Machine Learning: Project
Multi-Agent Learning in Canonical Games and Knights Archers Zombies

March 23, 2025

Dimitrios Mystriotis - Jan Cichomski
r1027781 - r1026448

---

1 Task 1:

1.1 A Comprehensive Survey of Multiagent Reinforcement Learning

From this paper we are interested mostly in the e-Greedy algorithm and the Boltzmann algorithm. This algorithms were implemented as described in the paper and tested on the canonical games. E-Greedy is a simple algorithm that chooses the best action according to the greedy policy 1 with probability 1 - ϵ and a random action with probability ϵ.

$$\bar{h}(x) = argmax_{a \in A} Q(x, a) \tag{1}$$

where $Q(x, a)$ is the Q value of the action a in the state x, and A is the set of all possible actions. The use of a probability ϵ allows for exploration of the problem space and satisfies the requirement for convergence that the agent needs to keeps trying all actions in all states with nonzero probability.

The Boltzmann algorithm is a more complex algorithm that chooses the next action with probability

$$P(a) = \frac{e^{Q(x,a)/\tau}}{\sum_{a' \in A} e^{Q(x,a')/\tau}} \tag{2}$$

where τ is the temperature parameter. The Boltzmann algorithm is a stochastic algorithm that allows for exploration of the problem space. The temperature parameter τ is responsible for the amount of exploration the algorithm will perform. A high τ value will make the algorithm explore more, while a low τ value will make the algorithm exploit more.

1.2 Evolutionary Dynamics of Multi-Agent Learning: A Survey

From this paper we are interested in the Lenient Boltzmann algorithm. The Lenient Boltzmann algorithm is a modification of the Boltzmann algorithm that evaluates actions

more optimistically. This is done by letting the agent evaluate their action against k possible actions from the opponent and keeping the best reward. In this way potential mistakes in the start of the training are not punished as hard as in the Boltzmann algorithm.

Replicator dynamics are also explained in this paper. The replicator dynamics are a set of differential equations that describe the evolution of the population of strategies in a game. Specifically, the model the change in the frequency of strategies in a population. This population can be described as a state vector $x = (x_1, x_2, ..., x_n)$ where x_i is the frequency of strategy i in the population. If the fitness of these strategies is given by the fitness function $f(x)$, the replicator dynamics are given by the equation

$$\dot{x_i} = x_i(f_i(x) - \bar{f}(x)) \tag{3}$$

where $f_i(x)$ is the fitness of strategy i and $\bar{f}(x)$ is the average fitness of the population.

2 Task 2:

2.1

- A game is in Nash equilibrium when no player can improve their outcome by changing their strategy, if the other player doesn't change their's.
- A game is in Pareto Optimal when it is impossible to make a player better off without making the total payoff worse.

(a) Stag Hunt:

- Nash Equilibria: (Hare,Hare) and (Stag,Stag)
- Pareto Optimal: (Hare,Hare) and (Stag,Stag)

(b) Subsidy game:

- Nash Equilibria: (Subsidy 2,Subsidy 2)
- Pareto Optimal: (Subsidy 1,Subsidy 1) and (Subsidy 2,Subsidy 2)

(c) Matching Pennis:

- Nash Equilibria: There is no Nash Equilibria for a pure strategy. For a mixed strategy, the Nash Equilibria is picking heads or tails with probability 0.5 each.
- Pareto Optimal: Every outcome is Pareto Optimal

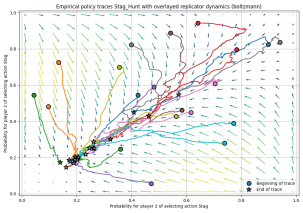
(d) Prisoner's Dilemma:

- Nash Equilibria: (Defect,Defect)
- Pareto Optimal: (Cooperate,Cooperate)

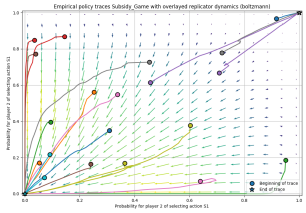
2.2

The e-Greedy algorithm converges to the Nash Equilibrium for all games. Even when an algorithm converges to the Pareto Optimal solution, it starts to move towards the Nash Equilibrium as the number of iterations increases. This is because the Nash Equilibrium is the most stable solution, which random choices will favor. For the learning trajectories of e-Greedy, the action choice depends on q values and it is picked deterministically. This makes it complicated to plot the learning trajectories, as the probabilities to chose an action are constant making it hard to plot a graph.

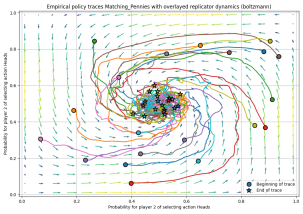
The Boltzmann algorithm converges to either the Nash Equilibrium or the Pareto Optimal solution. The Boltzmann algorithm is more likely to converge to the Pareto Optimal solution compared to the e-Greedy algorithm, and it doesn't deviate from that point. The algorithm will basically find an optimal solution (either Nash or Pareto) and stick to it. Which on is favored depends on the starting distance from the point and the rewards of the game. It seams to favor the Nash Equilibrium slightly more than the Pareto Optimal solution.



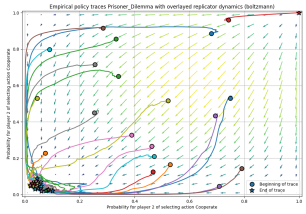
(a) Boltzmann Q-Learning Stag Hunt



(b) Boltzmann Q-Learning Subsidy Game

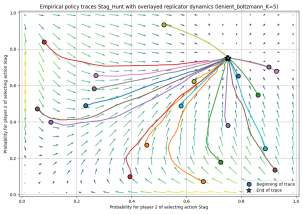


(c) Boltzmann Q-Learning Matching Pennies

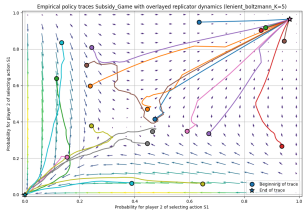


(d) Boltzmann Q-Learning Prisoner's Dilemma

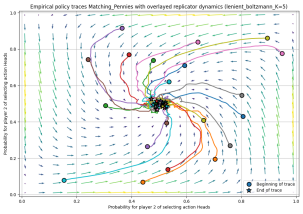
For the Lenient Boltzmann Q-Learning algorithm, the results are similar to the Boltzmann Q-Learning algorithm. The main advantage of the Lenient Boltzmann Q-Learning algorithm is that it evaluates multiple actions when performing an action thus allowing for better exploration of the problem space. This can be seen in the plots, where the algorithm is more likely to explore the problem space and find the Pareto Optimal solutions. The existence of the k value allow for the algorithm to explore more or less, depending on the value of k.



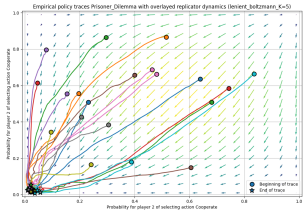
(a) Lenient Boltzmann Q-Learning Stag Hunt



(b) Lenient Boltzmann Q-Learning Subsidy Game

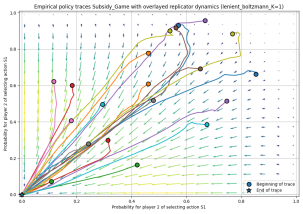


(c) Lenient Boltzmann Q-Learning Matching Pennies

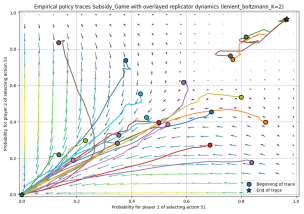


(d) Lenient Boltzmann Q-Learning Prisoner's Dilemma

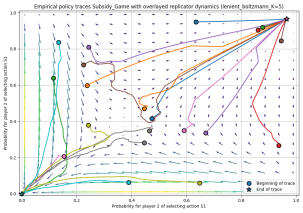
By changing the k value of the Lenient Boltzmann Q-Learning algorithm, the learning trajectories change. Higher k values provide significantly better results which can be seen in the plots 3a - 3d.



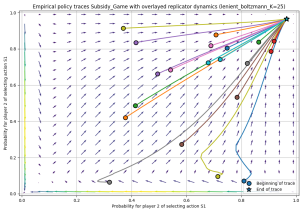
(a) Lenient Boltzmann Q-Learning Subsidy Game (K=1)



(b) Lenient Boltzmann Q-Learning Subsidy Game (K=2)



(c) Lenient Boltzmann Q-Learning Subsidy Game (K=5)



(d) Lenient Boltzmann Q-Learning Subsidy Game (K=25)

2.3

The learning trajectories are behaving as expected and do follow the replicator dynamics. For each of the games learning follow the dynamics, especially for the Lenient Boltzmann Q-Learning algorithm were the changes of the k value affects both dynamics and trajectories making the match between the two very visible. The results can be seen in the figures above 3a were the effect of the k value can shift the learning trajectories to converge to a better solution even if it's not the Nash Equilibrium.

3 Task 3:

4 Task 4: