https://github.gatech.edu/gt-omscs-rldm/7642Fall2020schoi344/tree/a2a2f65d47583f939cf3e6712533a3dc49efe6d9

# Correlated Q-Learning

## CS 7642

CS 7642 – Reinforcement Learning
Project 3

GTID: schoi344 / 903034930
Written by: Samuel Choi

## Abstract

In 2003, Amy Greenwald and Keith Hall published a paper titled "Correlated Q-Learning". In their report, they experimented with four different Q algorithms, Correlated-Q, Foe-Q, Friend-Q, and classic Q-Learning, and graphed the convergence of the Q-value difference between the updated Q of the starting state and the previous one. Once the difference in the algorithms' graphs converges to a difference of 0, we know that convergence is completed. The purpose of this report is to replicate the graph results of Figure 3 in the paper to confirm their findings, and show that Correlated-Q and Foe-Q yield similar graphs and that classical Q-Learning is the least efficient among the algorithms and fails to converge even after a million iterations. The Correlated-Q algorithm is based off the Nash-Q algorithm in the Hu and Wellman 1998 paper, but the Nash equilibrium was too cumbersome to calculate. As a result, correlated equilibriums (CE) was introduced as a solution that uses linear programing to enable a more digestible way of computing it similarly. Lastly, the Friend-Q and Foe-Q algorithms are derived from the Littman 2001 paper. Through clever implementation of multiplayer MDP with multi-agent Q-Learning, classical Q-Learning can be greatly outperformed.

## Experiment

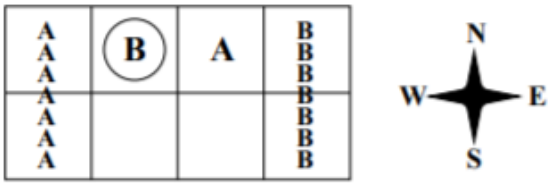



Figure 1: A diagram of the game used in the experiment [Greenwald Hall 2003]

A 2x4 soccer grid is used in the experiment as shown in Figure 1. There is some ambiguity in the exact rules of the game, so the rules used in this replication experiment will be explained here. There are two players, and there are 8 grid spaces, so there are a total of 8x8x2 possible player states with the 2 representing ball possession. The left and right spaces are the goal position for player A and B respectively. Player A and player B randomly start off in the middle four spaces, and they cannot be in the same grid space at once. If whoever possesses the ball ends up in the goal, the benefiting party gets +100 reward points and the opposite party gets -100 points. In all other scenarios, the reward for both parties is 0 points. Each player has 5 options for actions that can be taken: move East, West, South, North, or stick to current position. If a player tries to move out of bounds, the player will stick. The player that performs its action first is randomly determined, and if the player that moves first has the ball and tries to move into the position the other player is in, the ball possession will change to the other player.

The simulation soccer game that is used for the experiment is a form of a multiplayer Markov Decision Process (MDP) with notation <I, S, $A_i(s)$, P, $R_i$)>. I is the number of players, S is the states, $A_i(s)$ is the action vectors of players, P is the probabilistic transition function, and $R_i$ is a player's reward.

## Algorithm

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

Figure 2: Equation for Q-table updates

All four algorithms used the same Q update equation shown in Figure 2 with just the $V_i(s)$ portion being changed out for each algorithm.

Correlated-Q used a glpk solver from the cvxopt library to linearly solve the expected returns of player A and player B by generating a correlation matrix, generating b and c vectors, and inputting it into a solver.

Foe-Q was similar in that it used a glpk solver to linearly solve expected outcomes for its Q table. However, it only had one Q-table instead of two. Also, its A matrix was its current Q values with a column of ones attached at the end instead of a correlation matrix.

Friend-Q didn't use any solvers. It instead took the maximum value from the Q-table, and its Q-table was for both players. No epsilon was used.

Q-Learning used a classical approach with greedy decaying epsilon. It is almost identical to Friend-Q here, but instead of having a Q-table for both players, it was only for player A.

## Results & Discussion

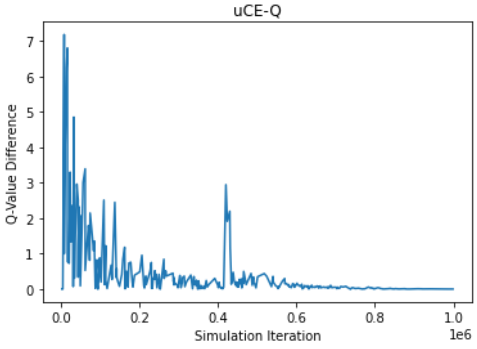
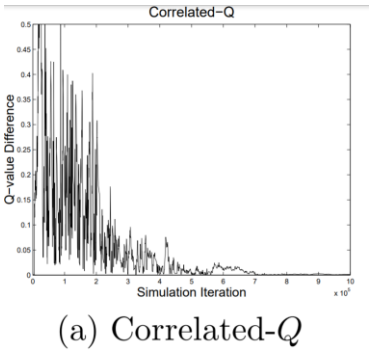



Figure 3: Simulated Graph for Correlated-Q




(a) Correlated-$Q$

Figure 4: Original Graph for Correlated-Q [GH03]

In figures 3 and 4, we see the simulated Q-value difference error for the simulated and the original graphs respectively. Remarkably, the trend of both graphs seems to be following the same trend up until a little bit past the 400 thousandth mark. Afterwards, the Q-value difference seems to be going back a bit upwards, whereas it has a continuous downwards trend in the original. This may be attributed to different alpha and gamma parameter tweaking. Another theory is that in the simulation, there may be some noise the is introduce in the correlation matrix that gets mixed in. We can see the proof of this in the appendix for another run of the algorithm.

https://github.gatech.edu/gt-omscs-rldm/7642Fall2020schoi344/tree/a2a2f65d47583f939cf3e6712533a3dc49efe6d9

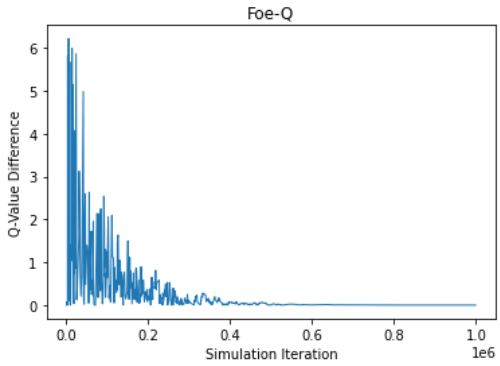
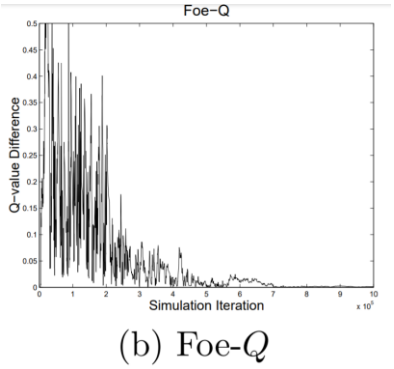



Figure 5: Simulated Graph for Foe-Q




(b) Foe-$Q$

Figure 6: Original Graph for Correlated-Q [GH03]

Next, we see the simulated and original graphs for Foe-Q. We can see that the graphs a very similar. Although we didn't get the results we wanted for the simulated Correlation-Q graph and Foe-Q also used the same solver, the simpler approach in creating the system of equations may have attributed to it coming out cleaner.

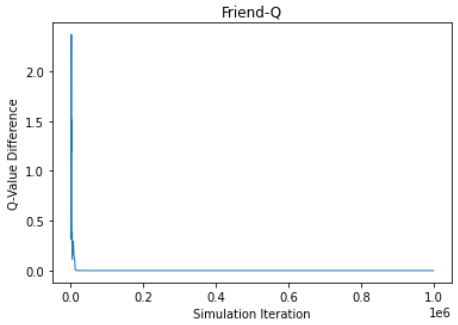
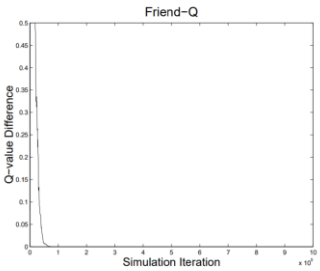



Figure 7: Simulated Graph for Friend-Q




(c) Friend-$Q$

Figure 8: Original Graph for Friend-Q [GH03]

The both graphs for Friend-Q converged very quickly. A theory behind its quick convergence is its multi-agent approach allows for one table to gather data and correct itself relatively quickly. Also, it doesn't get its expected values from a system of equations, but rather from gathering the best values. Always choosing the best value is what likely led it to its fast convergence.

https://github.gatech.edu/gt-omscs-rldm/7642Fall2020schoi344/tree/a2a2f65d47583f939cf3e6712533a3dc49efe6d9

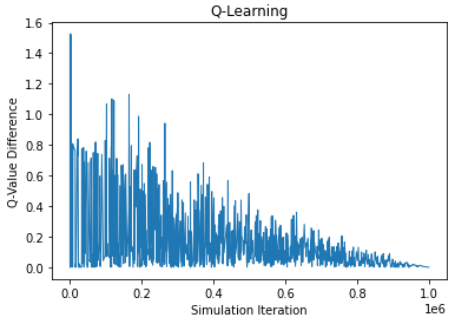
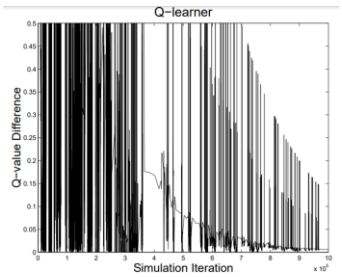



Figure 9: Simulated Graph for Q-Learning




(d) $Q$-learning

Figure 10: Original Graph for Q-Learning [GH03]

The original graph for Q-Learning is very convoluted. If it is not because of noise, we can see two downwards trends that alternate between each other. The simulated graph more closely resembles the graph that is part of the lower trend. The simulated Q-learning converges the slowest, but it is consistently on a downward trend. Although more similar to Friend-Q, which converged the fastest, its slower or non-converging trend can be attributed to it only considering one player as its agent, limiting cross-correlated data. As a result, Greenwald and Hall were able to show the advantage of multi-agent Q-Learning and its benefits of speed in comparison to traditional Q-Learning.

# Appendix

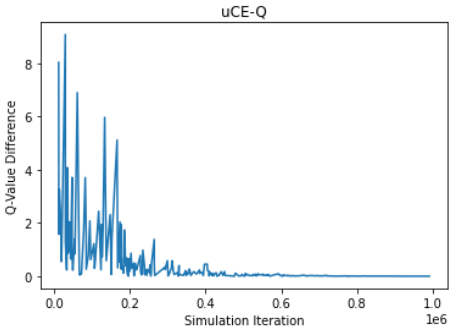



Figure 11: Another run on Correlated-Q

Running through the Correlated-Q algorithm yielded a cleaner graph. Confirming my theory of there being some noise in the middle of the first time running though the algorithm.

https://github.gatech.edu/gt-omscs-rldm/7642Fall2020schoi344/tree/a2a2f65d47583f939cf3e6712533a3dc49efe6d9

# References


[GH03] Amy Greenwald and Keith Hall. Correlated-Q Learning. Brown University, 2003

[HW98] Junling Hu and Michael P Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In Proceedings of the Fifteenth International Conference on Machine Learning, pages 242-250 July 1998

[L01] Michael L Littman. Friend or foe Q-Learning in general-sum Markov Games. In Proceedings of the Eighteenth International Conference on Machine Learning, pages 322-328, June 2001

[SB20] Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. 2nd Ed. MIT press, 2020. url: http://incompleteideas.net/book/the-book-2nd.html.