions.

II. GUARDING A TERRITORY PROBLEM

The problem of guarding a territory in this paper is the grid version of the guarding a territory game in [1]. The game is defined as follows:

- We take a $6 \times 6$ grid as the playing field shown in Fig. 1. The invader starts from the upper-left corner and tries to reach the territory before the capture. The territory is represented by a cell named T in Fig. 1. The defender starts from the bottom and tries to intercept the invader. The initial positions of the players are not fixed and can be chosen randomly.
- Both of the players can move up, down, left or right. At each time step, both players take one action and move to an adjacent cell simultaneously. If the chosen action will take the player off the playing field, the player will stay at the current position.
- The nine gray cells centered around the defender, shown in Fig. 1(b), is the region where the invader will be captured. A successful invasion by the invader is defined in the situation where the invader reaches the territory before the capture or the capture happens at the territory. The game ends when the defender captures the invader or a successful invasion by the invader happens. Then
a new trial starts with random initial positions of the players. The goal of the invader is to reach the territory without interception or move to the territory as close as possible if the capture must happen. On the contrary, the aim of the defender is to intercept the invader at a location as far as possible to the territory.

The terminal time is defined as the time when the invader reaches the territory or is intercepted by the defender. We define the payoff as the distance between the invader and the territory at the terminal time [3]:

\[
\text{Payoff} = |x_I(t_f) - x_T| + |y_I(t_f) - y_T|
\]

where \((x_I(t_f), y_I(t_f))\) is the invader’s position at the terminal time \(t_f\) and \((x_T, y_T)\) is the territory’s position. Based on the definition of the game, the invader tries to minimize the payoff while the defender tries to maximize the payoff.

III. STOCHASTIC GAMES

Reinforcement learning (RL) does not require the model of the environment and agents can take actions while they learn [9]. For a single-agent reinforcement learning, the environment of the agent can be described as a Markov Decision Process (MDP) [4]. A Markov Decision Process (MDP) is a tuple \((S,A,T,R)\) where \(S\) is the state space, \(A\) is the action space, \(T: S \times A \rightarrow \text{PD}(S)\) is the transition function and \(R: S \times A \rightarrow \mathbb{R}\) is the reward function. The transition function denotes a probability distribution over next states given the current state and action. The reward function denotes the received reward after the given action and state [8]. To solve a MDP, we need to find a policy \(\pi: S \rightarrow A\) mapping states to actions. An optimal policy will maximize the discounted future reward with a discount factor \(\gamma\). A conventional reinforcement learning method to solve a MDP is called Q-learning [10]. Q-learning is a model-free reinforcement learning method. Using Q-learning, the agents can learn online to act optimally without knowing the model of the environment.

The learning procedure of the Q-learning algorithm is given as [11]

1) Initialize \(Q(s,a)\) where \(Q(s,a)\) is an approximation of \(Q^*(s,a)\). \(Q^*(s,a)\) is defined as the expected discounted future reward given the current state \(s\) and action \(a\) and following the optimal policy after that.

2) Repeat
   a) Select action \(a\) from current state \(s\) based on mixed exploration-exploitation strategy.
   b) Take action \(a\) and observe the reward \(r\) and the subsequent state \(s'\).
   c) Update \(Q(s,a)\)

\[
Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]
\]

where \(\alpha\) is the learning rate and \(\gamma\) is the discount factor.

For a game with more than one agent, the MDP is extended to a stochastic game. A stochastic game is a tuple \((n,S,A_1,...,A_n,T,R_1,...,R_n)\) where \(n\) is the number of the players, \(T: S \times A_1 \times \cdots \times A_n \rightarrow \text{PD}(S)\) is the transition function, \(A_i(i = 1,...,n)\) is the action set for the player \(i\) and \(R_i: S \times A_1 \times \cdots \times A_n \rightarrow \mathbb{R}\) is the reward function for the player \(i\). The transition function in a stochastic game is a probability distribution over next states given the current state and the joint action of the players. The reward function for the player \(i\) in a stochastic game denotes the received reward after the given joint action and the current state. To solve a stochastic game, we need to find a policy \(\pi_i: S \rightarrow A_i\) that can maximize the player \(i\)’s discounted future reward with a discount factor \(\gamma\) [8].

The stochastic game can be classified as a fully cooperative game, a fully competitive game and a mixed game. If all the players have the same objective, the game is called a fully cooperative game. If one player’s reward function is always the opposite sign of the other player’s, the game is called a two-player fully competitive or zero-sum game. When some of the players are cooperative and others are competitive, the game is called a mixed game. The grid game of guarding a territory in Fig. 1 is a two-player zero-sum game since the invader and the defender have completely conflicting interests.

For a two-player zero-sum stochastic game, we can find a unique Nash equilibrium [12]. To solve a two-player zero-sum stochastic game and find the Nash equilibrium, one can use a multi-agent learning algorithm to learn the Nash equilibrium policy for each player. Unlike the deterministic optimal policy in MDPs, the Nash equilibrium policy of each player in stochastic games may be stochastic [7]. In order to study the performance of multi-agent learning algorithms, we define the following two criteria [4]:

- Stability: the convergence to a stationary policy. For a two-player zero-sum stochastic game, the two players’ policies should converge to a Nash equilibrium.
- Adaptation: If one player changes his policy to a different stationary policy, the other player will adapt to the change and learn a best response policy to the opponent’s new policy.

Among multi-agent learning methods, MARL methods have received considerable attention in the literature [4]. For the grid game of guarding a territory, we present two MARL methods in this paper.
IV. MINIMAX-Q LEARNING

Littman [7] proposed a minimax-Q learning algorithm specifically designed for the two-player zero-sum stochastic game. The minimax-Q learning algorithm can guarantee that one player’s policy will converge to a best response policy against the worst possible opponent.

We define the value of the state as the expect reward for the optimal policy starting from state $s$ [7]

$$V(s) = \max_{\pi \in \Pi(D)} \min_{o \in O} \sum_{a \in A} Q(s,a,o)\pi_a$$

(2)

where $Q(s,a,o)$ is the expected reward when the player and his opponent choose action $a \in A$ and $o \in O$ respectively and follow the optimal policies after that. The player’s policy $\pi$ is a mixed policy chosen from the player’s action space $A$. The reason of using a mixed policy is that any deterministic policy can be completely defeated by the opponent in the stochastic game [7].

Given $Q(s,a,o)$ in (2), we can solve equation (2) and find the player’s best response policy $\pi^*$, Littman uses linear programming to solve equation (2). Since $Q(s,a,o)$ is unknown to the player in the game, an updating rule similar to the Q-learning algorithm in Section III is applied.

The whole learning procedure of minimax-Q learning is listed as follows:

1) Initialize $Q(s,a,o)$, $V(s)$ and $\pi(s,a)$
2) Repeat
   a) Select action $a$ from current state $s$ based on mixed exploration-exploitation strategy
   b) Take action $a$ and observe the reward $r$ and the subsequent state $s'$
   c) Update $Q(s,a,o)$

   $$Q(s,a,o) \leftarrow Q(s,a,o) + \alpha [r + \gamma V(s') - Q(s,a,o)]$$

   where $\alpha$ is the learning rate and $\gamma$ is the discount factor
   d) Use linear programming to solve equation (2) and obtain $\pi(s,a)$ and $V(s)$

The minimax learning algorithm can guarantee the convergence to the Nash equilibrium if all states and actions are visited infinitely often. The proof of convergence for minimax learning algorithm can be seen in [13]. However, the execution of linear programming at each iteration will slow down the learning process.

Using the minimax-Q learning algorithm, the player will always play a “safe” policy in case of the worst scenario caused by the opponent. However, if the opponent is not playing his best, the minimax-Q learning method cannot make the player adapt his policy to the change in the opponent’s policy. The reason is that the minimax-Q learning method is an opponent-independent policy and it will converge to the Nash equilibrium policy no matter what policy the opponent uses. The Nash equilibrium policy is not a best response policy against a weak opponent. In other words, the best response policy will do better than the Nash equilibrium policy in this case. Therefore, the minimax-Q learning algorithm does not satisfy the adaptation criterion introduced in Section III. In the next section, we will introduce another MARL method satisfying both the stability and adaptation criteria.

V. WOLF POLICY HILL-CLIMBING LEARNING

The Win or Learn Fast policy hill-climbing (WoLF-PhC) learning algorithm is an extension of the minimax-Q learning method. The WoLF-PhC algorithm is an opponent-aware algorithm that can improve the player’s policy based on the opponent’s behavior. With the use of a varying learning rate, the convergence of the player’s policy is guaranteed in this algorithm. Therefore, both the stability and adaptation criteria are achieved in [8].

The whole learning procedure of the WoLF-PhC learning method is listed as follows [8]:

1) Initialize $Q(s,a) \leftarrow 0$, $\pi(s,a) \leftarrow 1/|A|$ and $C(s) \leftarrow 0$. Choose the learning rate $\alpha$, $\delta$ and the discount factor $\gamma$
2) Repeat
   a) Select action $a$ from current state $s$ based on mixed exploration-exploitation strategy
   b) Take action $a$ and observe the reward $r$ and the subsequent state $s'$
   c) Update $Q(s,a)$

   $$Q(s,a) \leftarrow Q(s,a) + \alpha [r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

   d) Update the estimate of average policy $\bar{\pi}$

   $$C(s) \leftarrow C(s) + 1$$

   $$\bar{\pi}(s,a') \leftarrow \bar{\pi}(s,a') + \frac{1}{C(s)} (\pi(s,a') - \bar{\pi}(s,a'))$$

   ($\forall a' \in A_i$)

   e) Step $\pi(s,a)$ closer to the optimal policy

   $$\pi(s,a) \leftarrow \pi(s,a) + \Delta_{sa}$$

   where

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

   $$\delta_{sa} = \min (\pi(s,a), \frac{\delta}{|A| - 1})$$

   $$\delta = \begin{cases} \delta_{sa} & \text{if } \sum_{a'} \pi(s,a')Q(s,a') > \sum_{a'} \bar{\pi}(s,a')Q(s,a') \\ \delta_{sa} & \text{otherwise} \end{cases}$$

The WoLF-PhC algorithm is the combination of the two methods: Win or Learn Fast method and policy hill-climbing method. The policy hill-climbing (PhC) method is a policy adaptation method. It improves the agent’s policy by increasing the probability of selecting the action $a$ with a highest value of $Q(s,a)(\forall a \in A)$ [8]. However, the convergence to the Nash equilibrium in non-stationary environments has not been shown in the PhC method [8].

To deal with the stability issue, the Win or Learn Fast (WoLF) method is added to the algorithm. The WoLF method changes the learning rate $\delta$ based on the winning or losing
situation. The learning rate $\delta_l$ for the losing situation is larger than the learning rate $\delta_w$ for the winning situation. If the player is losing, he should learn quickly to escape from the losing situation. If the player is winning, he should learn cautiously to guarantee the convergence of the policy. The proof of convergence to the Nash equilibrium for the WoLF-PHC method is shown in [8]. Combining the WoLF method with the PHC method, the WoLF-PHC algorithm can meet the requirement of the stability and adaptation criteria. In the next section, we will apply both the minimax-Q learning and WoLF-PHC learning algorithms to the grid game of guarding the territory in simulation.

VI. SIMULATION AND RESULTS

Now we use the minimax-Q learning and WoLF-PHC learning algorithms introduced in Section IV and V to simulate the grid game of guarding a territory. We first present a simple $2 \times 2$ grid game to explore the issues of mixed policy, stability and adaptation. These issues are discussed in the previous sections and we will compare both of the learning algorithms based on these issues. Next, the playing field is enlarged to a $6 \times 6$ grid and we will examine the performance of the algorithms for this large grid.

We set up two simulations for each grid game. In the first simulation, we apply the minimax-Q learning algorithm or the WoLF-PHC algorithm to both players and let both the invader and the defender learn their behaviors simultaneously. After learning, we test the performance of the minimax-Q trained policy against the WoLF-PHC trained policy. In the second simulation, we will fix one player’s policy and let the other player learn the best response policy against his opponent. The aforementioned two algorithms will be applied to train the learner individually. According to the discussion in Section IV and Section V, we will expect the defender with the WoLF-PHC trained policy has better performance than the defender with the minimax-Q trained policy in the second simulation.

A. $2 \times 2$ Grid Game

The playing field of the $2 \times 2$ grid game is shown in Fig. 2. The territory to be guarded is located at the bottom-right corner. The invader will start at the top-left corner while the defender will start at the same cell as the territory. To better illustrate the guarding a territory problem, we simplify the possible actions of each player from 4 actions to 2 actions. The invader can only move down or right while the defender can only move up or left. The capture of the invader happens when the defender and the invader move into the same cell excluding the territory cell. The game ends when the invader reaches the territory or the defender catches the invader before he reaches the territory. We suppose both players start from the initial state $s_1$ shown in Fig. 2(a). There are three nonterminal states $(s_1, s_2, s_3)$ in this game shown in Fig. 2. If the invader moves to the right cell and the defender happens to move left, then both players reach the state $s_2$ in Fig. 2(b). If the invader moves down and the defender moves up simultaneously, then they will reach the state $s_3$ in Fig. 2(c).

![Initial positions of the players: State $s_1$ right vs. defender in left vs. defender in bottom-left: State $s_2$ top-right: State $s_3$.](image)

Fig. 2. A $2 \times 2$ Grid Game

In states $s_2$ and $s_3$, if the invader is smart enough, he can always reach the territory no matter what action the defender will take. Therefore, starting from the initial state $s_1$, a clever defender will try to intercept the invader by guessing which direction the invader will go. In the $2 \times 2$ grid game, we will apply the aforementioned two algorithms to the players and let both players learn their Nash equilibrium policies online.

We first define the reward functions for the players. The reward function for the defender is defined as follows:

$$R_D = \begin{cases} dist_{IT}, & \text{defender captures the invader;} \\ -10, & \text{invader reaches the territory.} \end{cases}$$

where

$$dist_{IT} = |x(t_f) - x_T| + |y(t_f) - y_T|.$$  

The reward function for the invader is given by:

$$R_I = \begin{cases} -dist_{IT}, & \text{defender captures the invader;} \\ 10, & \text{invader reaches the territory.} \end{cases}$$

The reward functions in (3) and (4) will be the same for both the $2 \times 2$ grid game and the $6 \times 6$ grid game.

Before the simulation, we can simply solve this game using the minimax principle introduced in (2). In the states $s_2$ and $s_3$, a smart invader will always reach the territory without being intercepted. The value of the states $s_2$ and $s_3$ for the defender will be $V_D(s_2) = -10$ and $V_D(s_3) = -10$. We set the discount factor as $\gamma = 0.9$ and we can get $Q_D(s_1, a_{up}, o_{down}) = \gamma V_D(s_2) = -9$, $Q_D(s_1, a_{up}, o_{right}) = \gamma V_D(s_3) = -9$, $Q_D(s_1, a_{left}, o_{down}) = 1$ and $Q_D(s_1, a_{up}, o_{right}) = 1$, as shown in Table I(a). The probabilities of the defender moving up and left are denoted as $\pi_D^u(s_1, a_{up})$ and $\pi_D^l(s_1, a_{left})$ respectively. The probabilities of the invader moving down and right are denoted as $\pi_I^d(s_1, a_{down})$ and $\pi_I^r(s_1, a_{right})$ respectively. Based on the Q values in Table I(a), we can find the value of the state $s_1$ for the defender by solving a linear programming problem shown in Table I(b). Further explanation can be found in [7]. After solving the linear constraint in Table I(b), we can get the value of the state $s_1$ for the defender as $V_D(s_1) = -4$ and the Nash equilibrium policy for the defender as $\pi_D^*(s_1, a_{up}) = 0.5$ and $\pi_D^*(s_1, a_{left}) = 0.5$. For a two-player zero-sum game, we can get $Q_D^* = -Q_I^*$. Similar to the approach in Table I, we can find the minimax solution of this game for the invader as $V_I(s_1) = 4$, $\pi_I^d(s_1, a_{down}) = 0.5$ and $\pi_I^r(s_1, a_{right}) = 0.5$.

We now apply the minimax-Q learning algorithms to the game. To better examine the performance of the minimax-Q
TABLE I
MINIMAX SOLUTION FOR THE DEFENDER IN THE STATE s1

(a) Q values of the defender for the state s1

<table>
<thead>
<tr>
<th>Defender</th>
<th>Invader</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qb</td>
<td>Qd</td>
</tr>
<tr>
<td>up</td>
<td>down</td>
</tr>
<tr>
<td>left</td>
<td>right</td>
</tr>
</tbody>
</table>

(b) linear constraints for the defender in the state s1

Objective: Maximize V

\[-9 \cdot \pi_D(s_1, a_{up}) + 1 = V\]
\[1 \cdot \pi_D(s_1, a_{up}) + (-9) \cdot \pi_D(s_1, a_{left}) = V\]
\[\pi_D(s_1, a_{up}) + \pi_D(s_1, a_{left}) = 1\]

learning algorithm, we will use the same parameter settings as in [7]. The exploration parameter is given as 0.2. The learning rate \(\alpha\) is chosen such that the value of the learning rate will decay to 0.01 after one million iterations. The discount factor \(\gamma\) is set to 0.9. We run the simulation for 100 iterations. The number of iterations represents the number of times the step 2 is repeated in minimax-Q learning procedure in Section IV. After learning, we plot the defender’s policy and the invader’s policy in Fig. 3. The result shows that both the defender and invader’s policies converge to the Nash equilibrium policy after 100 iterations. The Nash equilibrium policy of the invader for the 2 \(\times\) 2 grid game is moving down or right with probability 0.5 and the Nash equilibrium policy of the defender is moving up or left with probability 0.5.

We now apply the WoLF-PHC algorithm to the 2 \(\times\) 2 grid game. According to the parameter settings in [8], we set the learning rate \(\alpha\) as \(1/(10+\gamma)/10000\), \(\delta_u\) as \(1/(10+\gamma)/2\) and \(\delta_d\) as \(3/(10+\gamma)/2\) where \(t\) is the number of the current iteration. The number of iterations denotes the number of times the step 2 is repeated in WoLF-PHC learning procedure in Section V. The result in Fig. 4 shows that the policies of both players will converge to their Nash equilibrium policies after 15000 iterations. Using WoLF-PHC algorithm, the players will take more iterations than the minimax-Q learning algorithm to converge to the equilibrium policy.

In the second simulation, the invader will play a fixed policy against the defender at state s1 in Fig. 2(a). The invader will move right with probability 0.8 and move down with probability 0.2. In this situation, the best response policy for the defender will be moving up all the time. We apply both of the two algorithms to the game and examine the learning performance for the defender. Results in Fig. 5(a) show that, using the minimax-Q learning, the defender’s policy fails to converge to the best response policy in this 2 \(\times\) 2 grid game. Whereas, the WoLF-PHC learning method will guarantee the convergence to the best response policy against the invader, as shown in Fig. 5(b).

In the 2 \(\times\) 2 grid game, simulation results show that both of the two algorithms can achieve the convergence to the Nash equilibrium policy. Under the adaptation criterion, the minimax-Q learning method fails to show the convergence to the best response policy in Fig. 5(a). The WoLF-PHC learning method can satisfy both the convergence to the Nash equilibrium and adaptation to the best response policy in this game. One drawback for the WoLF-PHC learning algorithm that the learning process is slow when compared with the minimax-Q learning algorithm.

B. 6 \(\times\) 6 Grid Game

We now change the 2 \(\times\) 2 grid game to a 6 \(\times\) 6 grid game. The playing field of the 6 \(\times\) 6 grid game is defined in Section II. The territory to be guarded is represented by a cell located at (5, 5) in Fig. 6. The position of the territory will not be
changed during the simulation. The initial positions of the invader and defender are shown in Fig. 6(a). The number of actions for each player has been changed from 2 in the $2 \times 2$ grid game to 4 in the $6 \times 6$ grid game. Both players can move up, down, left or right. The grey cells in Fig. 6(a) is the area where the defender can reach before the invader. Therefore, in the worst case, the invader can move to the territory as close as possible with the distance of 2 cells shown in Fig. 6(b). After every 100 iterations, the learning performance of the algorithms will be tested using the currently learned policies. During each test, we play the game 1000 trials and average the final distance between the invader and the territory at the terminal time for each trial.

We use the same parameter settings as in the $2 \times 2$ grid game for the minimax-Q learning method. The result in Fig. 7(a) shows that the average distance between the invader and the territory will converge to 2 after 50000 iterations. Now we use the WoLF-PHC learning algorithm to simulate again. We set the learning rate $\alpha$ as $1/(4+\delta t/50)$, $\delta_n$ as $1/(1+\delta t/5000)$ and $\delta_l$ as $4/(1+\delta t/5000)$. We run simulation for 200000 iterations. The result in Fig. 7(b) shows that the WoLF-PHC learning method can also satisfy the convergence to the distance of 2.

In the second simulation, we will fix the invader’s policy to a random policy, which means that the invader can move up, down, left or right with equal probability. Similar to the first simulation, the learning performance of the algorithms will be tested using the currently learned policies for each 100 iterations. For each test, we play the game 1000 trials and the average distance between the invader and the territory at the terminal time are plotted.

The results are shown in Fig. 8(a) and 8(b). Using the WoLF-PHC learning method, the defender can intercept the invader further from the territory (distance of 6.6) than using the minimax learning method (distance of 5.9). Therefore, by comparing the results in Fig. 8(a) and 8(b), we can see the WoLF-PHC learning method can achieve better performance than the minimax-Q learning method based on the adaptation criterion in Section III.

VII. CONCLUSIONS

This paper proposes a grid game of guarding a territory. The invader and the defender try to learn to play against each other using multi-agent reinforcement learning algorithms. Among multi-agent reinforcement learning methods, the minimax-Q learning algorithm and WoLF-PHC learning algorithm are applied to the game. The comparison between these two algorithms are studied and illustrated in simulation results. Both the minimax-Q learning algorithm and the WoLF-PHC learning algorithm can guarantee the convergence to the players’ Nash equilibrium policies. Using
the WoLF-PHC learning method, one player’s policy can converge to the best response policy against his opponent. Since the learning process for the WoLF-PHC learning method is extremely slow, more efficient learning methods will be studied for the game in the future. The study of the grid game of guarding a territory for three or more players is also necessary in future research.

REFERENCES


