Project 3-https://github.gatech.edu/proy36/project3

This projects introduces four types of CE-Q experiments which shows the convergence to equilibrium policies on a testbed of general-sum Markov games..Correlated Equilibrium Q-learning(CE-Q) is an Multi-agent Q-learning algorithms which generalizes Nash-Q, Friend-Q and Foe-Q.
Paper attempts to resolve this equilibrium selection problem by introducing four variants of CE-Q, based on four equilibrium selection functions.
The difficulty that poses while solving correlated equilibrium policies in general-sum games is the presence of multiple equilibria with multiple payoff values. The four variants of selecting equilibrium functions is described in the paper and below graphs represents the same. The experiment also discusses the theory of stochastic stability, which could be employed to describe the convergence properties of our algorithms.

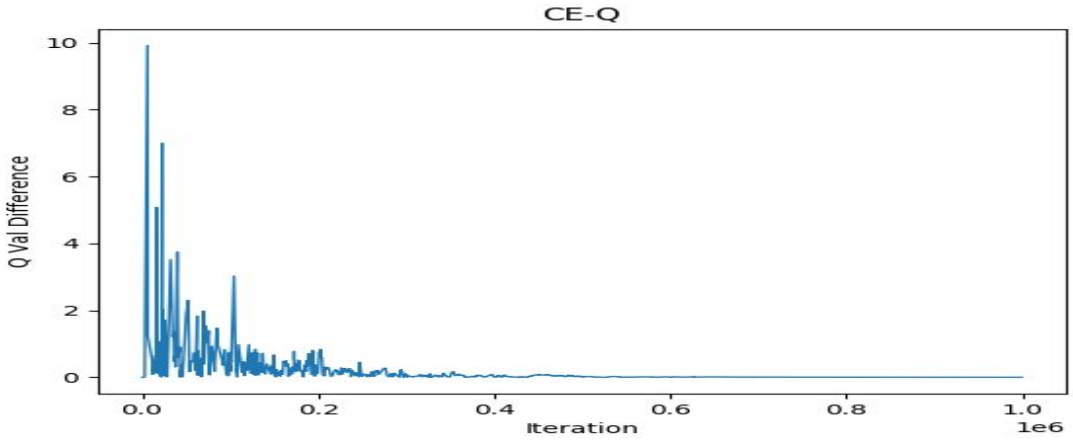
Below are the results of section 5 of Soccer Game and the analysis of the data collected by replicating Greenwald's graphs for Q learning,CEQ,Friend-Q and Foe-Q.

Game Rules

We have 2*4 grid representing a soccer field with two players and a ball(circle), We randomly chose 5 possible actions are North,South,East, West and Stick. First player moves if the sequence causes players to collide and the ball changes possession if the player with the ball moves second. Finally If the player with the ball moves into a goal, then he scores +100.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.In other words, if the player without the ball moves into the player with the ball, attempting to steal the ball, he cannot. But if the player with the ball moves into the player without the ball, the former loses the ball to the latter.

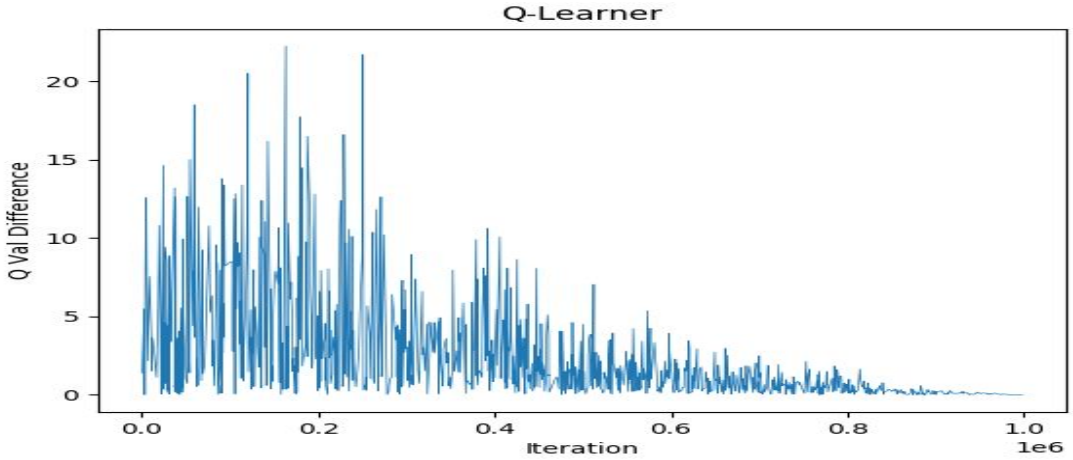
CEQ

All Nash equilibria are correlated equilibria. Four variants of CE-Q are utilitarian (uCE-Q), egalitarian (eCEQ), republican (rCE-Q) and libertarian (lCE-Q), which also demonstrate empirical convergence to equilibrium policies on a testbed of general-sum Markov games.



For CE-Q a set of linear equations for constraints of 2 players were taken into account and we used two Q tables attempted to solve complex set of equations regarding each player's choice of strategy given the knowledge about 2nd players' distribution gained from Player 1 own prescribed action, that player gains no increase in expected payoff by ignoring the prescribed action. This correlated joint policy is more efficient than Nash-Q, since it does not require the complex quadratic programming Nash equilibrium solver. Instead, according to conditional probability rule.correlated equilibria can be computed via linear programming by treating one player's action as a conditional constraint.Both players maintain equilibrium since following correlated strategy pairs will always be in their best interest.

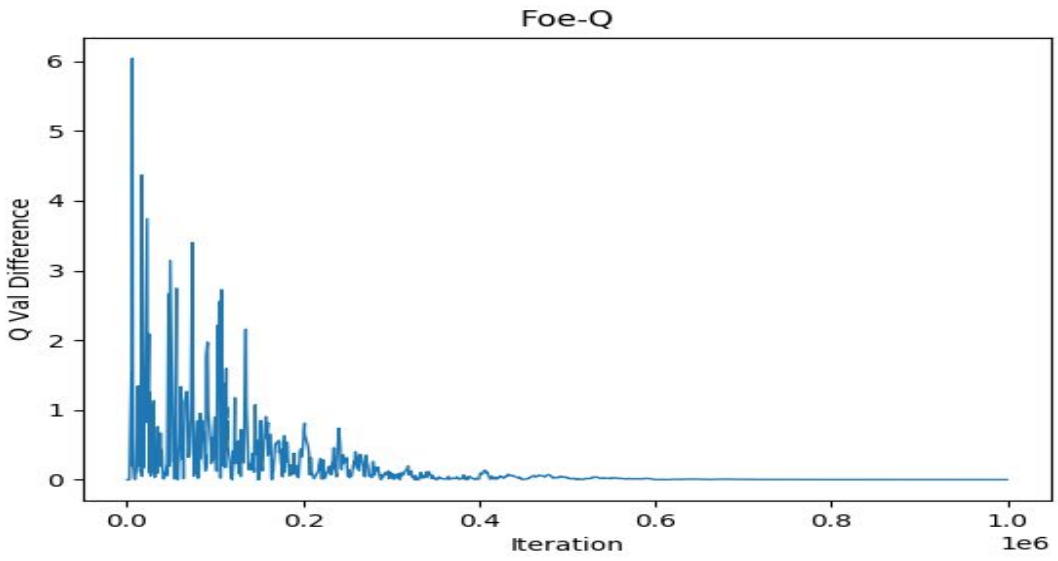
Q- Learning



Q-learning works very well to find optimal policies in MDPs in single-agent learning scenario. In multiagent learning environment like soccer game it does not perform very well. In a scenario where other players play a stationary strategy then Q-learning learns to play an optimal response to the other players. Finally it is difficult for Q-learning to play stochastic policies.
Our Q learning approach did not converge and we know that Q learning does not guarantee convergence.we used Bellman's Equation to update Q table for each player. We used epsilon-greedy approach and Below Parameters for the algorithm

I started with alpha 1 and decayed it gradually(linear) to 0.001 but decaying exponentially provided similar pattern with paper but the results were not a exact match. In the paper we can notice intermittent rise which is present for initial iteration but missing as the iterations increase.Further experimentation is needed to see the results of decaying alpha.

FOE-Q

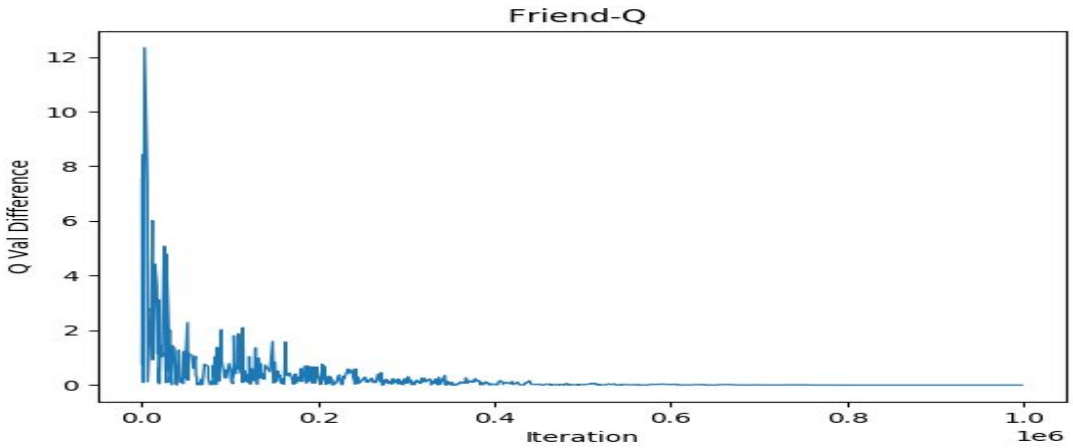


Friend-Q and FOE-Q is an equilibrium learner that extends Minimax-Q to include a small class of general-sum games. The algorithm assumes there are two kinds of competing agent in the stochastic games from one agent's perspective: either a friend or a foe. Knowing the labeling or inferring from the observed rewards, equilibrium policies can be learned in restricted classes of games: e.g. two-player, zero-sum stochastic games, which computes the basic zero sum linear program of the minimax equilibria (foe-Q). In our experiment we used a mixed strategy and each player did not know opponents strategy.Opponent action was based on probability for each player for maximum pay off.

We used a single Q table for FOE-Q. Player 2 goal was minimize Player 1 rewards and vice versa in other words each agent reward is negation of opponents return which is how a we can describe a nature of a zero-sum game. Using our minimax algorithm we are able to model our player 2 and we see that our FOE-Q converges really well.
For an agent in state S and action A we calculated the probabilities using linear programming and calculated future actions.

After close to 600K Iterations we can see our values converge

Friend Q



There is minimal learning in the Friend Q algorithm and it is also was very fast and it gradually converged since Player 2 provides the ball Player 1 so that he can score this

helps Player 1 in getting a maximum score. The algorithm takes random actions and we find that Friend-Q runs off-policy learning.

FOE-Q and CEQ-Q Table Differences

As per the paper both Foe-Q and CE-Q converges to same Q values and we were able to see that in our experiment and it matches the paper.

0.61701121	0.47419457	0.47117515	0.59289293	0.42531792
0.27063617	0.41522899	0.75458769	0.46201886	-0.13290417
0.40041829	-29.509164	0.49626941	0.10345327	0.50414546
0.4378459	0.50046284	0.44832698	0.5777054	0.4918819
0.45867295	0.44374643	0.39169819	0.59502167	0.59054063

Conclusion

I was able to replicate the algorithm and as far as possible had the graphs represented very close to the experiment results. Initial choice of 2 q tables provided output which did not match the graphs and therefore I chose 1 q table for the main player Instead of choosing 2 q tables for each player. Different choices of alpha and epsilon completely changed the graphs and different values was fed to get the results below.The gamma used was 0.9 for all of our experiments.