Introduction to Artificial Intelligence
Project 4 – Reinforcement Learning

Jianmin Li & Minlie Huang

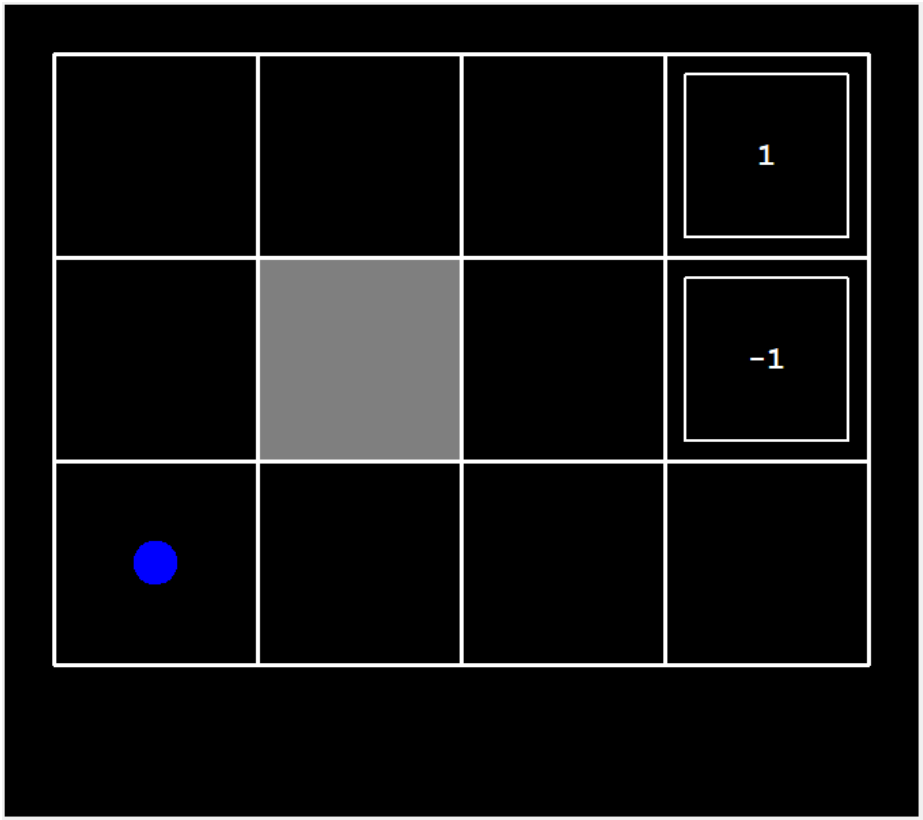
Spring 2018

Reinforcement Learning and Pacman

- Berkeley RL Pacman Project
  - http://ai.berkeley.edu/reinforcement.html
  - https://s3-us-west-2.amazonaws.com/cs188websitecontent/projects/release/reinforcement/v1/001/reinforcement.zip

Get Started

- python gridworld.py –m
- python gridworld.py -g MazeGrid



Basic Tasks

- Value Iteration
  - Implement ValueIterationAgent in valueIterationAgents.py
  - python gridworld.py -a value -i 100 -k 10
  - 7 points
- Parameter Analysis in Value Iteration
  - Change parameters so that the optimal policy causes the agent to attempt to cross the bridge
  - Put your answer in question2() of analysis.py
  - python gridworld.py -a value -i 100 -g BridgeGrid --discount 0.9 --noise 0.2
  - 1 point

Basic Tasks

- Q-Learning
  - implement the update, computeValueFromQValues, getQValue, computeActionFromQValues methods in QLearningAgent in qlearningAgents.py
  - python gridworld.py -a q -k 5 –m
  - 6 points
- Q-Learning and Pacman
  - Play pacman with Q-Learning
  - python pacman.py -p PacmanQAgent -x 2000 -n 2010 -l smallGrid
  - QLearningAgent.getAction is provided. Some code in QLearningAgent may be modified
  - 1 point

Bonus

- **Approximate Q-Learning**
  - Implement ApproximateQAgent in qlearningAgents.py
  - python pacman.py -p ApproximateQAgent -x 2000 -n 2010 -l smallGrid
  - 1 point
  - NOTE: Deep QL is inspired from this

Submission

- A 2-3 pages report (either Chinese or English)
  - Compare with different parameters in Value Iteration and Q-Learning, etc.
- Zip the files as the following structure
  - student_id.zip (e.g. 20090112xx.zip)
    - student_id.pdf
    - valueIterationAgents.py
    - qlearningAgents.py
    - analysis.py

Grading

- Due
  - 2018/7/4 23:59:59
  - For Senior/Graduate Students: 2018/6/25 23:59:59
- Correctness of algorithms (80%)
- Report (20%)