Assignment 3: Multi-Agent Reinforcement Learning

CSE5022 Advanced Multi-Agent Systems

DDL: 23:59, Jun 1, 2025

1 Overview

This assignment aims to test your ability to use Reinforcement Learning (RL) in Multi-Agent Systems. It encompasses two parts. In the first part, you will implement some Multi-Agent Reinforcement Learning (MARL) algorithms in a Petting Zoo[1] environment. In the second part, you will design your own environment that allows for effective training of MARL algorithms.

2 Part 1: Training MARL Agents

This part requires you to write your own MARL agents to solve a well-known and predefined environment in Petting Zoo. The chosen envrironment is *Pursuit*[2] from the SISL (i.e., Stanford Intelligent Systems Laboratory) environments. A detailed description of the environment can be found in the official documentation page. Your main objective is to examine different MARL algorithms, implement them, and train them to solve the environment. To evaluate the performance of these algorithms, you need to compile a comprehensive comparison analysis and provide a short justification of your observations.

2.1 Objectives

1. **Training:** Implement and train the Independent Q-learning (IQL) and Central Q-learning (CQL) MARL algorithms to solve the *Pursuit* environment. You can find the pseudocode for these algorithms in the MARL book[3].

2. **Evaluation:** Showcase the performance of your agents by running regular evaluations during and after training.

3. **Summary & Analysis:** Summarise and compare the evaluation results.

[1] Petting Zoo: https://pettingzoo.farama.org/
[2] SISL Pursuit environment: https://pettingzoo.farama.org/environments/sisl/pursuit/
[3] MARL book: https://www.marl-book.com/

2.2 Requirements

1. You are required to provide a brief explanation of the fundamental ideas of each algorithm (IQL and CQL) as well as the specific steps you took to adapt them to the *Pursuit* environment. You may refer to the code examples from the previous labs, which provide base implementations for the algorithms. Alternatively, you can extend the Single-Agent version of Deep Q-learning provided by CleanRL[4].

2. While training the agents (both IQL and CQL), you need to implement regular evaluation intervals. Specifically, the evaluation metric should be the cumulative reward that the policy achieves. Remember to turn off any exploration mechanism during evaluation to ensure the agents always select their currently best actions. The results should be visualised in a line chart where the x-axis represents the episode and the y-axis represents the cumulative reward.

3. It is very likely that the original hyperparameters your choose will lead to poor training performance. Although finding the optimal policy is not expected, moderate hyperparameter tuning is still required. For this assignment, it will be good enough if an upward trend in the cumulative reward is observable. To reduce noise, it's recommended to training the agents a few times (5-10) with different random seeds and average the results to get a more representative curve.

4. For the result analysis, you are required to provide a brief justification about the performance of each algorithm. Point out any notable observations you find and try to link them to the characteristics of the algorithms. Then, try to compare the performances of the two algorithms, if any.

3 Part 2: Designing your Own MARL Environment

This part requires you to define a new MARL environment (i.e., state/action space, reward mechanism) to solve the level-based foraging MARL environment. The main objective is to explore different environment specifications and examine how they affect the performance of the MARL algorithms implemented in Part 1.

3.1 Environment definition

Level-based foraging is a simple multi-agent scenario consisting of a grid projection as well as two entities, crops and forager agents. In each time step, each forager can move in one of the four directions, try to harvest a crop on its adjacent tile, or do nothing. Each forager agent and crop has a skill level, and one or more agents can try to harvest a crop if they are positioned next to it. When the sum of skill levels of these agents is not smaller than the crop's level, they will be able to harvest the crop successfully. The goal of the agents is to harvest all crops as quickly as possible. For more information about the scenario, please refer to the MARL book.

[4]DQL by ClearnRL: https://docs.cleanrl.dev/rl-algorithms/dqn/#dqnpy

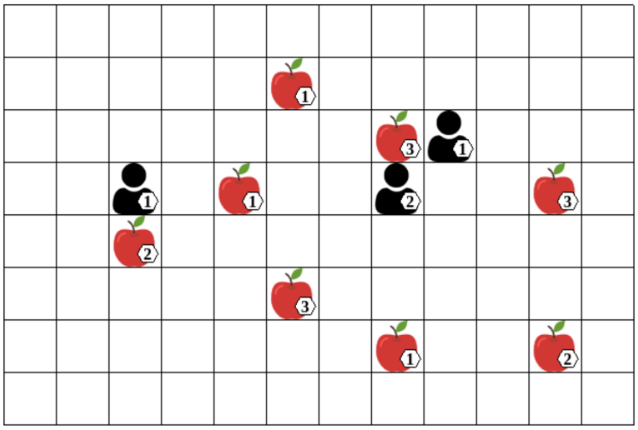

Figure 1: A Snapshot from the Level-Based Foraging Environment

3.2 Objectives

1. **Environment Logic:** Implement the logic of level-based foraging according to the definition provided above.

2. **Observation Space Definition:** Define a suitable observation space that gives each agent enough information to solve the environment.

3. **Reward Function Definition:** Propose two candidate reward functions. You need to consider both local and global rewards as part of your rewards functions.

4. **Evaluation:** Adapt either IQL or CQL to your designed environment and compare the effectiveness of your candidate reward functions based on the achieved learning performance.

3.3 Requirements

1. You are required to write your own implementation of the level-based foraging environment that can facilitate training of MARL algorithms (IQL or CQL). To achieve this, you are encouraged to create a custom petting zoo environment[5], but pure Python implementations are also fine.

2. You are required to create a parallel environment where agents act at the same time. Your implementation should handle any conflicts that can arise from parallel actions.

3. Once you implement the environment logic, you need to propose an appropriate observation space for the agents and two candidate reward functions. You need to justify your design choices in the report.

4. Finally, you are required to evaluate the rewards functions. The evaluation follows the same process as in Part 1. You need to visually compare the learning performance of the agent under each candidate reward function and justify the observed results.

[5]Custom PettingZoo Environment: https://pettingzoo.farama.org/content/environment_creation/

4 What to Submit

1. **A report in PDF format**: Clarify how you fulfil the aforementioned requirements. Use screenshots for clear demonstrations and include as much detail as possible.

2. **Source code**. Remember to describe in your report how to compile and execute your code.

Pack all files into `SID_NAME_A3.zip`, where `SID` is your student ID and `NAME` is your name (e.g., 11710106_ 张三 _A3.zip).