ReadMe

(Multi-agent Reinforcement Learning for Decentralized Stable Matching)

There are two .py files submitted:

1. MARL.py : This is the main .py file containing the implementation of multi-agent reinforcement learning approach. It outputs matches and rewards after the episode if any agents are matched at the end of that episode.
2. Instances.py : It contains lists of the instances of all the types. There are 10 instances of each problem(and preference structure) type. This file should be imported in MARL.py.

MARL.py imports several other python packages, mainly, PyTorch. You can install it as:

pip install torch

You can run the program by using following command:

python3 MARL.py

This program takes following command line arguments:

1. Argument: eps, default=350000, type=int, Description: Episodes.
2. Argument: steps, default=600, type=int, Description: Steps.
3. Argument: lr, default=0.00004, type=float, Description: Learning rate (at the start).
4. Argument: df, default=0.9, type=float, Description: Discount factor.
5. Argument: rb, default=40, type=int, Description: Replay buffer size in episodes.
6. Argument: bs, default=500, type=int, Description: Batch size.
7. Argument: epochs, default=5, type=int, Description: Epochs.
8. Argument: hu1, default=50, type=int, Description: Hidden units for layer 1.
9. Argument: hu2, default=25, type=int, Description: Hidden units for layer 2.
10. Argument: instance, default=0, type=int, Description: Instance number between [0,9].
11. Argument: arc, default=0.000006, type=float, Description: Exploration curve parameter affecting decay in exploration rate.
12. Argument: n, default=0.9, type=float, Description: Exploration curve parameter affecting starting exploration rate.
13. Argument: agents, default=10, type=int, Description: Total Agents.
14. Argument: grid, default=4, type=int, Description: Grid size.
15. Argument: minER, default=0.05, type=float, Description: Minimum exploration rate.
16. Argument: decay, default=0.999996, type=float, Description: Learning rate decay multiplier in each episode.
17. Argument: minR, default=-1, type=float, Description: Reward for NOT being in a match.
18. Argument: prefType, default='asym', type=str, Description: Preference Type: Symmetric or Asymmetric, Other options: ‘sym’.

19. Argument: problemType, default='sm', type=str, Description: Problem Type: SM, SMI or SMT, Other options: ‘smi’, ‘smt’.

For example, following command will run program with grid size 5X5 and 12 agents on it.

python3 MARL.py --grid=5 --agents=12

Note: MARL.py uses multi-processing. It is recommended to use the number of CPUs same as the total agents.