commit 524d09a1d3e65c380fcae8a4c242893b054f7633

# Replication of Correlated Q-Learning

Zhiyi Chen
*Computer Science Department*
*Georgia Institute of Technology*
zchen853@gatech.edu

***Abstract*— The correlated equilibrium Q-learning was proposed by Amy Greenwald and Keith Hall in 2003. It is able to generalize Nash Q-learning in multi-agent reinforcement learning framework, with the advantage of being solvable using linear programming. In Greenwald 2003, ordinary Q-learning, Littman's Friend-Q and Foe-Q, and CE-Q are compared on a zero-sum grid Soccer game. The key results from the paper are replicated in this work. The results from this work aligns with those presented in the Greenwald 2003, where ordinary Q-learning did not converge, Friend-Q converged to an irrational policy for this zero-sum game very quickly, and both Foe-Q and CE-Q converged to the same minimax equilibrium policy.**

***Keywords—Markov game, Q-Learning, Friend-Q, Foe-Q, uCE-Q, Mixed strategy game, Minimax equilibrium***

## I. Introduction

The objective of this work is to replicate some of the key results of multi-agent Q-learning algorithms on the grid soccer game presented in Greenwald 2003. Four Q-learning algorithms, namely the ordinary Q-learning, friend-Q, foe-Q and utilitarian correlated equilibrium-Q (uCE-Q), are trained on the grid soccer game. Convergence of each algorithm are studied. Section II will briefly discuss Markov games and explain the soccer environment. Section III briefly discusses the four Q-learning algorithms. Section IV lays out how each Q-learning algorithm is implemented and assumptions made. And lastly, Section V discusses results of the replication and some difficulties encountered during replication.

## II. Background

### *A. Markov Games*

A Markov decision process (MDP) is defined by a 4-tuple $(S, A, T, R)$ where S is the set of states, $A$ is the set of actions, $T: S \times A \rightarrow S'$ is the transition function from state $S$ to next state $S'$ given action $a \in A$, and R is the immediate reward reaching state S'. In the MDP formulation of reinforcement learning, a single agent interacts with the environment and learns the optimum policy to maximize reward. MDP and repeated games can be generalized as the Markov games, which formalizes the framework for multi-agent reinforcement learning. Markov games are defined by a 5-tuple $(I, S, \left(A_i(s)\right)_{s \in S, 1 \le i \le n}, T, (R_i)_{1 \le i \le n})$, where $I$ is a set of $n$ players, $S$ is the set of states, $A_i(s)$ is the $i^{th}$ player's set of actions at state $s$, $T$ is the transition function, and $R_i$ is $i^{th}$ player's reward for state $s \in S$ and joint actions $\vec{a} \in A(s) = A_1(s) \times \ldots \times A_n(s)$ [Greenwald 2003]. The state-action vector value function for Markov games follows nicely from the bellman's equation:

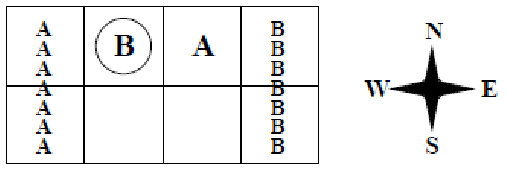
*Equation 1*

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

The MDPs are thus a special case of Markov games (one-player), where the Reward and Transition function are both a function of state-action pair instead of state-action vector pair as shown in equation 1.

### *B. Environment*

In this experiment, multi-agent Q learning is trained in the zero-sum Soccer environment described in Greenwald 2003. The soccer field is a 2-by-4 grid shown in figure below. The first column of the grid is the goal area for player A, and the last column is the goal area for player B. Every game starts with the status shown in figure, and player B always has the possession of the ball. During the game, each player will select an action from the action space [N, S, E, W, Stay], the actions will be executed in random order. Whenever an action is executed, if the new location resulted from the action is inbound and is not occupied by another player, the executor will go to the new location, or else, the executor will stay at the current location, hence the two players will never end up in the same location. When a player with possession of the ball runs into the other player, possession of the ball will be lost to the other player, however, if a player ran into the other player with the ball, possession will remain unchanged, which means ball cannot be tackled. When a player with the ball reaches his goal area, the player will score 100 points, while the other player scores -100. If the player with the ball reaches the other player's goal, he will score -100 points, while the other player scores 100 points.



## III. Related work

### A. Ordinary Q-Learning

Ordinary Q-learning solves Markov game in same fashion as it does in an MDP using bellman operation. Each Q-learning agent aims to solve for a deterministic optimum policy by only considering its own actions and ignoring its opponent's actions. The optimum state-action value function for each agent follows from equation 1:

*Equation 2*

$$Q^*(s,a) = (1-\gamma)R(s,a) + \gamma \sum_{s'} P[s'|s,a]V^*(s')$$

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

The optimum policy $\pi^*(s)$ is described by the actions that maximizes $Q^*(s,a)$ at each states $s$.

### B. Friend Q-Learning

A Littman's friend-Q updates equation 1 with a value function that consider both players' actions. When computing the value function, each agent assumes that the opponent will always play in its favor of achieving a higher reward. Equation 1 is updated with the value function:

*Equation 3*

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

The policy pi from Friend-Q is thus the actions that maximizes Q at each state s and the action vector a.

### C. Foe Q-learning

On the other hand, Littman's foe-Q updates equation 1 with the minimax value function in equation 5. In this case, the agent assumes that the opponent will always take action to minimize its potential rewards.

*Equation 4*

$$V_1(s) = \max_{\sigma\in\Sigma_1(s)} \min_{a_2\in A_2(s)} Q_1(s,\sigma_1,a_2) = -V_2(s)$$

The minimax value and policy are extracted from Foe-Q state and action-vector value function by solving equation 6 with linear programming.

### D. Utilitarian Correlated Q-learning

Utilitarian correlated equilibrium Q-learning (uCE-Q) was proposed in Greenwald 2003. It maximized the total reward of all agents by considering the joint probability distributions of players' actions and conditional on a player's own action. Correlated Equilabrium generalizes the Nash Equilibrium, which coincide with minimax strategies in zero-sum games. uCE-Q updates equation 1 with the value function in quation 6.

*Equation 5*

$$\sigma \in \arg\max_{\sigma\in CE} \textstyle\sum_{i\in I} \sum_{\vec{a}\in A} \sigma(\vec{a}) Q_i(s,\vec{a}) \text{ c}$$

## IV. Implementation details

The Q-learning algorithms are implemented as explained in the Greenwald 2003. However, some implementation details are not explicitly pointed out the paper or is ambiguous. The template for multi-agent Q-learning copied from Greenwald 2003 is shown in table 1.

*Table 1: Multi-agent Q-Learning.*

```
MULTIQ(MarkovGame, $f$, $\gamma$, $\alpha$, $S$, $T$)
  Inputs       selection function $f$
               discount factor $\gamma$
               learning rate $\alpha$
               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 $\alpha$ according to $S$
```

The main differences between implementations of the 4 Q-learning algorithms are around how actions are simulated in step 1, the computation of value function in step 3a, as well as different parameters used. Rest of the section breaks down the implementation by on-policy algorithm (ordinary Q-learning) vs off-policy (Friend-Q, Foe-Q, CE-Q).

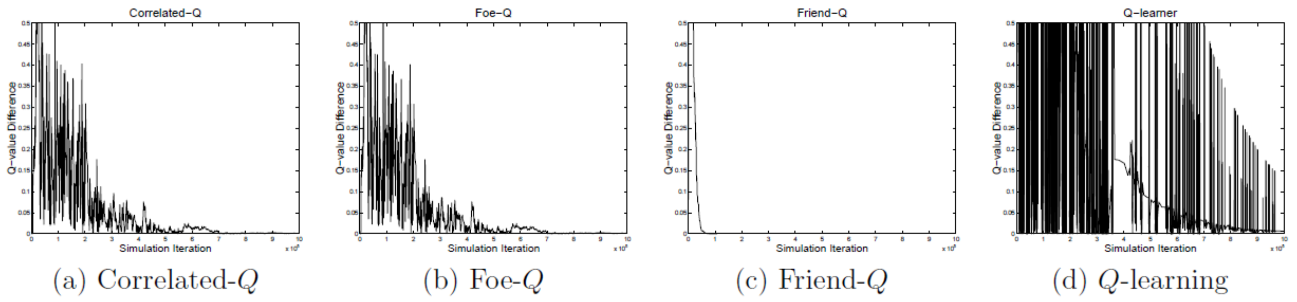
### A. On-policy

Q-learning is implemented as epsilon-greedy in the Greenwald 2003, however, it is referred to as "on-policy" by the author. In the paper, author mentioned about decaying both alpha and epsilon to 0.001, and setting gamma to 0.9. However, initial values for alpha and epsilon, as well as the decay schedule are not provided in the paper. Based on Littman 1994, initial values for alpha and epsilon are set to 1.0 and 0.2 respectively, and exponential decay is used, where decay is set to $10^{\frac{\ln(0.01)}{10^6}} = 0.999993$ both alpha and epsilon, which decays alpha from 1 to 0.001 in 1 million steps. Value function for Q-learning is shown in equation 2.

### B. Off-policy

Friend-Q is implemented as off-policy in the paper, hence, at step 1 of table 1, actions are simulated randomly at each step. Alpha is initialized at 0.1 and exponentially decay to 0.001 in 1 million simulation steps, gamma is again set to 0.9. Value at step 3a of table 1 is calculated based on equation 3. Foe-Q is implemented similarly as Friend-Q. The differences being that alpha is initialized at 0.05 and value is calculated based on equation 4. Lastly, CE-Q is implemented with parameters exactly same as that for Foe-Q, the only difference is that equation 5 is used to compute value at each Q-table update. Both equation 4 and 5 are solved using linear programming.

*Figure 1:Q-value difference for each Q-learning algorithm presented in Greenwald 2003.*



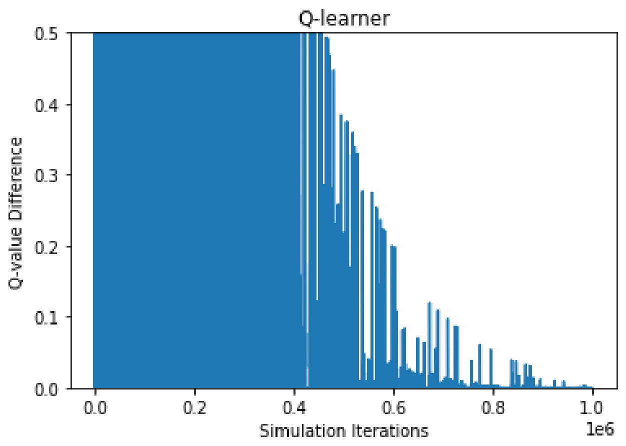
## V. Results and Discussion

In this experiment, each Q-learning algorithm is trained on the Soccer environment for 1 million simulation iterations. The temporal differences of the Q-value for the state depicted in figure 1, with player A taking action S and player B Staying, is plotted against simulation iterations. The temporal difference is calculated according to $\mathrm{ERR}_i^t = \left|Q_i^t(s,\vec{a}) - Q_i^{t-1}(s,\vec{a})\right|$. Figure 1 shows results presented in the Greenwald 2003.

### *A. Q-learning*

Figure 2 shows the result obtained in this experiment for the Q-learner. The result generally aligns with that shown in Greenwald paper. Q-value difference fluctuated while the average trending downward. However, as is explained in the Greenwald paper, such decrease is resulted by the decaying alpha. Evidently, fluctuations are still observed at the very end of the iterations, which means the Q-learner is still changing its actions. This result is expected since the Q-learner does not perceive the existence of an opponent, who changes the state from time to time, in other word, Q-learning does not learn mixed strategy games.

*Figure 2:Q-value difference plot for ordinary Q-learning.*



### *B. Friend-Q*

Figure 3a shows the Q-value difference for Friend-Q with difference lagged by 1 iteration and Figure 3b shows the Q-value difference for the same Friend-Q with difference lagged by 5000 iterations. Both plots show that Friend-Q converged very quickly, with the latter case aligned closely with that presented in the Greenwald 2003. The reason that lagging 5000 iterations smooth out the Q-value difference curve is because the state is not visited many times throughout the 1 million iterations, as a result, there will be many zeros in the differences if lagged by only 1 iteration. Such fast convergence is because Friend-Q assumes opponent will help it to score. At state S, player B would want to pass the ball to player, in hope player A will score for him, on the other hand, player A expect player B is score immediately for her, thus does not care about what action to take next, resulting in a uniformly distributed probabilities for the next action. Even though Friend-Q converges, such convergence is not rational for the zero-sum soccer game.

*Figure 3a: Q-value difference plot for friend-Q lagged by 5000 iterations.*

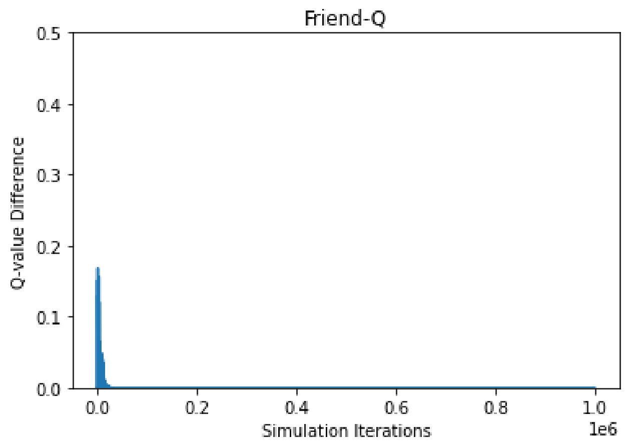
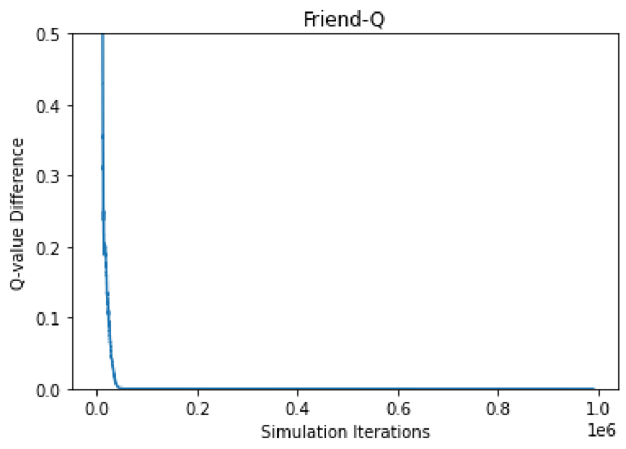


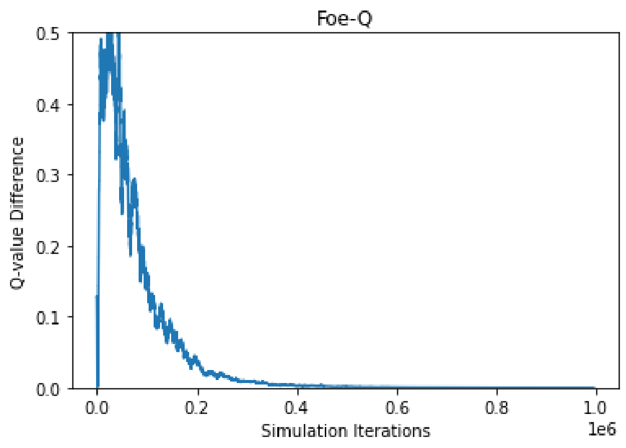
*Figure 3b: Q-value difference plot for Friend-Q with 1 iteration lag.*



## C. *Foe-Q*

Figure x shows the Q-value difference plot for Foe-Q with differences lagged by 5000 iterations. The result generally aligns with that presented in the Greenwald. Less fluctuation observed might be resulted by slightly different parameters used, however, this cannot be verified since the exact parameters used by Greenwald 2003 are not disclosed. It is observed that Foe-Q has converged after around 400,000 iterations. Since this algorithm appropriately considered the rivalry relationship between the two players, it did not converge recklessly like Friend-Q did.

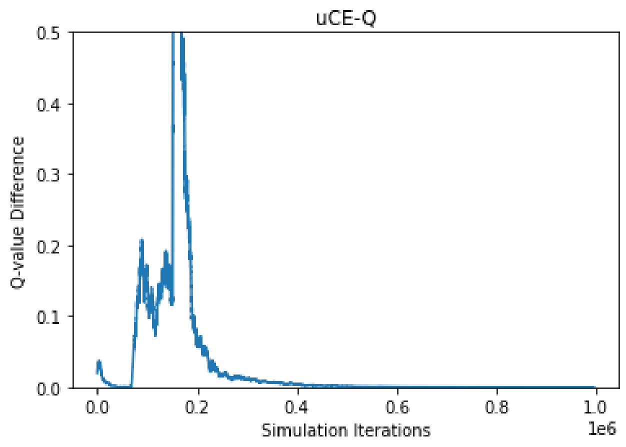
*Figure 4: Q-value difference plot for Foe-Q.*



## D. *Correlated-Q*

Figure x shows the Q-value difference plot for uCE-Q, again differences are lagged by 5000 iterations for better visualization. The result, relative to that for Foe-Q in this experiment, is comparable to that presented in Greenwald 2003. uCE-Q converged to mixed strategy policy for both players. However, more fluctuation is observed at begin for uCE-Q compared to Foe-Q. But eventually, uCE-Q is able to learn the same minimax equilibrium policies as Foe-Q for this two-player, zero-sum Soccer game.

*Figure 5: Q-value difference plot for uCE-Q.*



## VI. Conclusion

In this experiment, results presented in Greenwald 2003 are replicated. The results are largely replicable with some pitfalls encountered. The first pitfall was with the ambiguous description of the Soccer environment. The Author did not specify some of the rules such as what happen when player go out of bound, which is assumed that player would stay at current location without change of possession in this experiment. Secondly, the parameter settings for each of the Q-learning algorithm were not fully specified. For instance, the initial values for alpha, and epsilon in the case of Q-learning, as well as the decay schedule for the parameters. In this case Littman 1994 has to be referenced. Lastly, the author did not disclose the post-processing procedure for the Q-value differences, which is believed to have resulted in the specific visualization presented in the original paper.

In general, the results from this experiment aligns with those presented in the Greenwald 2003, where ordinary Q-learning did not converge on the Soccer game, Friend-Q converged to an irrational policy for this zero-sum game very quickly, and both Foe-Q and uCE-Q converged to the same minimax equilibrium policy.

## References

[1] A. Greenwald and K. Hall. Correlated-Q Learning, 2003

[2] M. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedingsof the Eleventh International Conference on Machine Learning*, pages 157-163, July 1994