Final Project Guideline

Data Science & Reinforcement Learning
Graduate School of Data Science
Seoul National University

1 Logistics

Please read the instructions below carefully and follow them faithfully.

- The project is to be performed in groups of 3-4 students.
- There are two environments that each team will be solving. Teams need to solve *both* environments, *not* a choice between the two.
- Each algorithm will be evaluated 3 times in total: 6/8, 6/15, and 6/22 (Final). The deadline for all submissions will be 23:59 (KST) on the date your assignment is due. For the algorithms submitted in time, we will provide a leaderboard of the performance of each algorithm against the test environment.

The score breakdowns are as follows.

- 10 points will be given for each submission regardless of performances. Hence, if you submit an *error-free* code in time (that we can run) for all three submissions, you will receive a total of 30 points even if your algorithm performs poorly.
- 30 points will be given to the final code evaluation (due 6/22) and the performance of the submitted algorithm based on the criteria (see Section 5). The final code should include the algorithm described in the final report. If you use a code (even partially) from any GitHub repo or any other sources, please make sure to provide references to the original source. Failure of references may result in 0 points for this part.
- 40 points will be given to the final report due 6/22. The rigor (and novelty if there is any) of the method used will be counted toward this part.
- In addition to a total of 100 points for the final project, an extra credit will be awarded to the top 3 teams on each of the 6/8, 6/15 and 6/22 leaderboards based the criteria explained in Section 5.

2 Objective

The objective of this project is to provide the opportunity to experiment with RL algorithms. To keep the process fun, interactive, and competitive, we are providing a starter code kit and a leaderboard that will allow the agents developed to compete with each other.

2.1 Task 1: `Chain MDP`

An illustration of a general `Chain MDP` is shown in Figure 1. We provide a Chain MDP of length 10 (i.e., $N = 10$). Episode length is 18. that is, an episode will last 18 steps after starting from the initial state (S_2). There are two possible actions $\{$left:0, right:1$\}$. The agent needs to reach the right end (S_{10}) of the chain and perform the right action in order to receive a reward of 1. A deceptive small reward, 0.001, is given when the agent reaches the left end (S_1) of the chain and performs a left action. Since the reward is sparse, your agent needs "deep" exploration in order to find the optimal policy, which is to collect the high reward on the right end.

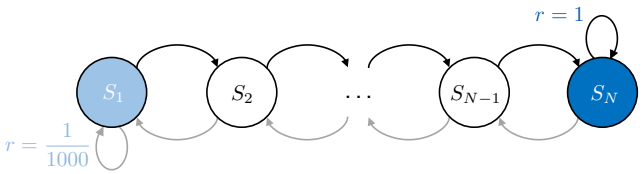

Figure 1: Chain MDP

2.2 Task 2: `Lava`

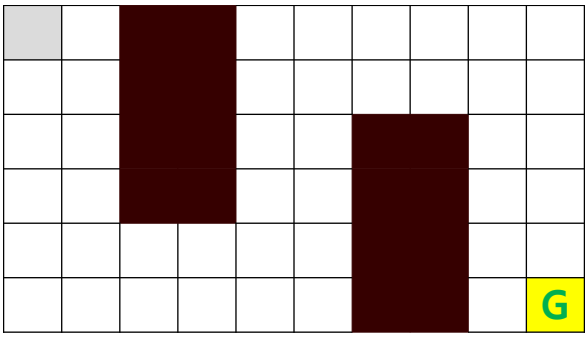

Figure 2: Lava

The second task is to train your agent in the gridworld environment. `Lava` is a 6×10 grid with discrete states and actions, shown in Figure 2. There are four possible actions {left:0, up:1, right:2, down:3} in each state. An agent starts in the top-left corner and must reach the goal state to receive a $+1$ reward while avoiding stepping into the lava (-1 reward) on its way.

Caution: Any heuristic algorithm, e.g. one that always chooses to only go to the right for Task 1, is not what we intended. This type of algorithm is not acceptable.

3 Getting Started

We have uploaded a skeleton code on eTL. The skeleton code consists of two python codes for each task. One is the file in which the environment is implemented (e.g., `lava_grid.py`), and the other is the file that can test the interaction between the environment and your agent (e.g., `lava_test.py`). You will need to write your agent by modifying the designated agent class and methods inside the `agent_lava.py`. You can add methods and classes to this script, but do not change the signature of the placeholder methods provided since it would lead to failure.

4 Leaderboard Submission

Each team (e.g, contact person) will submit their own `agent.py` file for each task to the eTL by 23:59 on 6/8, 6/15, and 6/22 (Final Submission). Two python files (`agent_chainMDP.py`, `agent_lava.py`) must be combined as a zip file, and please set the zip name as `RL_Team##.zip` (## is the team number) (e.g., `RL_Team01.zip`). Please make sure that your code works with the agent class in the test file before submitting. If your agent fails to run with this, it will not run on the evaluation framework either.

5 Code Evaluation

In performance evaluation of algorithms, your agent will be evaluated on 3 categories; **performance**,**sample-efficiency**, and **adaptability** on a new environment. This new environment is a modified version of the environments explained above. For example, the starting point of `Chain MDP` (or the location of lava) could be changed. We briefly explain what each criterion stands for.

Performance. This score captures the final performance of the trained RL agent submitted by the students. The submitted (learned) agent will be evaluated for a fixed number of steps for evaluation. We evaluate it on the same task/environment explained above on a set of 5 seeds for 50 episodes and the final performance is the mean over this.

Sample Efficiency. This score tries to capture how fast an agent learns starting from randomly initialized weights (i.e. the sample efficiency). We randomly initialize the weights of the agent (if applicable) submitted by the students and train it for 1,000 episodes with a fixed horizon length for `ChainMDP` (3,000 episodes for `Lava`). We repeat this process over 20 seeds and the final sample efficiency score is the Area Under the Curve (AUC) of the mean evaluation performance.

Adaptibility. (Only for 6/22 submission) This score is to evaluate a trained agent's capacity to adapt to a new held out task. This new task is a modified version of the environment originally provided. This evaluation will only be performed during the final evaluation stage (6/22 submission), but not during leaderboard submissions (on 6/8 and 6/15) for the rest of the submission window. The trained agent will be re-trained on this new task for a fixed number of steps and sample efficiency and final performance will be measured as mentioned before. We will follow the same evaluation procedure as above, but for a slightly different task. To test if your agent has the adaptability to a new environment, you need to modify the environment as you wish and aim to train an agent that is adaptable to the new environment.

Note: The team will be ranked for each criterion. To calculate a score for each team, we average the team ranks on all criteria, respectively. For example, if the team is placed the 1st on Performance and the 2nd on Sample Efficiency, then the score will be 1.5.

6 Report

For the final evaluation, please submit a 3-4 page project report (without references) with your final submission on 6/22. This covers the methods, motivation behind choosing them and the final results demonstrating performance, sample efficiency.