Correlated Q-Learning Analysis

Travis Latzke

1 Introduction

Life is filled with games that live on a wide spectrum ranging from simple to complex. Defining a game to be simple or complex, or anywhere in between can be tricky due to slight nuances in different game variations. However, games are typically considered to be less complex when there are fewer agents, states, actions, and when the best strategy for the agents is deterministic. One example of a simple game could be two players (A and B), where player A's objective is to push player B as far as possible and player B's objective is to push player A as far as possible. Each player is rewarded for how far they push the other player. In this game, the strategy is easy to calculate for both players; each player simply pushes the other player with as much force as they can apply. If any player deviates from this strategy, then the player would not obtain its maximum reward for playing the game. Thus, it is beneficial for both players to never deviate from this strategy, which results in both players having a deterministic strategy.

Games become more complex when the number of players, states, and actions increase and when the optimal strategies for each player is stochastic. An example of a complex game (relative to the simple game described previously) is the game of Soccer, which is explained in section 3. Littman and Greenwald have led studies that illustrate different methods for calculating optimal stochastic strategies to be used by players in complex games. Greenwald's paper on correlated learning shows how her methods generalize some of Littman's methods that he introduced in his Foe-Friend Q-Learning paper. Their methods and the analysis of their methods will be the primary focus of this paper. The algorithms used in their methods rely on the mathematical frameworks of Markov Decision Processes (MDP's) and Markov Games, so the next sections are devoted to define those frameworks.

2 Mathematical Framework

2.1 Markov Decision Process (MDP)

MDP's are formally defined as a 4-tuple $\langle S, A, P, R\rangle$ where S is a finite set of states, A is a finite set of actions, $P : S \times A \times S \rightarrow Pr(s'|s)$ is the probability transition function that denotes the probability of transitioning from state $s \in S$ to state $s' \in S$ when taking the action $a \in A$. $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function for the agent when taking action $a \in A$ in state $s \in S$ and transitioning to state $s' \in S$. The goal of the MDP is to find a policy $\pi : S \rightarrow A$ that maximizes the total reward that an agent receives if the agent were to follow π in every state. For the Soccer game described in section [11111] the goal is to find a policy that when followed is best suited for the agent to score a goal. The traditional Q-learning update step for calculating an optimal policy is as follows:

$$Q(s,a) \leftarrow (1-\alpha) * Q(s,a) + \alpha(r + \gamma * V(s')) \quad (1)$$

$$V(s) \leftarrow argmax_{a \in A} Q(s,a) \quad (2)$$

The core pitfall in the above framework is that it does not account for how to act based on how other agents in the environment will act. Fortunately, the MDP framework can easily be extended into a Markov game framework by extending some of its core mathematical set structures into joint sets structures.

2.2 Markov Game

Stochastic games generalize repeated games and Markov decision processes (MDPs). A stochastic game is a tuple $\langle I, S, (A_i(s))_{s \in S, 1 \leq i \leq n}, P, (R_i)_{1 \leq i \leq n}\rangle$, where I is a set of n players, S is a set of states, $A_i(s)$ is the ith player's set of actions at state s, P is a probability transition function that describes state transitions, conditioned on past states and joint actions, and $R_i(s, \vec{a})$ is the ith player's reward for state $s \in S$ and joint actions $\vec{a} \in A(s) = A_1(s) \times ... \times A_n(s)$. Stochastic games for which the probability transitions satisfy the Markov property are called Markov games: $i.e..$, for $\vec{a_t} = (a_1, ..., a_n)_t, P[s_{t+1}|s_t, \vec{a_t}, ..., s_0, \vec{a_0}] = P[s_{t+1}|s_t, \vec{a_t}]$ [Greenwald, 2004].

One of the core differences between traditional MDP's and Markov games, is that Markov games encapsulate the joint action space, joint transition probability, and joint reward function for all players. This allows for more information to be fed into the value function, which is conducive to finding an equilibrium policy (evidence for this is shown in figures 1, 2, 3, and 4).

Recall the value function function equation (2) from the previous section. The equation states that $V(s)$ is updated based on the $argmax$ function on the right hand side of the equation. $argmax$ is not sufficient in calculating an equilibrium policy because it is impossible for $V(s)$ to converge for all $s \in S$ in some games. To illustrate why (2) is not sufficient for finding an equilibrium policy, imagine a two player game with players A and B. Lets say that after several moves, the current state of the game is s and

the best action for A in s is a^* and the best action for B in s is o^*. Suppose that state s is highly stochastic in terms of the reward for both players. Lets assume that the game stochastically favored A and penalized B for the a^*, o^* action pair taken at state s. This could result in player B updating its q-table at $Q(s, o^*)$ in a way that might result in B choosing a different action o' the next time it is the game reaches state s. However, since we assumed that o^* was the best action for B at state s, the learning algorithm will eventually update itself so that B will eventually choose o^* again. And this process could continue indefinitely, due to the stochasticity of state s.
To prevent indefinite cycles in value states like the example above, Littman showed how the minimax function can be incorporated into the value function, so that equilibrium policies could be calculated in highly stochastic games. The minimax value function is defined as:

$$V_1(s) = max_{\sigma \in \Sigma_1(s)} min_{a_2 \in A_2(s)} Q_1(s, \sigma_1 a_2) = -V_2(s) \quad (3)$$

On the contrary, Littman also studied stochastic games where players act on the same team and cooperate with each other to maximize the reward for the $team$ or both players. The value function for this scenario is defined as:

$$V_i(s) = max_{\vec{a} \in A(s)} Q_i(s, \vec{a}) \quad (4)$$

Littman's value functions pertain to two player zero sum games. Greenwald's correlated equilibrium (CE) defines the value function in a way that generalizes (3) to include more than two players if the right objective function is chosen. Greenwald defines this value function as:

$$V_i(s) \in CE_i(Q_1(s), ..., Q_n(s)) \quad (5)$$

To clarify how [5] generalizes [3] in special cases, we need to look at how we can relate [5] to [3] with the correct objective function:

$$\sigma \in argmax_{\sigma \in CE} \sum_{i \in I} \sum_{\vec{a} \in A} \sigma(\vec{a}) Q_i(s, \vec{a}) \quad (6)$$

Recall that [3] can be computed by using the method of linear programming. The goal of the linear program is to return a probability distribution for all joint action pairs, with higher probabilities being associated with an action pair that is more likely to result in a higher reward for agent. The following constraints could be fed into a linear program for a two player (A and B) game where A's actions at state s are $\{a_1, a_2, a_3, a_4, a_5\}$ and B's actions at state s are $\{o_1, o_2, o_3, o_4, o_5\}$.

$$\pi(a_1, o_1) * Q(a_1, o_1) \geq \pi(a_1, o_1) * Q(a_2, o_1)$$

$$\pi(a_1, o_1) * Q(a_1, o_1) \geq \pi(a_1, o_1) * Q(a_3, o_1)$$

$$...$$

$$\pi(a_1, o_2) * Q(a_1, o_2) \geq \pi(a_1, o_2) * Q(a_1, o_2)$$

$$...$$

$$\pi(a_5, o_5) * Q(a_5, o_5) \geq \pi(a_5, o_5) * Q(a_4, o_5)$$

Which can be extended to multiple players by the family of inequalities (this notation was taken from TA's at Georgia Tech):

$$\sum_{a_{-i} \in A_{-i}(s)} \pi(a) Q_i(s, a) \geq \sum_{a_{-i} \in A_{-i}(s)} \pi(a) Q_i(s, (a_{-i}, a_i'))$$

Greenwald also shows that [5] generalizes [3] experimentally through a series of tests on grid games. One of Greenwald's experiments included a simplified version of the Soccer game that Littman used as a game environment in his friend vs foe paper.

3 Soccer

The simplified version of the game of Soccer that Greenwald used in her experiments consists of two players (A and B) and a 2x4 game grid. The leftmost column of the grid signifies the goal for player A and the rightmost column signifies the goal for player B. If player A reaches the leftmost goal with possession of the ball, then each player will be rewarded with one hundred points. If player B reaches the rightmost column with possession of the ball, then each player will rewarded negative one hundred points. Therefore, the game is a zero-sum game with player A trying to maximize the reward and player B trying to minimize the reward.
Table 1 illustrates an example of a state that the game of soccer could be in. The circle around A is used to indicate that player A has possession of the ball. Table 1 also illustrates a stochastic point in the game where certain actions could potentially lead to a possession change. For example, if player B moves North (relative to the grid) and player A moves West, then both players would be attempting to occupy the same square. Since both players cannot occupy the same square at once, one player is randomly chosen to move first. If player A moves second, then player B will obtain possession of the ball. This is because possession is given to the player who enters a square first, if both players attempt to move to the same square. If a player chooses to stay in the same square, they are automatically chosen to move first and will remain in the same square if another player attempts to move in the already occupied square.
Greenwald left out several details about the game including how the initial state s_0 is set. Instead of guessing or randomly initializing s_0, I looked at how Littman initialized s_0 in his version of the soccer game. Littman stated that the positions of players A and B were fixed at s_0, but the possession of the ball was randomly given to A or B. Using this information, I defined s_0 to be the state portrayed by Table 1.

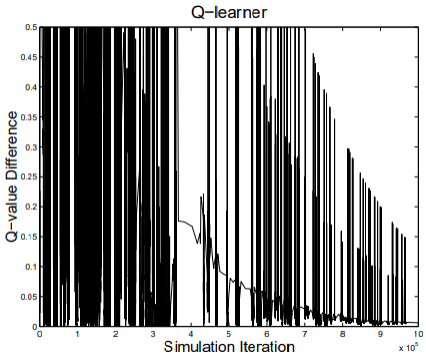

Fig. 1

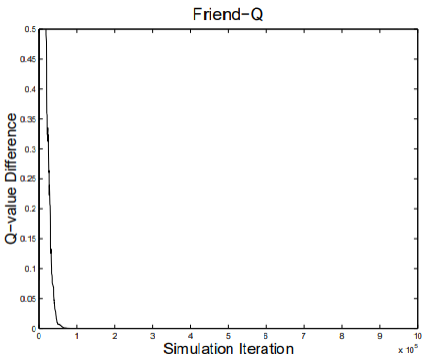

Fig. 2

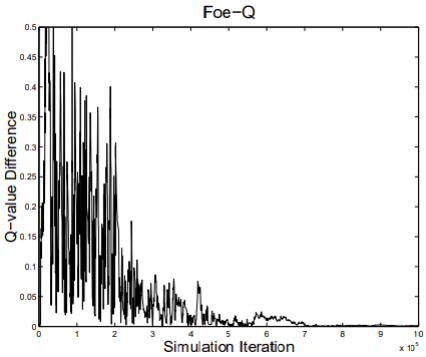

Fig. 3

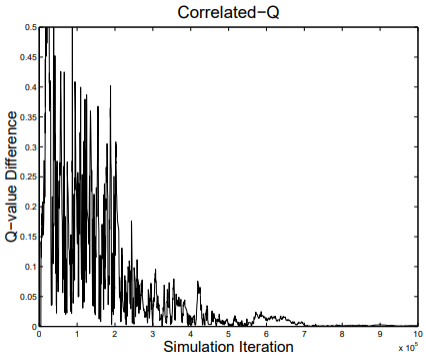

Fig. 4

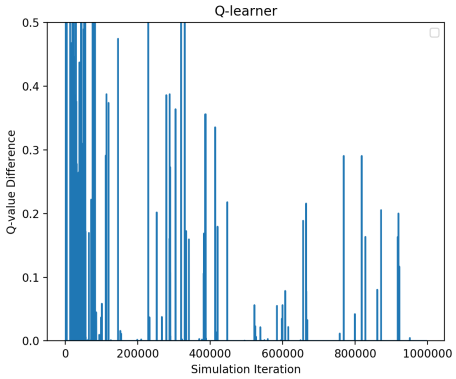

Fig. 5

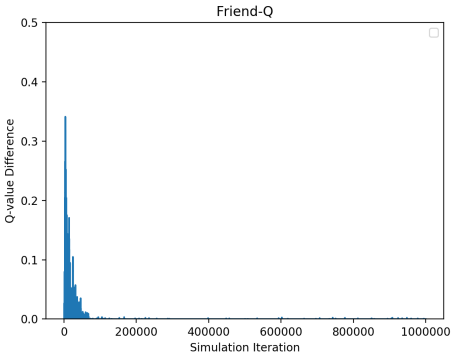

Fig. 6

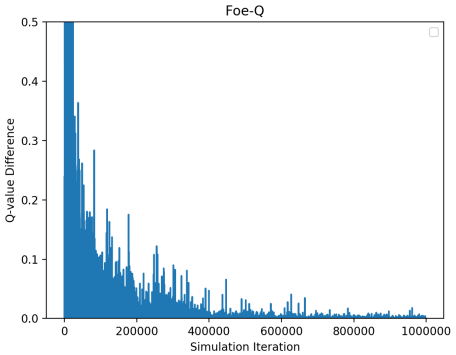

Fig. 7

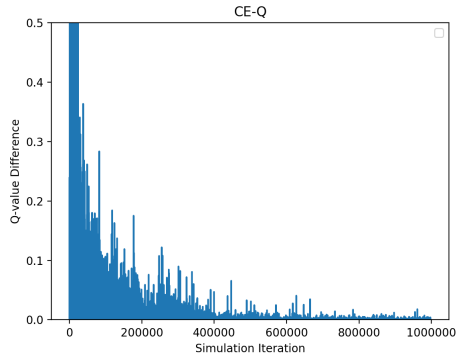

Fig. 8

TABLE 1: Example State in Soccer Grid

Player A Goal		Ⓐ	Player B Goal
Player A Goal	B		Player B Goal

. To train agents A and B to play the Soccer game, the following Q-Learning algorithm from [Greenwald, 2004] was used to calculate the policies for both players:

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

for $t = 1$ to T
1. simulate actions $a_1, ..., a_n$ in state s
2. observe $R_1, ..., R_n$ and next state s'
3. for $i = 1$ to n
(a) $V_i(s') = f_i(Q_1(s'), ..., Q_n(s'))$
(b) $Q_i(s, \vec{a}) = (1 - \alpha)Q_i(s, \vec{a})$
$+\alpha(R_i + \gamma V_i(s'))$
4. agents choose actions $a'_1, ..., a'_n$
5. $s = s'$, $a_1 = a'_1, ..., a_n = a'_n$
6. decay α according to S

4 Results

Figures 1-4 are the experimental results that Greenwald achieved in her study regarding Correlated learning. Specifically, they match the graphs a, b, c, and d in figure 3 of her paper. From here after, Greenwalds graphs a-d will be referred to as figures 1-4 correspondingly. Figures 5-8 are the experimental results achieved when trying to replicate her experiments by following the procedures outlined in her paper. The Q-value difference on the y-axis for all of the figures is the error in terms of $E_i^t = |Q_i^t(s, \vec{a}) - Q_i^{t-1}(s, \vec{a})|$

One important topic that reoccurred numerous times throughout her paper was the claim that the strong convergence properties of Q-learning cease to hold in a certain class of games [Greenwald, 2004]. Greenwald shows evidence of this claim by training two Q-learning agents (A and B) to play against each other in the soccer environment. Figure 1 shows that the Q-values for agent A never converge and only appear to converge because the learning rate decreases as the number of simulations increase. Figure 5 is an attempt to replicate figure 1, which also shows that the traditional Q-learning algorithm is not sufficient for the Q-values to uniquely converge. One noticeable difference between figures 1 and 5 is the frequency of the error fluctuations. The reason for this difference could be a subtle difference in how the soccer game environment was calculated or if there difference in how the error was accumulated over time. The latter was not specified in Greenwald's paper. However, the important piece figures 1 and 5 is that they both show that Q-learning fails to converge in this type of game environment.

On the contrary, Greenwald shows in figures 2, 3, and 4 that using a different value function causes Q-learning to converge. Figures 3 and 4 are actually almost identical because of the fact that her definition for uCE generalizes the value function of Littman's minimax value function, which is also true for figures 7 and 8. Although figures 7 and 8 don't exactly replicate figures 3 and 4, the graphs are identical in showing a logarithmic drop in the Q-value error. Greenwald did not specify who she chose as the opponent for player A in her experiments, which could be why the error vacillations are more extreme in her figures. I ran the Foe-Q and CE-Q experiments against an opponent with a random strategy, where as Greenwald may have ran the experiment against the same type of player. Figure 2 was easier to replicate than figures 3 and 4 because there was no linear programming involved and the value function was more straight forward. Figure 6 is quite similar to figure 2 in shape and convergence timeliness.

Although Greenfield did not show a general result for convergence, she did show strong empirical evidence for convergence. This is supported by the replicated figures in this paper.

5 Git Hash

1e99244781a373456c23014610a0f287337a3879

References

[1] Amy Greenwald, Keith Hall. Correlated-Q Learning. arXiv preprint arXiv:1711.05225, 2004.
[2] Michael Littman. Friend-or-Foe Q-learning in General-Sum Games. arXiv preprint arXiv:1711.05225, 1994.
[3] Michael Littman. Markov games as a framework for multi-agent reinforcement learning. arXiv preprint arXiv:1711.05225, 2001.