**Markov Decision Processes Analysis**

John Seon Keun Yi

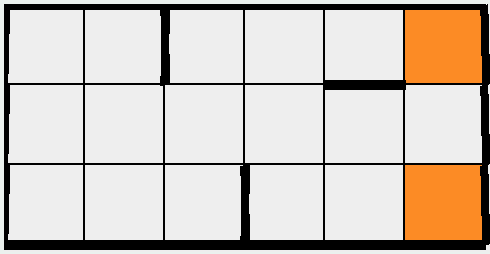
CS4641

**Introduction**

In this assignment we analyze three reinforcement learning algorithms: value iteration, policy iteration and Q-learning. The algorithms are performed in two different Markov decision processes. (MDPs) An MDP provides a mathematical framework for decision making, particularly when the outcome of a decision is partly stochastic. That is, when the outcome following a decision or action is sometimes random, and other times is as expected. In this analysis, the three algorithms are run on different MDPs and the results between the algorithms are compared through various metrics.

**GridWorld**

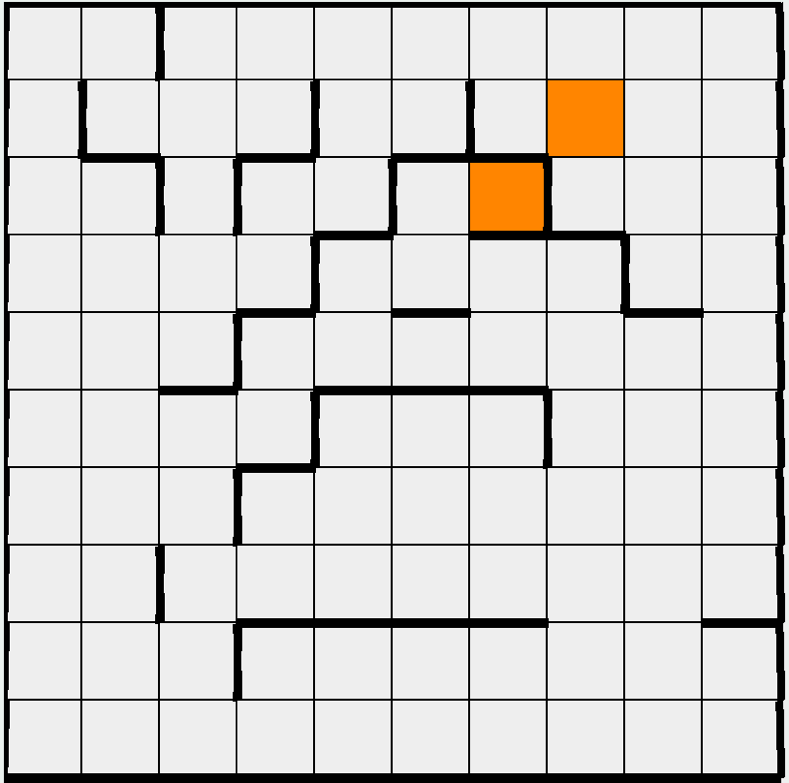
I used two “GridWorld” problems for the MDPs, similar to the examples we went over in class. GridWorld refers to a world in which an agent (robot) moves through a discrete state space represented as a block of grids. Each state is a cell in the grid, and there exists obstacles and one or more terminal (goal) states. I used RL\_sim to generate the two problems and test them on the three algorithms.

 Basic rules apply in this world. The agent can move in four directions: up, down, left, right. Thus, there are four actions the agent can take. Also, to add in the stochastic nature of MDPs, a variable is added to represent noise in the environment. Noise of an action, named “pjog” in RL\_sim, determines the probability that the agent will take some other action and end up in different states. For example, if pjog is 0.3, and the action taken is a=RIGHT, the agent will instead take action UP, DOWN and LEFT at a probability of pjog/(number of actions-1) = 0.1 each.

**Figure 1.** Map of the small maze.

When the agent makes a transition of one state to another, it gains a path cost of 1. However, when the agent hits a wall, it stays on the same state and receives a penalty of -50. The purpose of the penalty is to encourage the agent to avoid hitting walls and converge quickly. Finally, the goal state(s) is marked as an orange block in the maze.

The first MDP problem is a small maze, as shown in Figure 1. The small maze is a 3x6 maze with 18 states. The start state is the lowest leftmost block. The walls are represented as bold lines in the maze. There are two goals in this maze. The reason I put two goals is that I thought it would be interesting to see which one of the goals the algorithms prefer, and if the preference changes between algorithms.

**Value Iteration**

**Figure 2.** Map of the big maze.

The first algorithm applied to solve the two MDP problems is value iteration. Value iteration applies arbitrary utilities (values) to each state and updates the utilities by adding the immediate reward of the state with the expected discounted reward of the state if the agent takes optimal actions onward. For each state, value iteration simply chooses the action that maximizes the utility of the following state. The process of value iteration is as follows:

* Assign arbitrary utility values to each state
* Update the utility of each state based on the utilities of its neighbors
* Use the Bellman update equation to update the utility for each state
* Repeat steps 2 and 3 until convergence

Value iteration requires some randomness in the agent’s actions in order to find the optimal path. If there is no noise in an action, the value for the transition function will be always 1, and the optimal path will be stuck in a local maximum. In RL-sim, the variable PJOG takes care of the noise. As explained in the introduction,

**Q-learning**