# Multi Agent Reinforcement Learning

Anshuman Dangwal
Roll no: 21044
Assignment 1

August 28, 2024

## 1 Student MDP Problem

### 1.1 Introduction

The student Markov Decision Process (MDP) is defined with the following components shown in the table 1:

Table 1: State, Action, Transition Probability, and Reward

| **Current State** | **Action** | **Next State** | **Transition Probability** | **Reward** |
|---|---|---|---|---|
| Hostel | Attend Classes | Canteen | 0 | - |
| | Attend Classes | Hostel | 0.5 | -1 |
| | Attend Classes | Academic Building | 0.5 | -1 |
| | Hungry | Canteen | 1 | -1 |
| | Hungry | Hostel | 0 | - |
| | Hungry | Academic Building | 0 | - |
| Academic Building | Attend Classes | Canteen | 0.3 | 3 |
| | Attend Classes | Hostel | 0 | - |
| | Attend Classes | Academic Building | 0.7 | 3 |
| | Hungry | Canteen | 0.8 | 3 |
| | Hungry | Hostel | 0 | - |
| | Hungry | Academic Building | 0.2 | 3 |
| Canteen | Attend Classes | Canteen | 0.1 | 1 |
| | Attend Classes | Hostel | 0.3 | 1 |
| | Attend Classes | Academic Building | 0.6 | 1 |
| | Hungry | Canteen | 1 | 1 |
| | Hungry | Hostel | 0 | - |
| | Hungry | Academic Building | 0 | - |

The student MDP can also be graphically represented as shown in the figure 1:

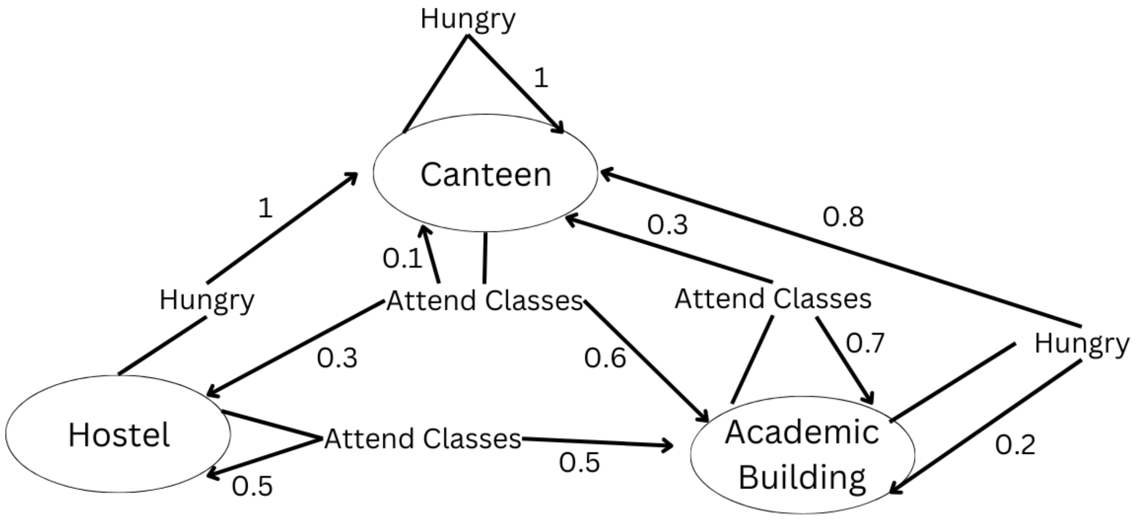

Figure 1: Student MDP graph

## 1.2 Representation of State, Reward, Actions, and Probability Transition Function in code

The Student Markov Decision Process (MDP) is defined as follows:

- **States (S)**: The set of all possible states in the MDP is represented as a list:

$$S = \{\text{'canteen', 'hostel', 'academic building'}\}$$

- **Actions (A)**: The set of all possible actions is also represented as a list:

$$A = \{\text{'attend classes', 'hungry'}\}$$

- **Reward Function (R)**: The reward function is represented as a dictionary where each state is mapped to a numerical reward:

$$R = \{\text{'canteen'} : 1, \text{'hostel'} : -1, \text{'academic building'} : 3\}$$

- **Probability Transition Function (P)**: The state transition probabilities are represented as a nested dictionary. For each state $s$, there is an associated dictionary that maps an action $a$ to another dictionary, which in turn maps the next state $s'$ to a probability:

$$P(s, a, s') = \text{probability of transitioning from state } s \text{ to } s' \text{ given action } a$$

The transition probabilities sum to 1 for each state-action pair.

## 1.3 Value Iteration and Policy Iteration Results

The results obtained from performing value iteration and policy iteration on the designed student Markov Decision Process (MDP) are as follows:

### 1.3.1 Value Iteration Results

The optimal policy and the optimal value function obtained from value iteration are as follows:

- **Optimal Policy**:

  {'canteen' : 'attend classes', 'hostel' : 'attend classes', 'academic building' : 'attend classes'}

- **Optimal Value Function**:

  {'canteen' : 17.96, 'hostel' : 15.19, 'academic building' : 20.98}

### 1.3.2 Policy Iteration Results

The optimal policy and the optimal value function obtained from policy iteration are as follows:

- **Optimal Policy**:

  {'canteen' : 'attend classes', 'hostel' : 'attend classes', 'academic building' : 'attend classes'}

- **Optimal Value Function**:

  {'canteen' : 17.96, 'hostel' : 15.19, 'academic building' : 20.98}

## 1.4 Discussion of Results

- Both policy iteration and value iteration produced the same optimal policy: always choosing the "attend classes" action in all states and converged to the same value function.
- The state 'academic building' was given the highest value by both the algorithms followed by 'canteen' and then 'hostel'.

# 2 Robot MDP

## 2.1 Problem Formulation

This MDP models a robot navigating a $9 \times 9$ grid environment. The states, actions, rewards, and state transition probabilities are defined as follows:

- **States (S)**: The set of all possible positions $(i, j)$ in a $9 \times 9$ grid, excluding the positions occupied by walls. The walls are located at:

$$\{(2,4),(3,4),(4,4),(4,3),(4,2),(6,9),(6,8),(6,7),(6,6),(7,6),(8,6),(9,6)\}$$

- **Actions (A)**: The robot can move in four directions: 'up', 'down', 'left', and 'right'.
- **Reward Function (R)**: The robot receives a reward of 1 upon reaching the goal state $(9, 9)$. All other states have a reward of 0.
- **State Transition Probability (P)**: The state transition probabilities are defined as follows:
  - Moving 'up' increases the $j$ coordinate by 1.
  - Moving 'down' decreases the $j$ coordinate by 1.
  - Moving 'left' decreases the $i$ coordinate by 1.
  - Moving 'right' increases the $i$ coordinate by 1.

  Transitions to states outside the grid or into walls have zero probability. Special rules apply for state $(3, 3)$, which is the IN state of the tunnel, hence taking 'right' from $(2, 3)$ and 'up' from $(3, 2)$ will lead directly to state $(7, 7)$ with probability 1 which is the OUT state of the tunnel.

Figure 2 shows the visualization of the possible actions in each state that will lead to the movement of the robot.

## 2.2 Results

Upon performing Value iteration and Policy iteration using the above formulation, the results are shown in Figure 2. Both Policy Iteration and Value Iteration achieved the same deterministic optimal policy and optimal value function.

**NOTE:** ChatGPT was only for visualization and plot of the Robot MDP. The rest all the code and formulation were completely done by myself and no help from any AI tools was used. Moreover, the code provided by ChatGPT for visualization was not blindly used and I did read and understand the code for the plot.

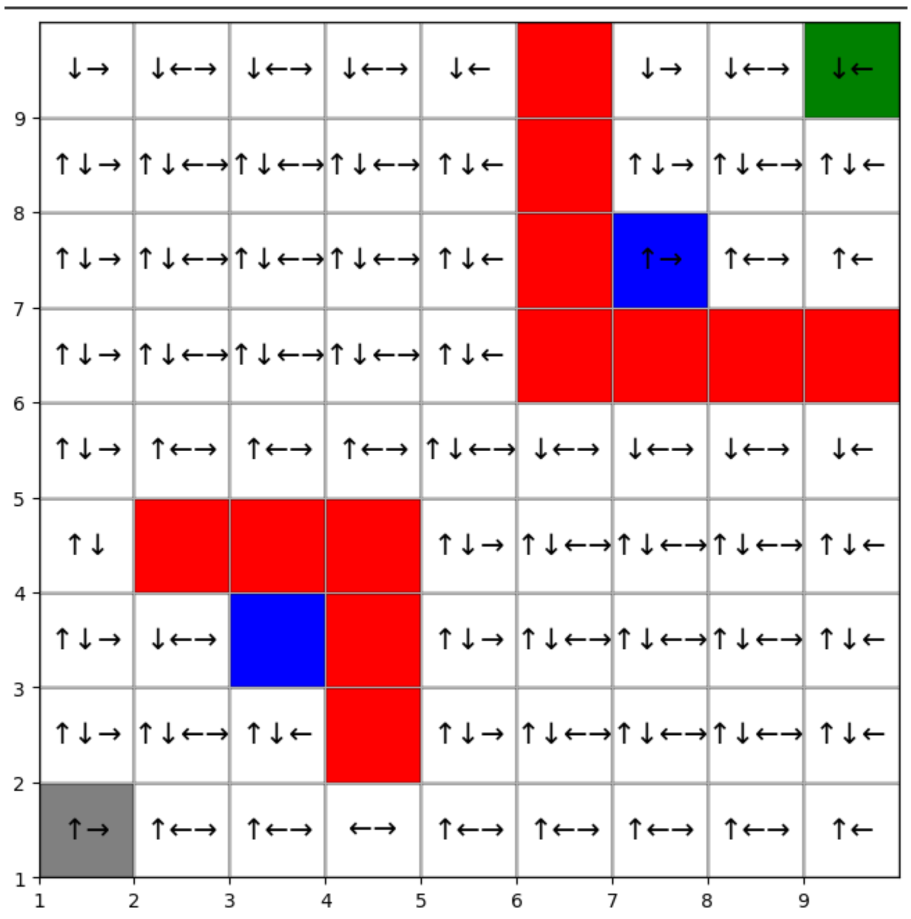

Figure 2: Robot MDP

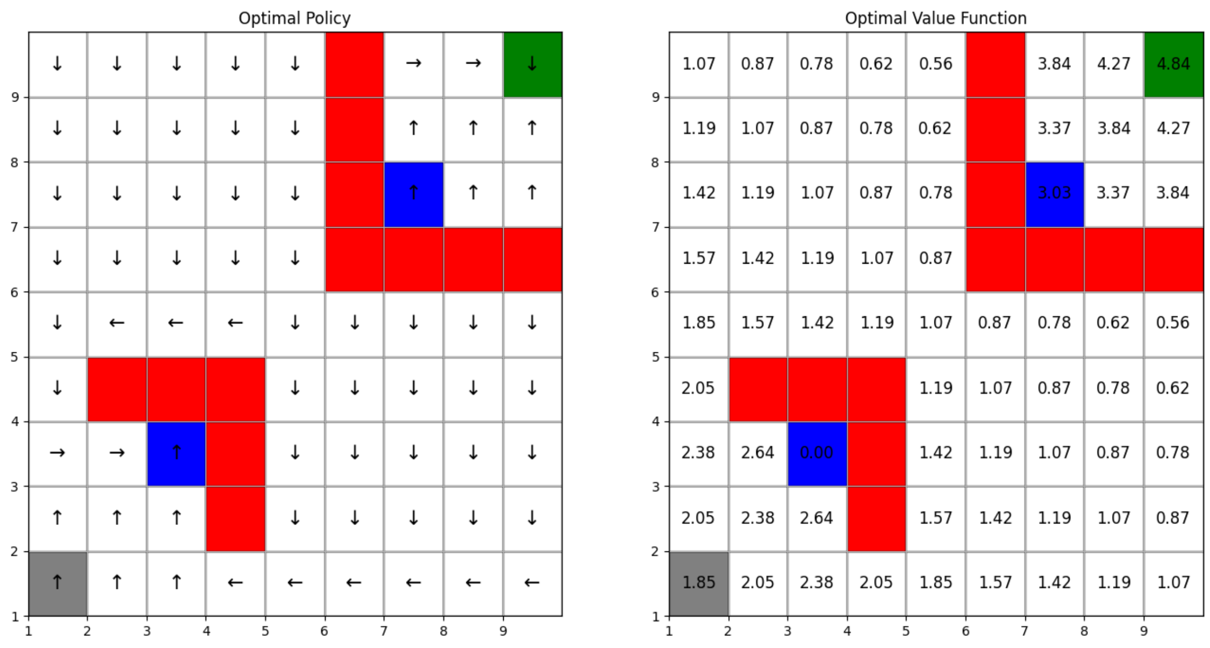

Figure 3: Robot MDP results