Ship Docker Problem

A MARL Perspective

Bhavin Jawade
Vamshi Gujjari

Computer Science and Engineering

University at Buffalo
School of Engineering and Applied Sciences

Multi Agent Reinforcement Learning

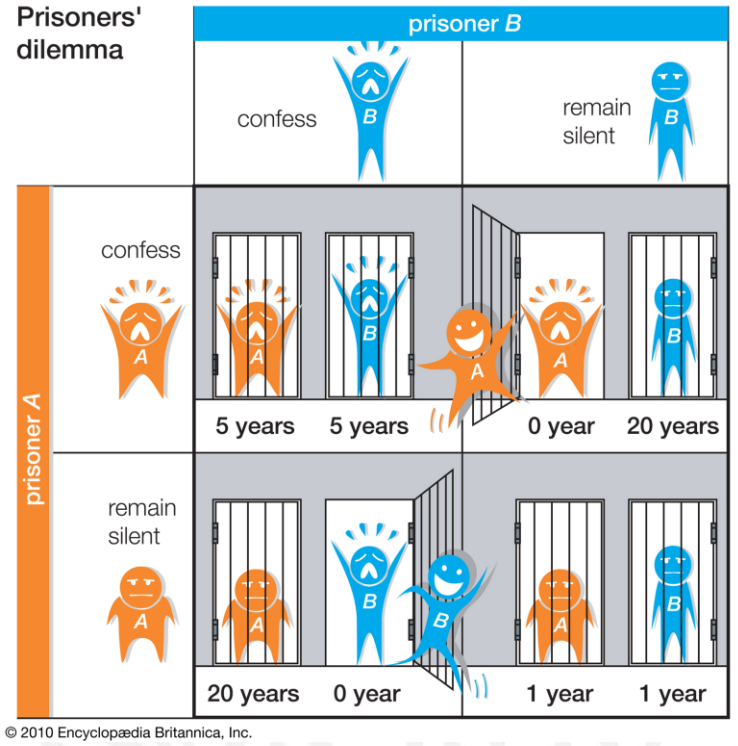
A Brief overview

Matrix Games

Matrix Games also known as Strategic games

Matrix games are the ones that have multiple agents where each agent has its own set of actions, but the environment has only one state with an associated reward structure.

Example: Rock, paper, scissor or Prisoner's Dilemma



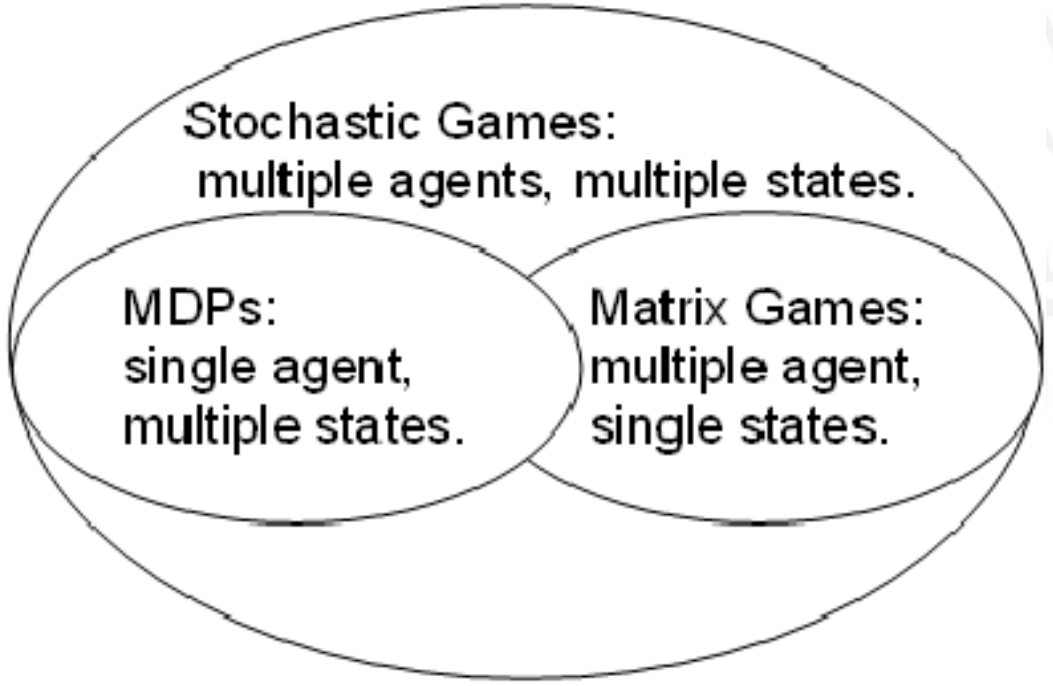
Stochastic Games

Stochastic Games also known as Markov Games

Generalization over Matrix Games, and MDPs

Joint Policy:

$$\pi = [\pi_i, \pi_{-i}]$$



Types of Stochastic Games

1. Zero-sum games
2. Team games
3. General-sum games
4. Iterated games

Optimality in Stochastic Games

a is the joint action

$$V_i^{\pi}(s) = E\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}^i \middle| s_t = s, \pi\right\} =$$

$$= \sum_a \pi(s,a) \sum_{s'} T(s,a,s') \left[R_i(s,a,s') + \gamma V_i^{\pi}(s')\right]$$

Best Response Policy: a best-response policy for player i is one that is optimal with respect to some joint policy off the other players.

Nash Equilibrium: Everyone using their best response policy

What are the classes of methods that solve Stochastic games

Best Response Policies

- Just try to learn a policy that is optimal with respect to the policies of the other players.
  - **Advantage:** Other players might not be having BRP, try to get higher returns
  - **Disadvantage:** Do not adopt easily, difficult to converge against an agent which does not have stationary policy.

Equilibrium learners

- Equilibrium Learners specifically try to find policies which are Nash equilibria for the stochastic games

Best Response Learners: Wolf-PHC

- Win or Learn Fast: policy hill climber

Equilibrium Learners:

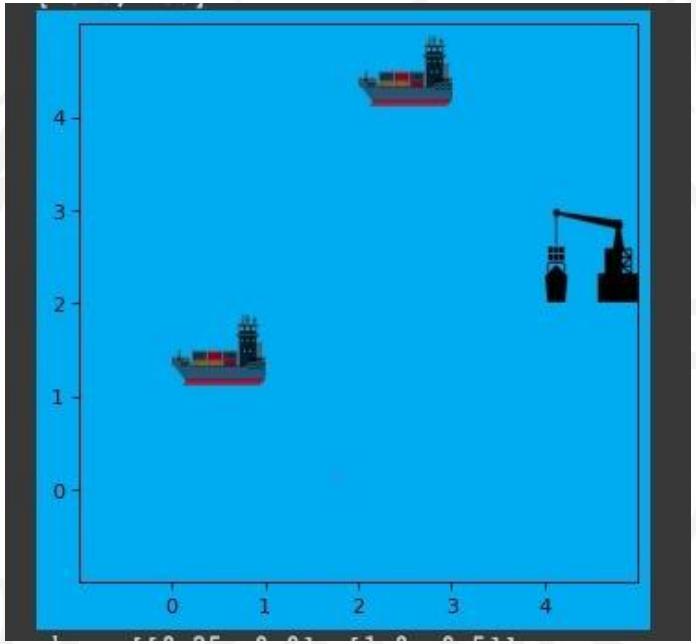
- Minimax-Q
- Nash-Q
- Friend or Foe-Q
- Correlated-Q

Ship Docker Problem

- Container ships wait at the docks for months to unload goods as a result of Global trade expansion.
- Port is said to be efficient, only if the waiting times for all the ships are minimised.
- If there is no centralized scheduling authority, it becomes a challenging problem as the ships start to compete for unloading.
- Without a scheduling authority, efficiency can be achieved only with mutual co-operation between the ships.
- We propose a Multi Agent Reinforcement Learning (MARL) solution to tackle this problem, where every ship behaves as an individual agent.

Environment

- The environment for this problem is taken as a mXm grid world, where there are n ships(agents) and y docking stations(goals).
- For example: scenario in the image, m = 5, n = 2, y = 1.
- In ideal scenario, each agent tries to reach the goal without colliding with the other.
- The agent can perform 4 actions(Right, Left, Up, Down).
- This is a deterministic, fully observable, episodic, discrete environment.



Reward Dynamics

- Reached Goal position: +1000
- Collision: -200
- Staying in the same position: -500
- Moving towards goal: +20
- Moving away from the goal: -20

In some experiments rewards were clipped between 0 and 2, but in same ratio as above. We did try different reward dynamics as well.

State:

Observation for reach agent is represented as: [A1x,A1y,A2x,A2y,G1x,G1y,G2x,G2y]

Explain Multi-agent Actor Critic

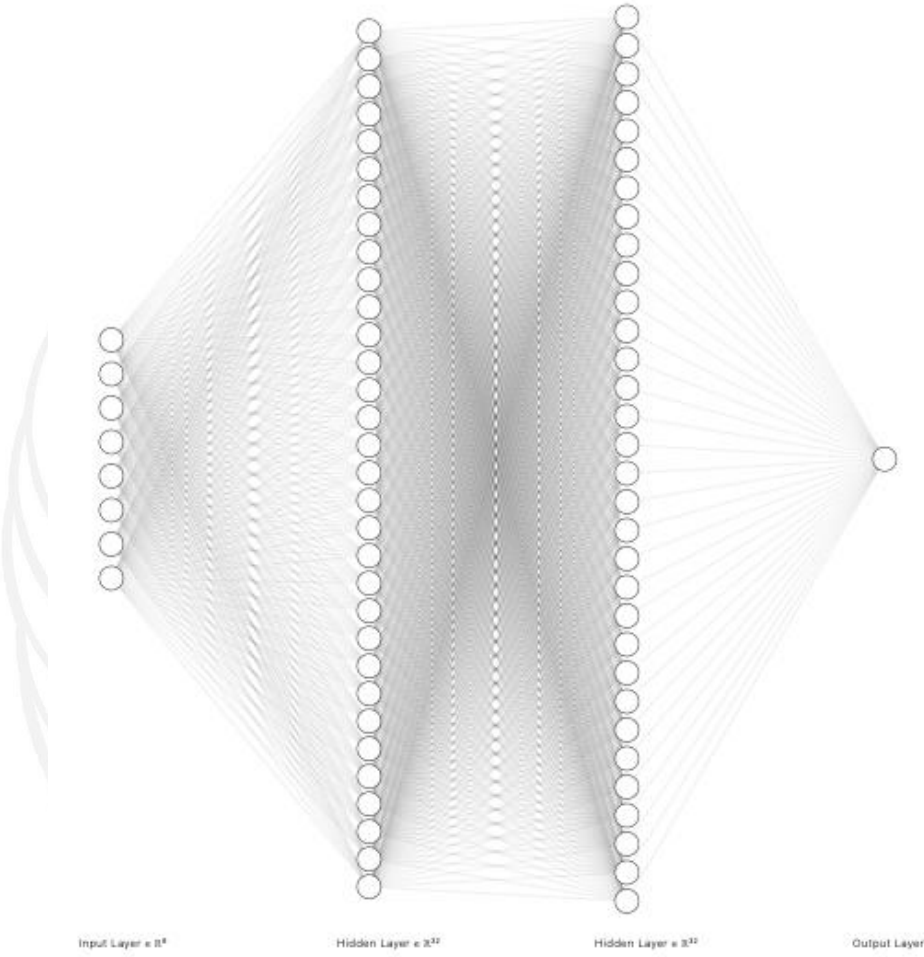
Actor Architecture

Input layer: 8 (A1x, A1y, A2x, A2y, G1x, G1y, G2x, G2y)

Hidden layer 1: 32

Hidden layer 2: 32

Output layer: 4 (4 actions)



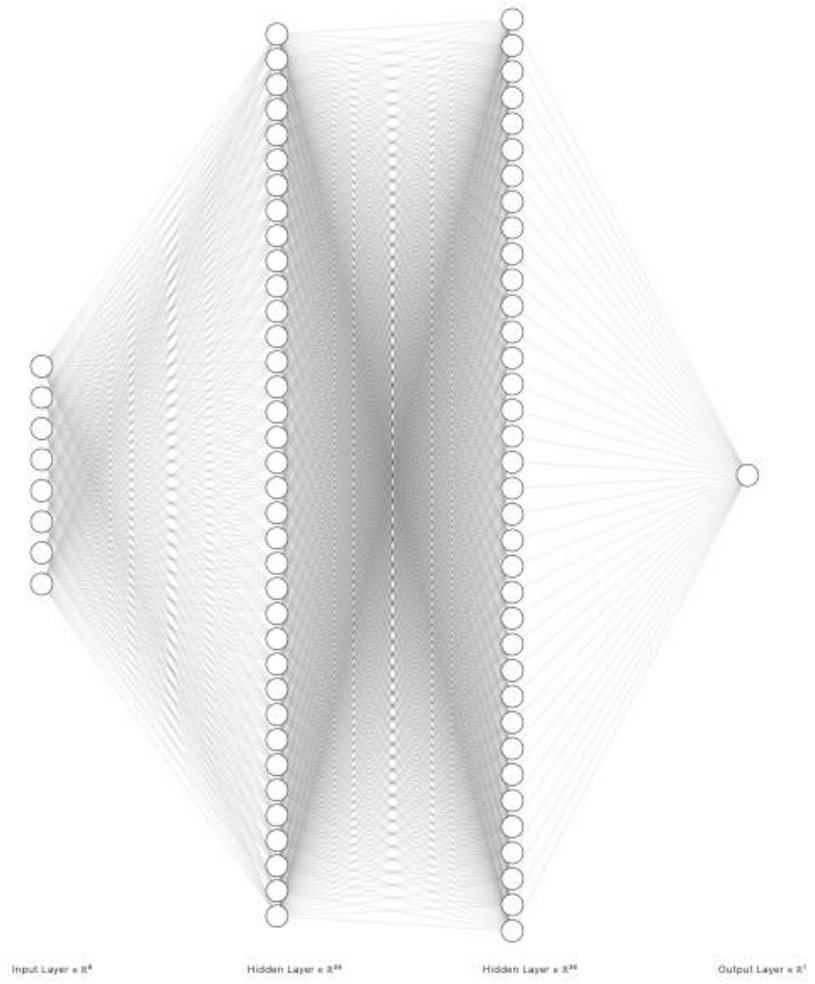
Critic Architecture

Input layer: 8 States

Hidden layer 1: 32 + 4

Hidden layer 2: 32 + 4

Output layer: 1



Number of actor networks = Number of agents

Number of critics = Number of agents

Critics can be concurrent or centralized

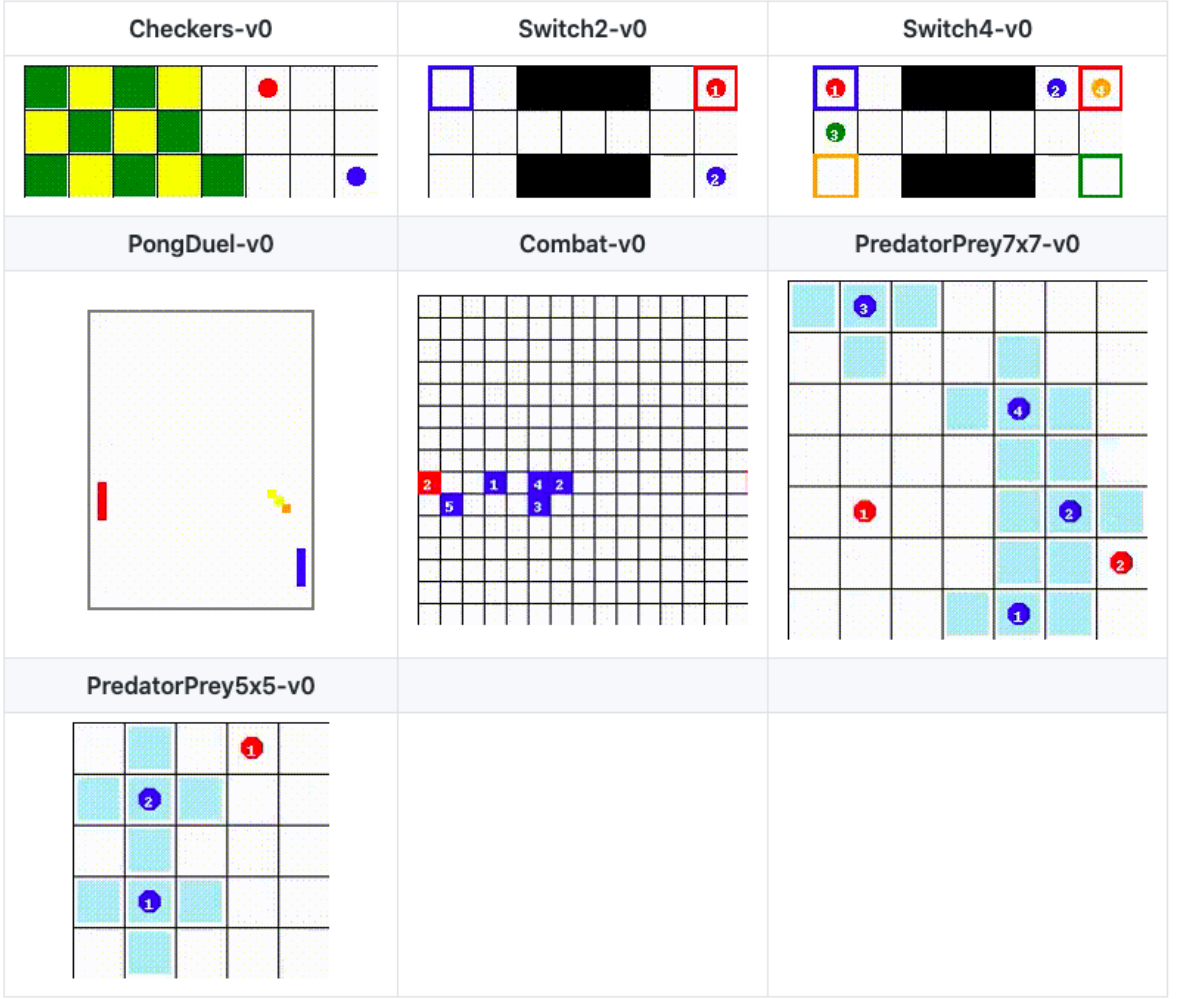
Concurrent: Every critic takes the observation of the specific agent.

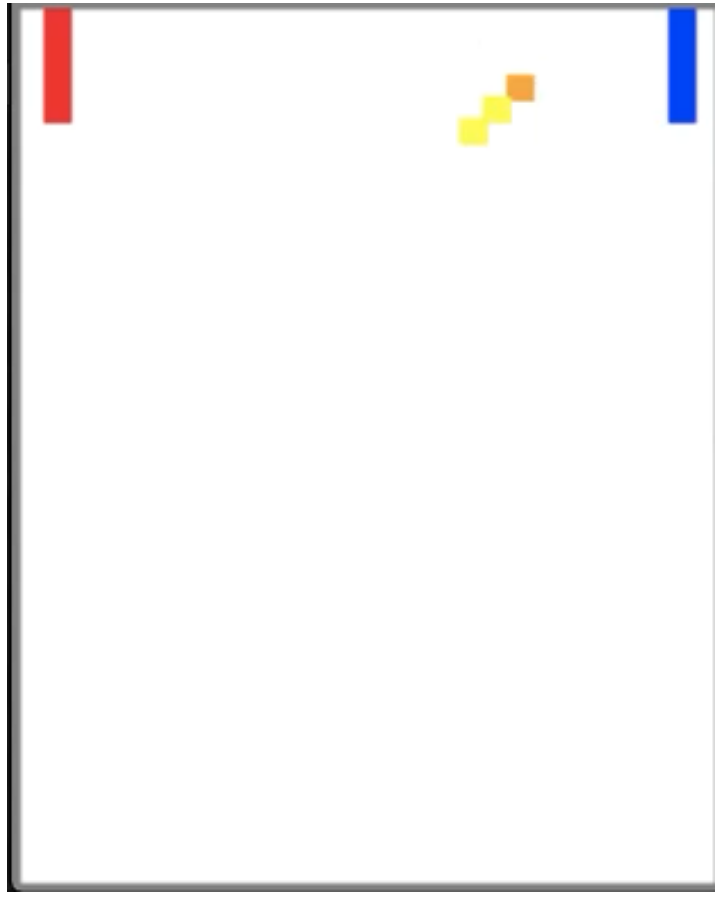
Optimizer: Adam or RMSProp.

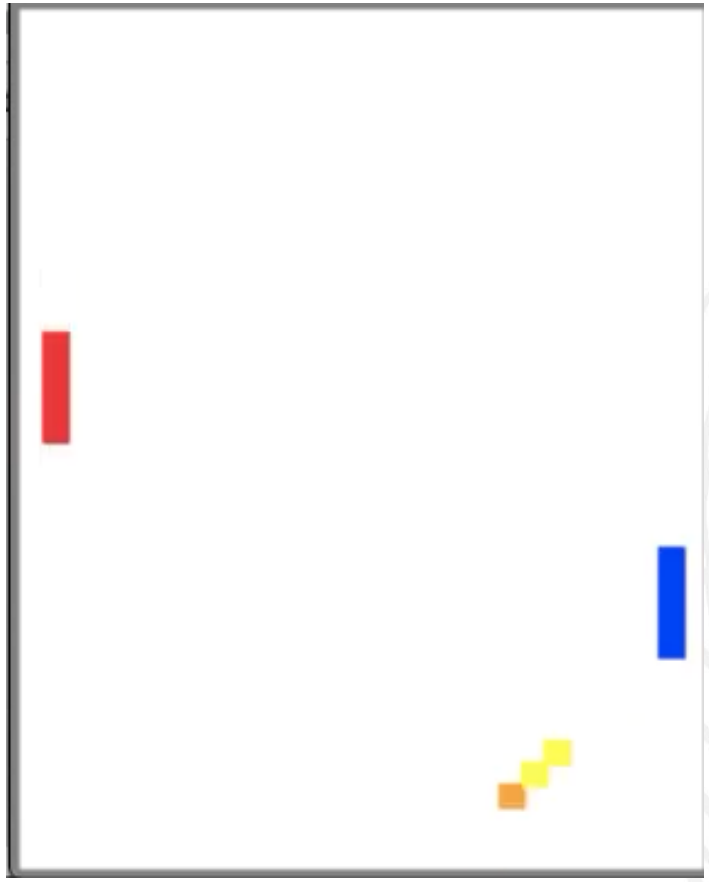
MA-GYM

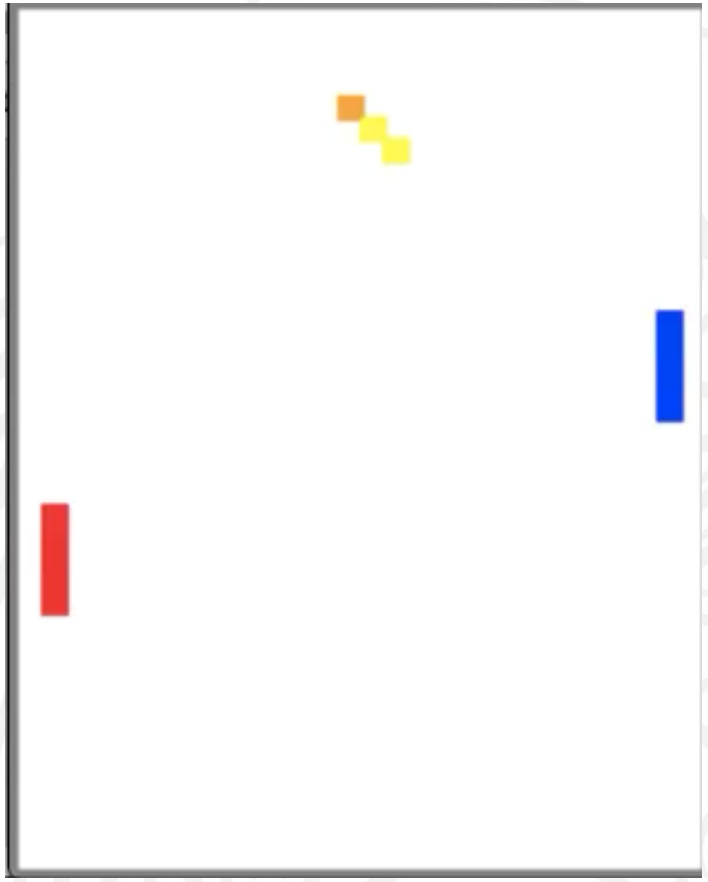
- https://github.com/koulanurag/ma-gym

Solving MA-GYM environment - DualPong





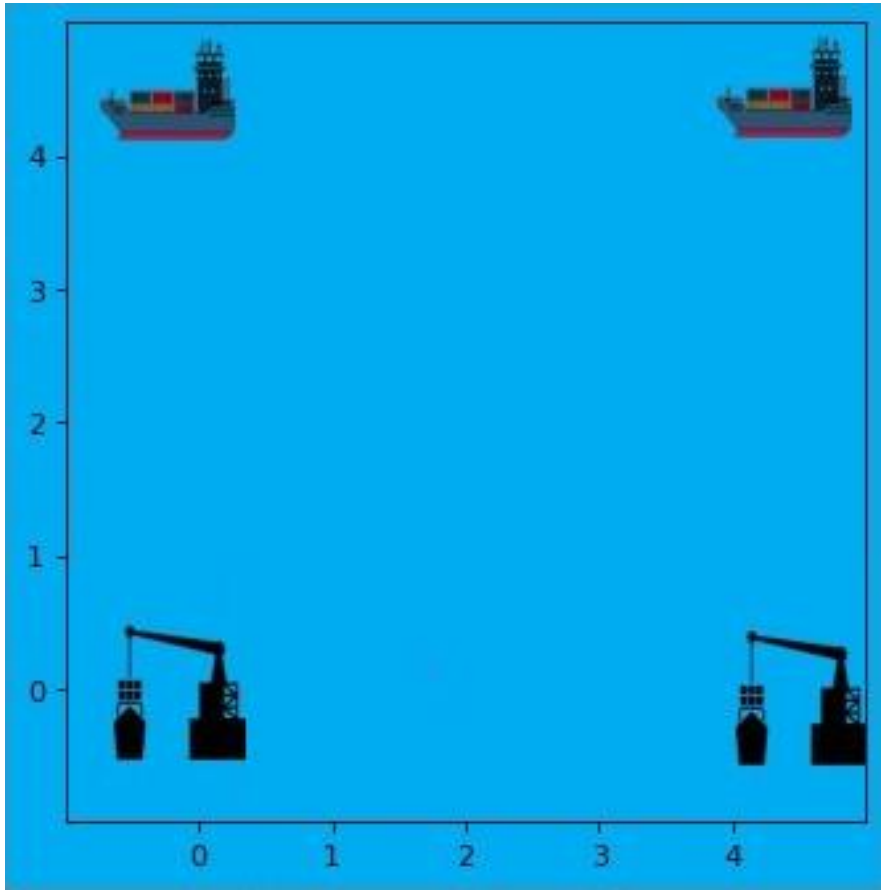




Experiments:

1. Testing our Multi-agent Actor Critic on MA_Gym Dual Pong.
2. Simple 5 * 5 environment with 2 agents 1 end goal.
   a. Different Reward Dynamics
   b. Collision avoidance
   c. Priority Replay Buffer

Result Cases: Nash Equilibrium



What will be the nash equilibrium for this situation?

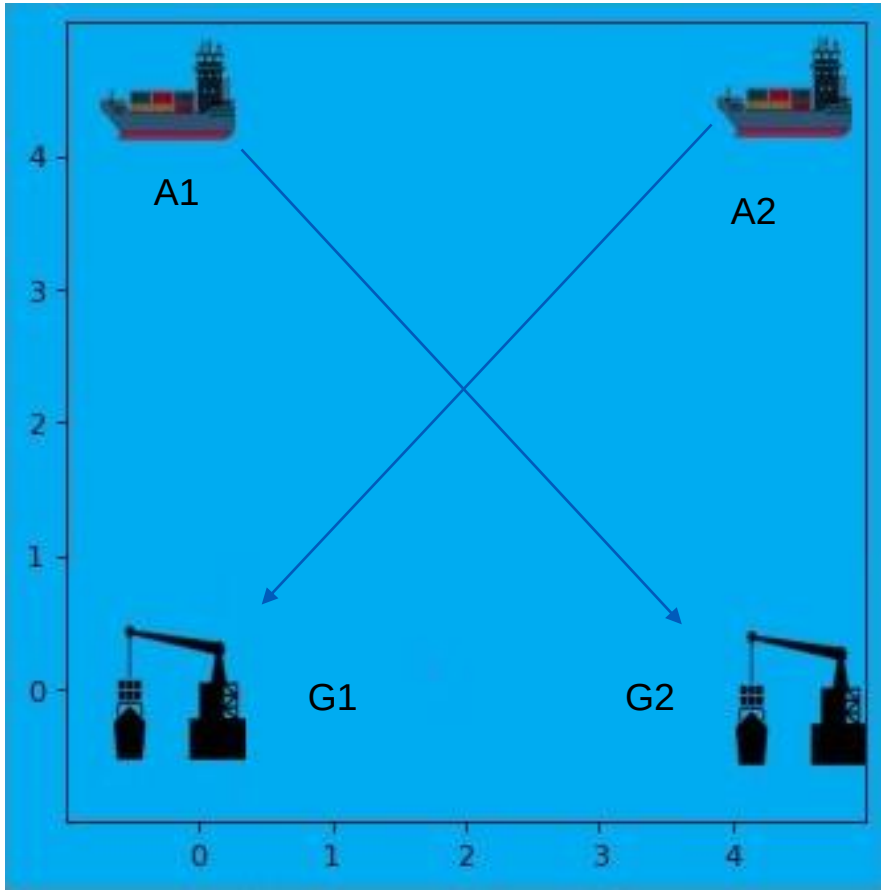
Game ends when both reach a goal, or they collide.

Reward Dynamics:

1. +1 for reaching any goal
2. +0.5 moving towards a goal
3. -1 for colliding

- They will reach the nearest port.

Result Cases: Nash Equilibrium



What will be the nash equilibrium for this situation?

Agent 2 has to reach goal 1 and agent 1 has to reach goal 2. (Specific goals for each agent).

Reward Dynamics:

1. +1 for reaching any goal
2. +0.5 moving towards a goal
3. -1 for colliding

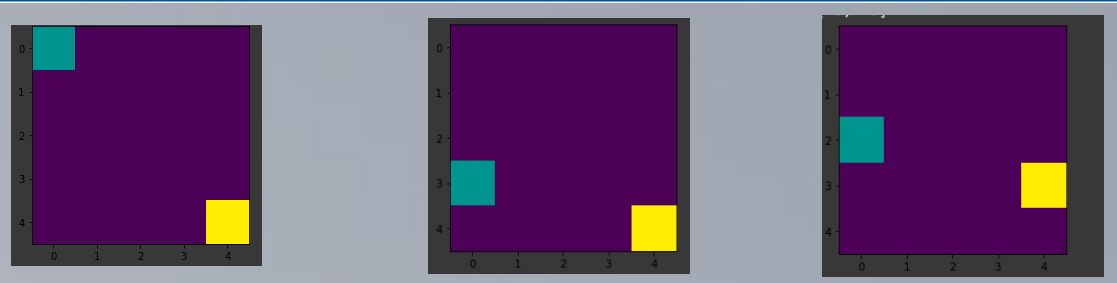
- They will still go to the nearest goal, and stay there for remaining number of timesteps.

Why? Strict Nash equilibrium, suboptimal result as agent tries to converge to a policy which is independent of other agents policy

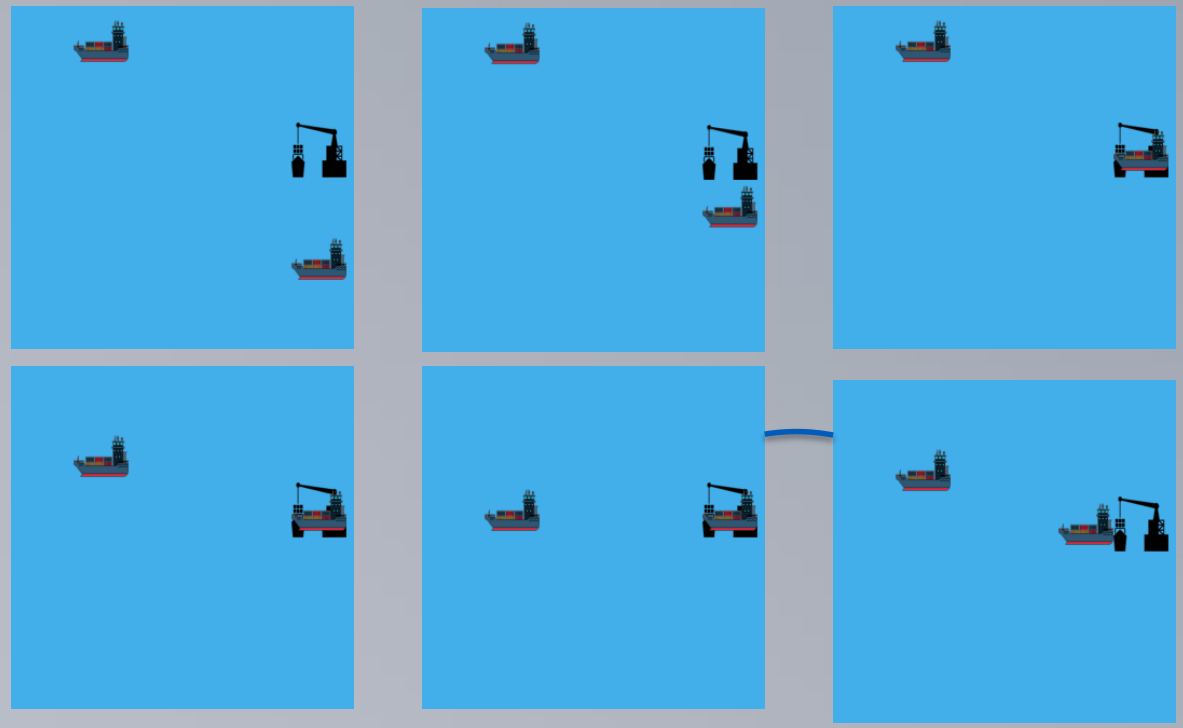
Reward function is extremely important

Intimidation behavior:

The agent closer to goal state reaches the goal, stays there to collect large positive reward. Then moves towards the other agent, so that the other agent takes one step back to avoid collision. This agent then again goes back to goal state to collect large positive reward.

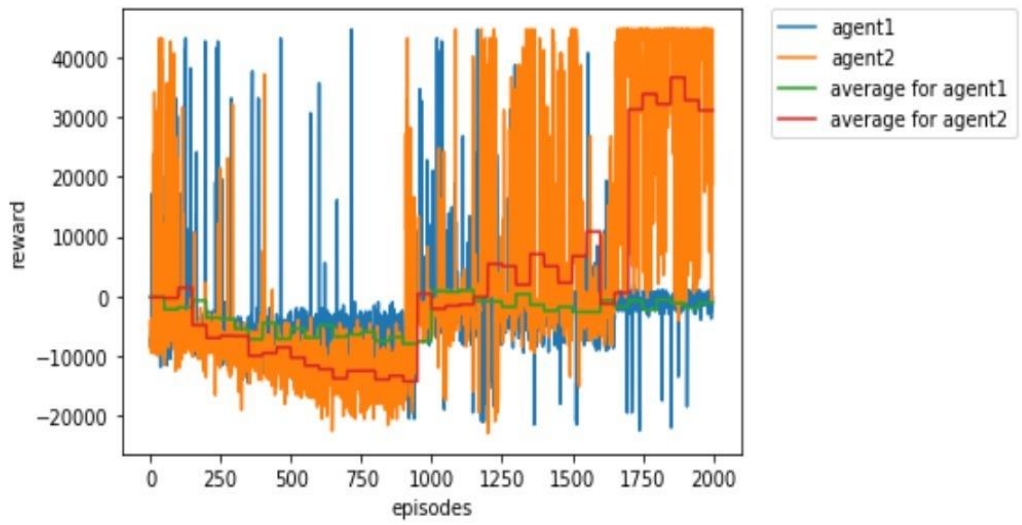


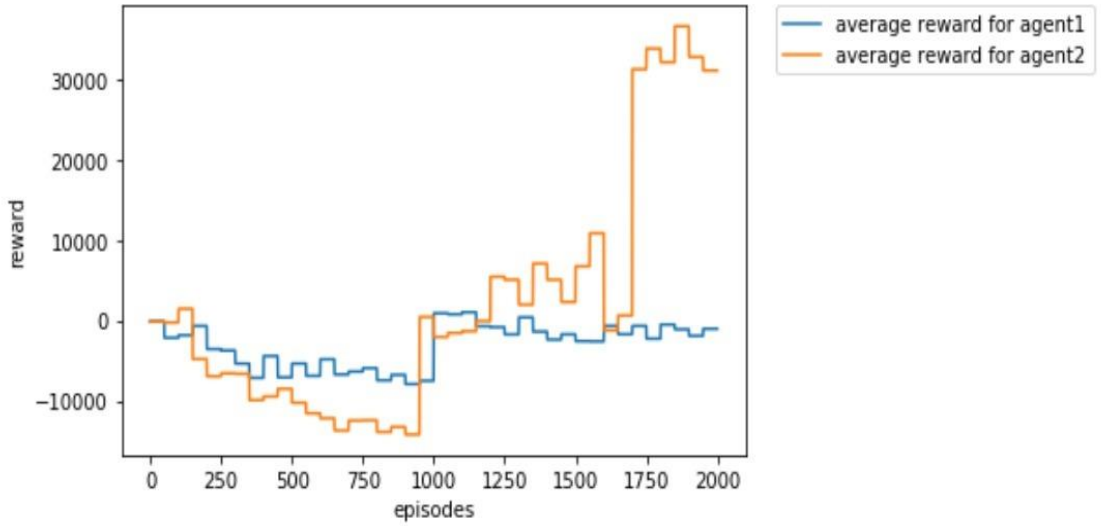
Case 1



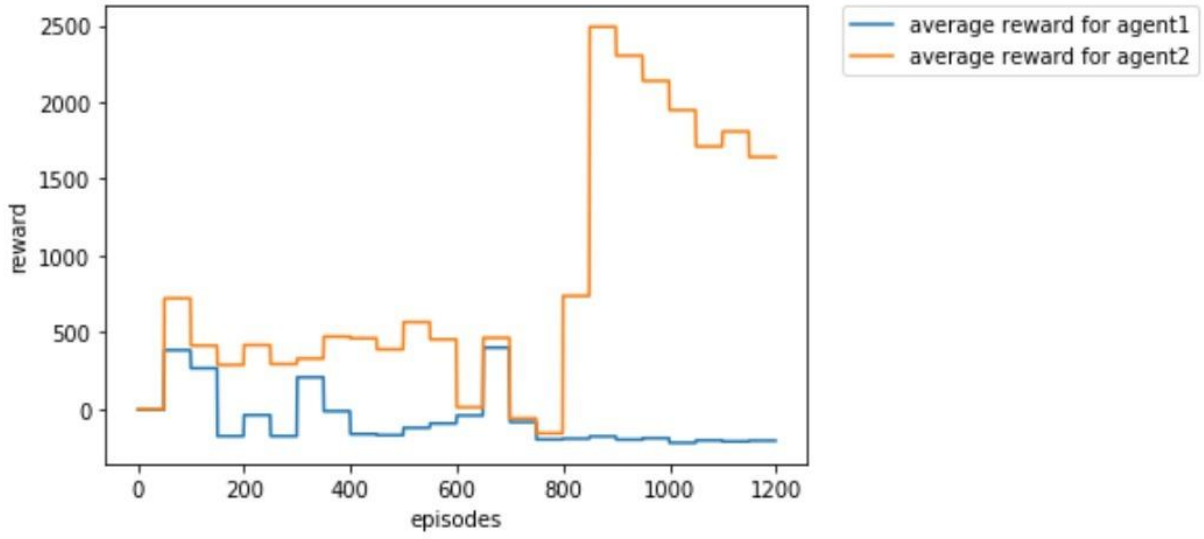
Case 2

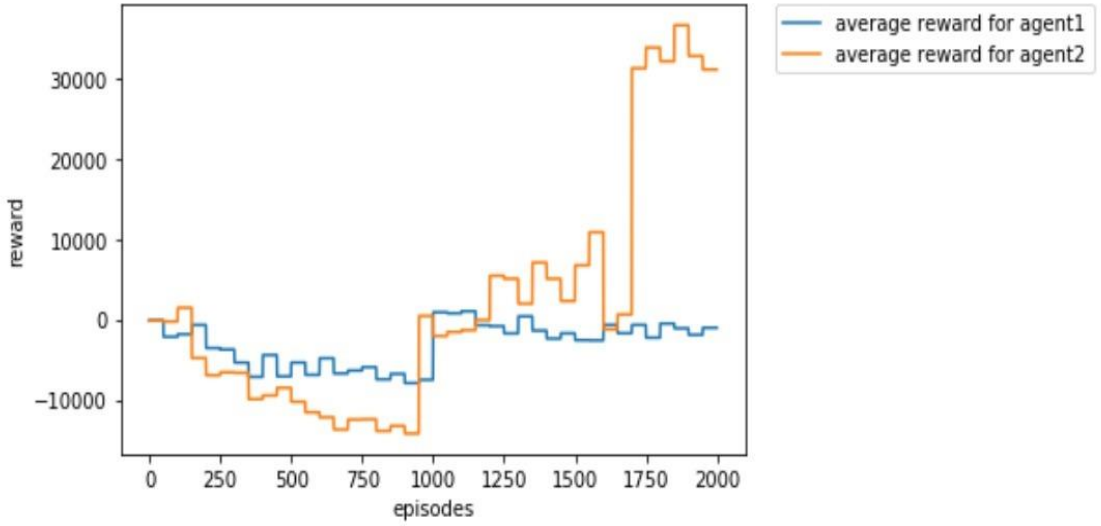
Reward Graphs:



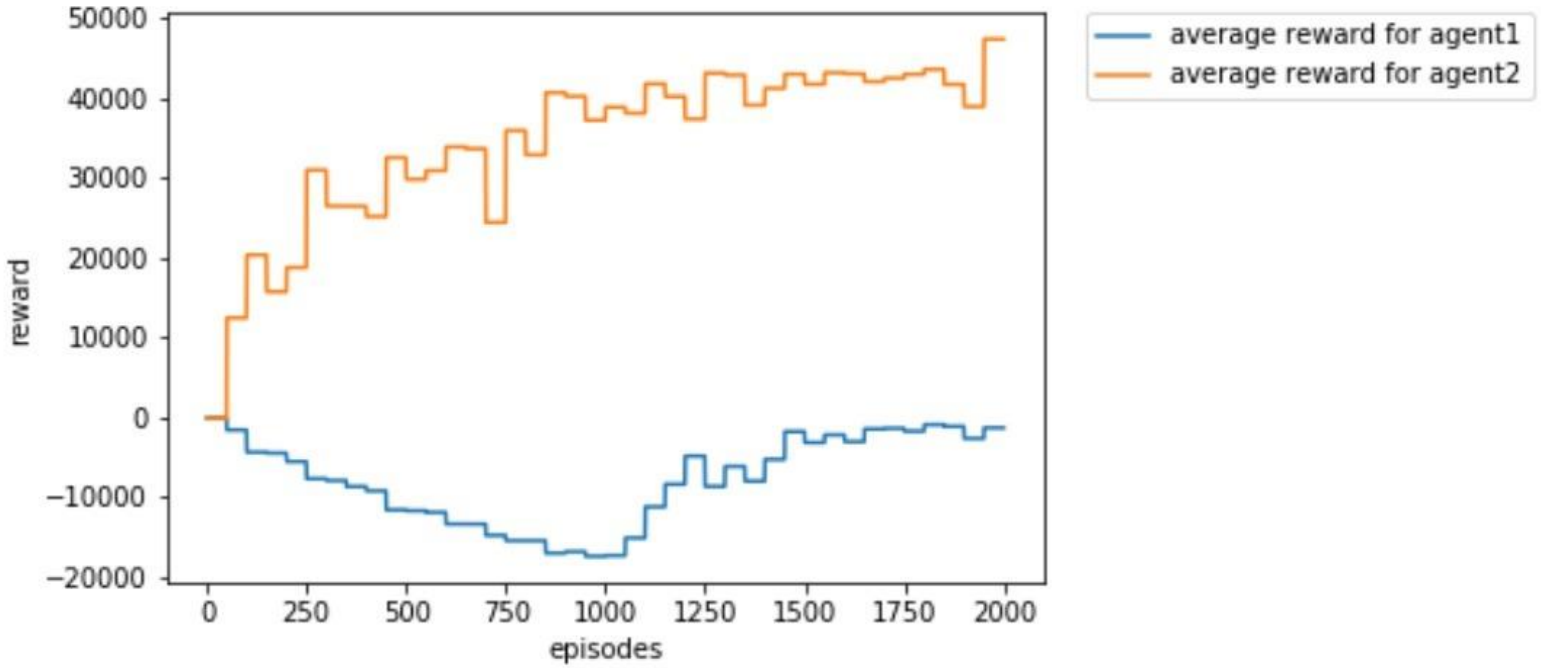


Reward Graphs:





Reward Graphs:



Thank You!

Any Questions

Deepmind Parkour Agent