Multi-Agent Zero-Sum Soccer Game Experiment

Beixi Chen

Spring 2019
bchen372@gatech.edu
git hash: 39136aecc188637451180149619d2bdb1d285f4a

ABSTRACT

Multi-agent Q-learning algorithms attract attention of many scholars in the reinforcement learning field, in that in Markov Game context, convergence criteria rely on equilibrium being found within the payoff matrices of multiple agents. The numerical value in these matrices represent the motivations of different actors. Finding equilibrium in payoff metrics is difficult, this difficulty stem from the equilibrium selection problem: how do multi-agent select among multiple equilibria. In paper "Correlated Q-Learning" [1], Greenwald introduced *correlated Q-learning*, which generalizes both *Nash-Q* and *Friend-and-Foe-Q*. The purpose of this report is to replicate Greenwald's experiment of implementing classic *Q-learning* with $\epsilon-$greedy and other three variants of correlated Q-learning algorithms in a zero-sum soccer game.

I. ENVIRONMENT AND ALGORITHMS

A. Environment

The environment of the zero-sum soccer game is depicted in Fig.1 and can be described as follow: There are two players and either of both has possession of the circle ball. Four possible actions are N, S, E, W, and stick, but if a player attempts to move out of grid, it can only sticks to that position. The players' actions are executed in random order and the one being selected first moves first. **If the position a player tempts to go is already occupied by the other one, it cannot move to that position unless the other player goes to somewhere else.** If both players are attempting to go to the same place, then only the first can do that. If the player with the ball moves into the player without the ball, the former loses the ball to the latter. **If the player with the ball moves into a goal, then he scores $+100$ if it is in fact his own goal and the other player scores -100, or he scores -100 if it is the other player's goal and the other player scores $+100$.** In either case, the game ends. Note that building an environment consistent with the original paper is very important to replicate experiments in that a slightly difference from original rule could produce a wildly different picture. Moreover, text marked in bold would be discussed further in the pitfall part.

B. Multi-Agent Algorithms

1) General Template: The algorithm shown in Fig.2 was a full reference from Greenwald's paper "Correlated Q-Learning" [1]. Fig.2 did a great job in depicting a general template for all three multi-agent algorithms this article was about to implement, in that different variants of correlated-Q could be generated by only effecting step 3, saying, value function $V_i(s')$ and matrix-vector $\vec{Q} = (Q_1, Q_2, ..., Q_n)$, and such effect could be realized by changing the input parameter f, which is the selection function. As mentioned before in *Abstract*, a solution of multi-agent problem can be regarded as converged only when the equilibrium policy has been reached, different equilibrium selection methods would generate different equilibrium, hence different variants of correlated-Q learning. The following content would explain in detail what adaption to made to create different multi-agent Q-learning, and this article would implement these algorithms closely.

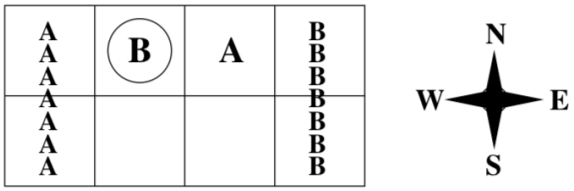

Fig. 1: Soccer Game. State s.

2) Correlated-Q: Differ from Nash equilibrium (NE), a vector of independent probability distributions over actions, correlated equilibrium (CE) permits dependencies among the players' actions. An example of CE in daily life is the traffic signal at an intersection, one signal (player)'s action is correlated with that of the other signal (player), eg. if one signal (player) is green, the other signal (player) should be red to avoid collision. Besides, CE can be computed easily via linear programming while NE cannot. [1] This property makes correlated-Q learning more desirable. Correlated-Q learning can be divided into four categories: utilitarian, egalitarian, republican, and libertarian. This article adhered to the Greenwald's paper and only implemented utilitarian correlated Q-learning, in which CE maximizes the sum of both agent's rewards.

3) Foe-Q and Friend-Q: Littman [2] took the concept of Nash equilibrium and further extrapolated it to adversarial and coordination equilibrium. [2] Adversarial equilibrium is designed for where agents attempt to maximize their own utility while minimizing their opponents'. In contrast, coordination

MULTIQ(MarkovGame, f, γ, α, S, T)

Inputs	selection function f
	discount factor γ
	learning rate α
	decay schedule S
	total training time T
Output	state-value functions V_i^*
	action-value functions Q_i^*
Initialize	$s, a_1, \ldots, a_n$ and $Q_1, \ldots, Q_n$

for $t = 1$ to T

1. simulate actions $a_1, \ldots, a_n$ in state s
2. observe rewards $R_1, \ldots, R_n$ and next state s'
3. for $i = 1$ to n
   (a) $V_i(s') = f_i(Q_1(s'), \ldots, Q_n(s'))$
   (b) $Q_i(s, \vec{a}) = (1-\alpha)Q_i(s, \vec{a}) + \alpha[(1-\gamma)R_i + \gamma V_i(s')]$
4. agents choose actions $a'_1, \ldots, a'_n$
5. $s = s'$, $a_1 = a'_1$, $\ldots$, $a_n = a'_n$
6. decay α according to S

Fig. 2: Multi-agent Q-Learning.

equilibrium are solutions where the agents aim to achieve a maximum utility for all players throughout the state-action set.

II. Implementation

A. Environment Setup

The soccer game field was represented by 2×4 array, with index started from value 1. The amount of states is $8 \times 8 \times 2 = 128$, each state s was represented by 3 values: coordinate of player A, coordinate of player B, whether A possesses the ball, respectively. Actions a available are $[N, S, E, W, Stick]$, represented as $[0, 1, 2, 3, 4]$. Rewards and transition function would be setup as described in previous introduction section. Game would be restarted with players starting in random opposing states, specifically, they can be restarted definitely at different column but not necessarily different row. The order of who goes first is stochastic and either of them has 50% probability going first. State is deterministically updated after the successive two actions.

Later, four algorithms were implemented and show how Q-value was updated for player A and for a specific state-action pair. The process of update would be depicted by error term $ERR_i^t = |Q_i^t(s, \vec{a}) - Q_i^{t-1}(s, \vec{a})|$, where s is the state denoted in Fig.1, i represents player A, action-vector a is A chooses South and B Sticks, t means the number of episode when $(s, \vec{a})$ being chosen and to be updated, $t-1$ means the last time before t when $(s, \vec{a})$ being chosen. The expected final results are shown in Fig.$3-4$: [1]

An MDP is a one-player Markov game, while soccer game of this paper is actually a multi-players Markov game. Recall Bellman's equations that characterize the optimal state-action values for a single agent and an MDP, this can be extended to multi-agent Markov games, which share many property as MDP. Thus, multi-agents Markov games can be solved using value iteration with generalized Bellman equation as follows, referring to $Eq.4$ in [1]:

$$Q_i(s, \vec{a}) = (1-\gamma)R_i(s, \vec{a}) + \gamma \sum_{s'} P[s'|s, \vec{a}]V_i(s') \qquad (1)$$

where that i represents the i-th player and $1 \leq i \leq n$, and the action-vectors $\vec{a} = (a_1, a_i, ..., a_n)$. Here $\sum_{s'} P[s'|s, \vec{a}] = 1$ since transition is deterministic. Note that in different multi-agents Q-learning algorithms, value function $V_i(s')$ varies.

B. Q-Learning

First of all, the classic Q-learning. The selections of value function $V_i(s')$ in formula (1) is, referring to $Eq.2$ in [1]:

$$V^*(s) = \max_{a \in A(s)} Q^*(s, a)$$

Note that this paper uses $V^*(s) = 0$ if game ends. As shown above, there is no action-vector $\vec{a}$ in Q-learner algorithm but only action, since two independent Q-table would be implemented for each player desperately instead of mixed Q-table for them both, in that Q-learner was supposed to be isolated and lack of knowledge about another agent. The isolation may lead to not converge of Q-learning, and it didn't in this soccer game. The way of action being chosen by two players was implemented using an ϵ-greedy approach with ϵ starting at 0.02 and linearly reducing to 0.01 after the $T = 10^6$ episodes. (Note that T represents the amount of all steps, not the number of whole game stage should play.) In this paper, learning rate α started at $\alpha_1 = 1$ and exponentially decayed to $\alpha_T = 0.001$ after the $T = 10^6$ episodes. Exponential decay can be written as $\alpha_T = \alpha_0 e^{-\lambda t}$, and λ was calculated as:

$$\lambda = -\frac{1}{T} \log \frac{\alpha_T}{\alpha_0} = -\frac{1}{10^6} \log \frac{0.001}{1} = -\frac{1}{10^6} \log 10^3$$

C. Friend-Q Learning

However, if a mixed Q-table was introduced and the Q-learner picked the best action which benefit both players from the join action space, classic Q-learner would be turned into *Friend-Q*. Refer to $Eq.6$ in [1], value function $V_i(s')$ would be wrtten as:

$$V_i(s) = \max_{\vec{a} \in A(s)} Q_i(s, \vec{a})$$

From the above formula, action-vectors $\vec{a}$ appears, it implies that a mixed Q-table would be introduced here. This is because *Friend-Q* learning is off-policy, meaning players' actions were randomly selected. Off-policy and zero-sum game property $(V_1 = -V_2)$ allows algorithm to be implemented using only Q-table of one player's instead of that of both. In this paper, Q-table was updated with α starting from 0.3 and exponentially decayed to 0.001. Note that all three multi-agents Q-learning algorithms in this paper except classic Q-leaner is off-policy, or, in other words, on-policy with $\epsilon-$greedy where ϵ fixed to 1. Therefore, no ϵ decay was needed in these three multi-agents Q-learning algorithms.

D. Foe-Q Learning

Foe-Q learner utilizes adversary strategy, in which each agent selects an action that minimizes one others utility. Here, the multi-agents Q-learning algorithm being used is called minimax algorithm, where player was trying to find an equilibrium which allowed it gained a maximum payoff conditioned on its opponent who tended to minimize it. The value function $V_i(s')$ being chosen in *Foe-Q*, referring to $Eq.5$ in [1], therefore, is:

$$V_1(s) = \max_{\sigma_1 \in \sum_1(s)} \min_{a_2 \in A_2(s)} Q_1(s, \sigma_1, a_2) = -V_2(s)$$

$\sum_1(s)$ represents the probabilistic action space (PAS) of player 1 (player A) at state s. *Foe-Q* was modeled with a single Q-table, in that in zero-sum game, the rewards of adversary are the negation of that of Player A's. In this paper, a series of linear equations was constructed to implement this minimax algorithm and to solve the probabilities of each action for player A to take on state s, the output is the equilibrium, which would be used as value of $V_1(s')$.

The equilibrium can be computed via linear programming by incorporating the objective function into the linear programming formulation. Refer to $Eq.3$ in [3], the constraints can be written as:

$$\begin{aligned} \text{maximize} \quad & v \\ \text{subject to} \quad & v - \sum_{i=1}^{m} a_{ij} x_i \leq 0, \qquad \text{for all } j = 1, ..., n \\ & \sum_{i=1}^{m} x_i = 1 \text{ and } v \in \mathbb{R} \end{aligned}$$

Here i represents i-th action player A can take, j represents j-th action player B can take. The output v is the expected equilibrium. Then Q-value of current state s would be updated using the generalized Bellman Equation shown in formula (1) α being chosen started from 1 and exponentially decayed to 0.0005. Besides, $5 + 5 + 2 = 12$ equations and $1 + 5 = 6$ variables have been used to implement this linear program.

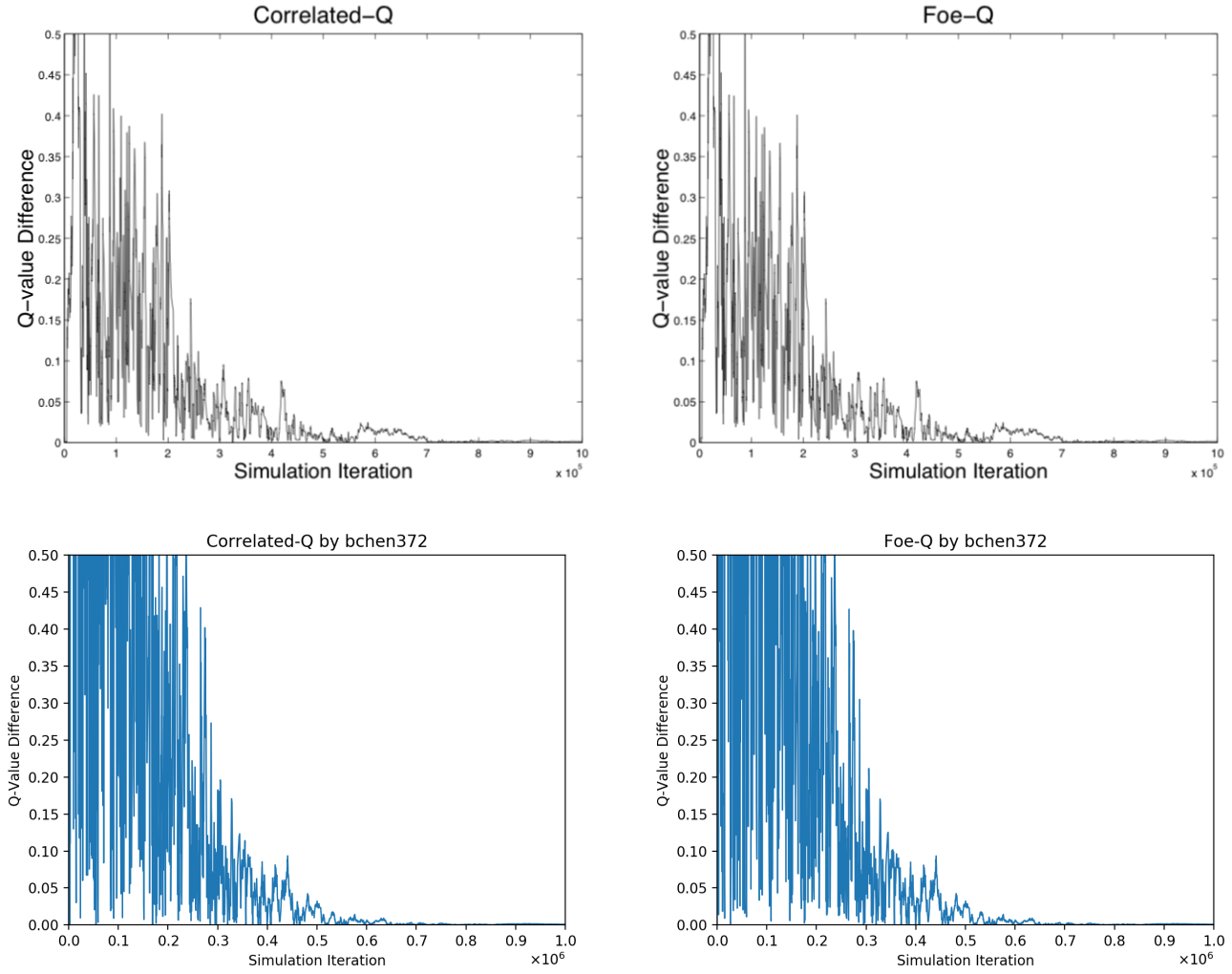

Fig. 3: Convergence in the soccer game (*CE-Q* and *Foe-Q*)

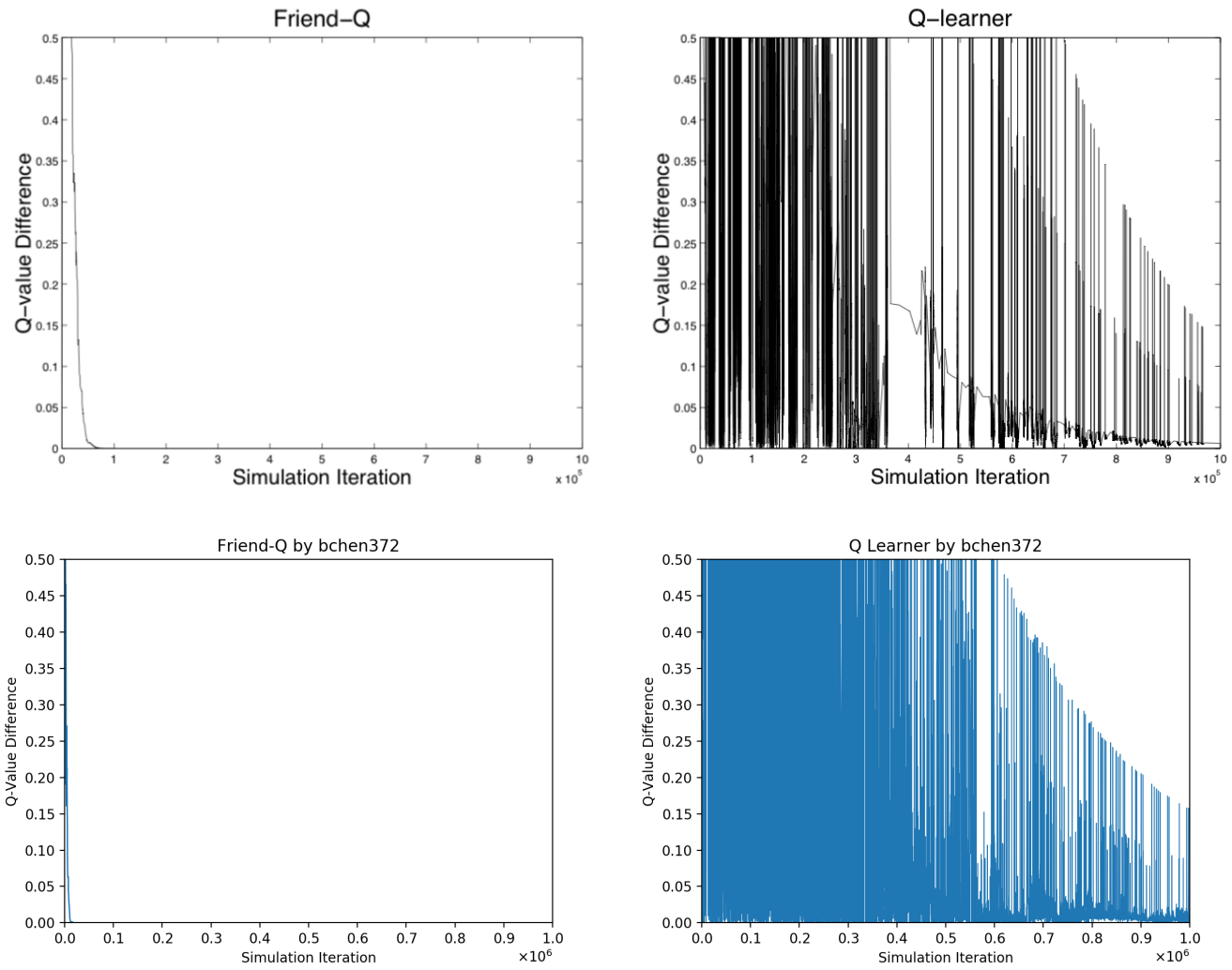

Fig. 4: Convergence in the soccer game (*Friend-Q* and *Q-learning*)

E. Correlated-Q Learning

uCE-Q aims to maximize the sum of both players' rewards. The aggregate of linear equations would be used to form a convex hull which can be easily maximize and then output an expected correlated equilibrium. Both player would benefit from this correlated equilibrium compared with payoff value if they acted independently.

According to Greenwald, the objective function can be written as:

$$\sigma \in \underset{\sigma \in CE}{\operatorname{argmax}} \sum_{i \in I} \sum_{\vec{a} \in A} \sigma(\vec{a}) Q_i(s, \vec{a})$$

The output of *uCE-Q* is called correlated equilibrium. Refer to [4], correlated equilibrium conditions can be written as the following linear program:

$$\text{maximize} \quad \sum_{(i,j) \in A} x_A \sum_p u_A^p$$

$$\text{subject to} \quad \sum_{\bar{a} \in A_{-p}} (u_{i,\bar{a}}^p - u_{j,\bar{a}}^p) \geq 0, \ \forall p \text{ and } \forall i, j \in A_p$$

$$x_a \geq 0, \ \forall a \in A$$

$$\sum_{a \in A} x_a = 1$$

Note that $p = 1, 2, ..., n$ represents p-th player, here $n = 2$. A_p means actions available to player p, $A = \prod_{p=1}^{n} A_p$ means all action pairs, here $A = 5 \times 5 = 25$, $(i, j) \in A$ where i, j are actions chosen from A, B, respectively. $u_{i,j}^p$ equals Q-value of player p when action pair chosen is (i, j).

According to [4], the distribution x is a correlated equilibrium if and only conditioned on player p accepting the

recommended strategy i, i.e. the expected payoff from playing the recommended strategy is no worse than playing any other strategy. The output $\sum_A x_A \sum_p u_A^p$ is the expected result of $V_p(s')$, and then Q-table would be updated according to generalized Bellman Equation with α started from 1 and exponentially decayed to 0.0005. Likewise, $2\times5\times4+5^2+2 = 67$ equations and $1+5 = 6$ variables have been used to implement this linear program.

III. Experiment Results

The expected results, according to Greenwald's paper [1], and results of this paper are shown in Fig.$3-4$:

A. Correlated-Q and Foe-Q

Q-value difference of both experiments fluctuated significantly before the first 3×10^5 episodes and gradually converged as α decreased, consistent with results of Greenwald's. However, Q-value difference of this paper reduce to less than 0.45 only until 2×10^5 episodes was reached, which differs from Greenwald's results where difference was no more than 0.45 just after 10^5 steps. This is likely due to randomness or different α decay pattern as reference's decay more sharply.

Moreover, results from *uCE-Q* and *Foe-Q* are almost identical. This is because in non-zero games, maximizing player's reward also minimizes that of opponent, more precisely, the reward one receives equals the reward its adversary losses. Another reason is that, in a zero-sum game, the minimax algorithm which *Foe-Q* used produced same probability distribution as that of *uCE-Q*, in that probabilities were calculated in a way that forced the player to predict the probabilities of actions that its malicious opponent would take, and so does its opponent.

B. Friend-Q

Friend-Q of this paper quickly converged monotonically, similar to that of reference except that the reference converged about 0.4×10^5 steps slower. This inconsistency may due to different form of α decay as indicated by less sharpness in reference. The quick converge in reference and this paper results from player A's assumption that its opponent would be actively helping player A to reach a maximum score, i.e. by passing it ball and letting it score or moving to A's goal field directly with a ball.

C. Q-Learning

Q-learning is not guaranteed to converge and it didn't in this paper, which consistent with Greenwald's finding. The decrease in Q-value difference curve comes mostly from the decay of α. Shape of decrease in implementation was most identical to reference's, yet this paper fail to replicate the gap located in 4×10^5 as shown in reference. This may due to randomness in that the soccer game was designed to restart from a rational but random opposing position, certain starting position might be chosen subsequently in reference.

IV. Pitfalls and Assumptions

A. Unclear hyper-parameters: α and ϵ

- Reference was silent about the decay pattern of learning rate α and the starting value of it. This is the same issue for choosing ϵ. They could be linear or exponential decay but unlikely a special form of decay since if that's the case, the reference should specify it.
- Since the assumptions on how to decay α and ϵ would change the resulting graph dramatically, this paper then did some fine tuning and finally assumed α decayed exponentially while ϵ decayed linearly, but for different algorithms, the starting value of α and ϵ varies, in order to produce consistent results.

B. Details in environment set up

- As stated in the bold-marked text before, a correct and precise environment set up was crucial for further algorithms implementation, thus an accurate comprehension of reference was required. For instance, an own goal can occur in reference's soccer game, ignoring this would fail further replication as this paper once did.
- Other details including, whether the term *iteration* appearing in reference means one step or one whole game stage, or, can soccer game be restarted at different position, what kinds of starting position can be used, can player A/B possesses the ball at the beginning. These details are not mentioned in reference, and misunderstanding of them would incur huge inconsistency.
- Thus several tests have been conducted in this paper to determine all set up details in reference. Finally, implementation of this paper assumes: only player B can get the ball when game starts; restart states are constrained to the real world soccer game rules with players standing on either East or West. If more possible restart states were considered, i.e. with A at West-North and B at West-South, then algorithm would explore more states and the Q-table update process would be affected, hence the final graph.

V. Conclusion

In multi-agent zero-sum soccer game experiment, *Foe-Q*, *uCE-Q* and *Friend-Q* except for classic *Q-learning* converged. *Friend-Q* has the fastest convergence speed but it converged to an irrational optimistic equilibrium. As for *Foe-Q* and *CE-Q*, they share identical results in that their probability distribution on joint action space were identical because of the property of zero-sum. In contrast, *Q-learning* did not converge as players lack information of their opponent and act independently.

References

[1] K. H. Amy Greenwald, "Correlated-q learning," proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC.

[2] M. L. Littman, "Friend-or-foe q-learning in general-sum games."

[3] T. Roughgarden, "The minimax theorem and algorithms for linear programming," CS261: A Second Course in Algorithms.

[4] A. Saberi, "Correlated equilibria," Computation of Equilibria.