Project 3: Reinforcement learning and Inverse Reinforcement learning

Large Scale Social and Complex Networks: Design and Algorithms

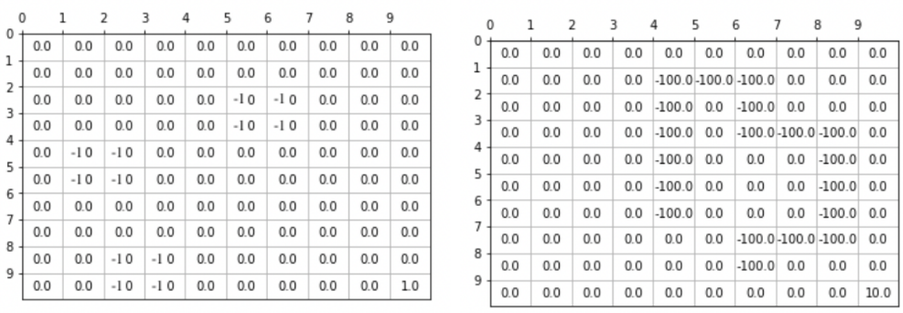
Ananya Deepak Deoghare, Kimaya Milind Kulkarni, Shruti Mohanty

005627628, 805528337, 705494615

ECE 232E Spring 2022

1 Reinforcement Learning

In this project, we will look at various Reinforcement Learning (RL) and Inverse Reinforcement Learning ideas. In particular, in the first phase of the study, we will investigate the optimum policy of an agent travelling in a two-dimensional environment. Because the project handout has a detailed description of the RL components, we will only briefly cover them in this report. Before delving into the RL algorithms, we'll utilize this question to create heat maps for the two Reward functions we'll be using in this project. Figure 1 shows how they are defined. Because the value of reward function 1 is difficult to differentiate, we shall consider them as 1 or +1. All the following results are derived based on this assumption, which might cause somewhat difference to the standard result.



	0	1	2	3	4	5	6	7	8	9
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0
4	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	-1.0	-1.0	0.0	0.0	0.0	0.0	0.0	1.0

(a) Reward function 1

	0	1	2	3	4	5	6	7	8	9
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	-100.0	-100.0	-100.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	-100.0	0.0	-100.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	-100.0	0.0	-100.0	-100.0	-100.0	0.0
4	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0	-100.0	0.0
5	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0	-100.0	0.0
6	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0	-100.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	-100.0	-100.0	-100.0	0.0
8	0.0	0.0	0.0	0.0	0.0	0.0	-100.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	10.0

(b) Reward function 2

Figure 1: Reward functions for the 2D square grid

Question 1:

In this topic, we are requested to create heatmaps for two reward functions for an agent traveling in a two-dimensional environment (grid-world). Figure 2 depicts the plots. We used the pcolor() function from the matplotlib.pyplot package to create the charts.

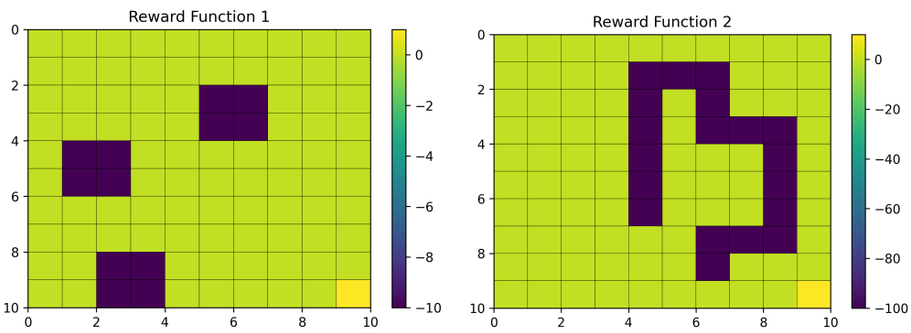


Figure 2: Heatmap visualization for ground-truth reward functions

Within a single glance, the produced heatmaps can highlight the states with the highest rewards as well as the places to avoid. In our plots, the darker regions correspond to states with the lowest reward, while the brighter portions correspond to ones with the highest reward. State 99 (bottom right corner) in particular delivers the greatest payout for both reward functions.

2 Optimal policy learning using RL algorithms

Question 2:

In this question, we are asked to set up the state-space, action set, transition probabilities, discount factor, and reward function for the agent, implement the initialization and estimation steps in the value iteration algorithm, and generate a plot of the optimal values of each state and snapshots of state values at linearly distributed steps until convergence.

Reinforcement learning (RL) is the process of discovering policies via interaction with the environment by identifying appropriate responses in certain settings to maximize a given reward and achieve a goal. It entails an agent that interacts with the environment by performing some action A_t after observing the environment O_t at time t. The reinforcement learning agent interacts with its surroundings and learns from the outcomes of its activities. At time t, the environment acknowledges the interaction and awards the reward R_t. The agent chooses future actions based on its own prior experiences. The history h_t at time t is formed by the sequence of acts, observations, and rewards up to time t. Assuming that the environment is completely visible, with $S_t = O_t$:

$$h_t = \{O_1, R_1, A_1, O_2, R_2, A_2, ..., A_{t-1}, O_t, R_t\}$$

The goal of the agent is to find A_t as a function of $h_t : f(h_t)$, thereby approximating $f(h_t)$ to maximize the cumulative reward for horizon T.

In this project, we use the Markov Decision Process to simulate the environment (MDP). The Markov process is a stochastic process in which the most recent values seen determine the future probability. A Markov state is an information state in which the future is independent of the past given the present, i.e. the current state contains the history in a latent space:

$$P(S_{t+1}|S_t, A_t, ..., S_1, A_1) = P(S_{t+1}|S_t, A_t)$$

The agent in MDP has control over the activities it can conduct at different states. The agent not only receives a reward from the environment, but also has the ability to transition to the next state as defined by the state transition matrix. An MDP's domain is defined as $\{S, A, P^a_{SS'}, R^a_{SS'}, \gamma\}$ where:

- S: State, Environment representation/configuration
  We assume that the environment state S^e_t is the same as the observation state O_t for most Markov processes (fully observable environment). The state space in this project is a 10x10 square grid with 100 states. The state space is bounded.
- A: Action, interaction with the environment.
  In this project, an agent can do one of four actions within the grid world: move up, move down, move left, or move right. The action space, like the state space, is discrete.
- $P^a_{SS'}$: State transition matrix
  Describes a Markov chain / transition probability from one state S to another state S' for action a. It can be thought of as a model of the system when coupled with R. It follows the Markov process. For this project, each action corresponds to a movement in the intended direction with probability $1 - w$, but has a probability of of moving in a random direction instead due to wind. Several other heuristics are as follows:

  - If 2 states S and S' are not neighbors then $P(s_{t+1} = s | s_t = s, a_t = a) = 0$
  - For inner grid states (non-boundary states), an agent moves in the desired direction (via desired action) with probability $1 - w + 0.25w$, while the probability of moving in the other three directions due to wind is $0.25w$. The agent cannot be in the same state at the next time step.
  - At corner states (states with only two neighboring states), an agent can be blown of the grid in two directions. For moving of the grid consciously, the probability of being of the grid is $1 - w + 0.25w + 0.25w$, while the probability of being blown of the grid unconsciously (when the agent actually decides to stay on the grid) is $0.25w + 0.25w$.
  - At edge states (states with three neighboring states), an agent can be blown of the grid in one direction. For moving of the grid consciously, the probability of being of the grid is $1 - w + 0.25w$, while the probability of being blown of the grid unconsciously (when the agent actually decides to stay on the grid) is $0.25w$.

- $R^a_{SS'}$: Reward
  Immediate scalar feedback from the environment indicating how the agent is performing when transitioning from state S to another state S' for action a. The ground truth rewards from the environment are provided by the two reward functions from Question 1. In other words, for this project, we assume that the reward function only depends on the state the agent transitions to (S').

- $\gamma \in [0, 1)$: Discount factor
  Is used to compute the present value of future reward. If γ is close to 0, future rewards are discounted more and vice versa.

To find the optimal policy π^*, we can use the value iteration algorithm. Policy is defined as a set of rules/probability distribution that maps states to actions (describes agent's behavior). The state-value function V is the total discounted reward an agent is expected to receive in its lifetime starting from state S and following policy π. V is used for policy evaluation and value iteration (find optimal value for fixed policy) through dynamic programming. Optimal state-value function for all policies, V^* is defined as:

$$V^*(s) = max_\pi V^\pi(s)$$

An optimal policy π^* is one whose return/reward is greater than all other policies for all states.

$$\pi^* \geq \pi' \iff V^{\pi^*}(s) \geq V^{\pi'}(s), \ \ \forall s \in S$$

In other words, optimal policies achieve V^*, given by the recursive Bellman optimality equation:

$$V^*(s) = \max_{a \in A} \mathbb{E}[r_t + \gamma V^*(s_{t+1}) | s_t = s, a_t = a] = \max_{a \in A} P^a_{s',s}[R^a_{s',s} + \gamma V^*(s')]$$

To solve for V^* iteratively, the value iteration algorithm is used. It is composed of three steps:

- Initialization: Set the value for all states to 0 and initialize variable Δ to ∞. Select a value ϵ, which dictates when to stop the value iteration algorithm (convergence criterion).
- Estimation: While $\Delta > \epsilon$, for all states s in state space S:
  - Set Δ to 0.
  - Copy old value of V(s) to variable v.
  - Compute V(s) using Bellman Equation.
  - Find the maximum of Δ and $|v - V(s)|$ and set the value of Δ to the maximum value.
- Computation: Find the actions corresponding to the optimal policy for each state.

$$\pi^*(s) \leftarrow \arg\max_{a \in A} P^a_{s',s}[R^a_{s',s} + \gamma V^*(s')]$$

For this question, we set w to 0.1, γ to 0.8,ϵ to 0.01 and use reward function 1. The number of states is 100 and the number of actions is 4. Figure 3 shows the optimal values for all the states found using value iteration.

Figure 3: Optimal values of the states for reward function 1

We observed that the algorithm converges in 22 steps (N = 22).
Figure 4 shows the snapshots of the state values for 5 different steps linearly distributed from 1 to N. We show the snapshots for step 1, step 6, step 11, step 16 and step 21.

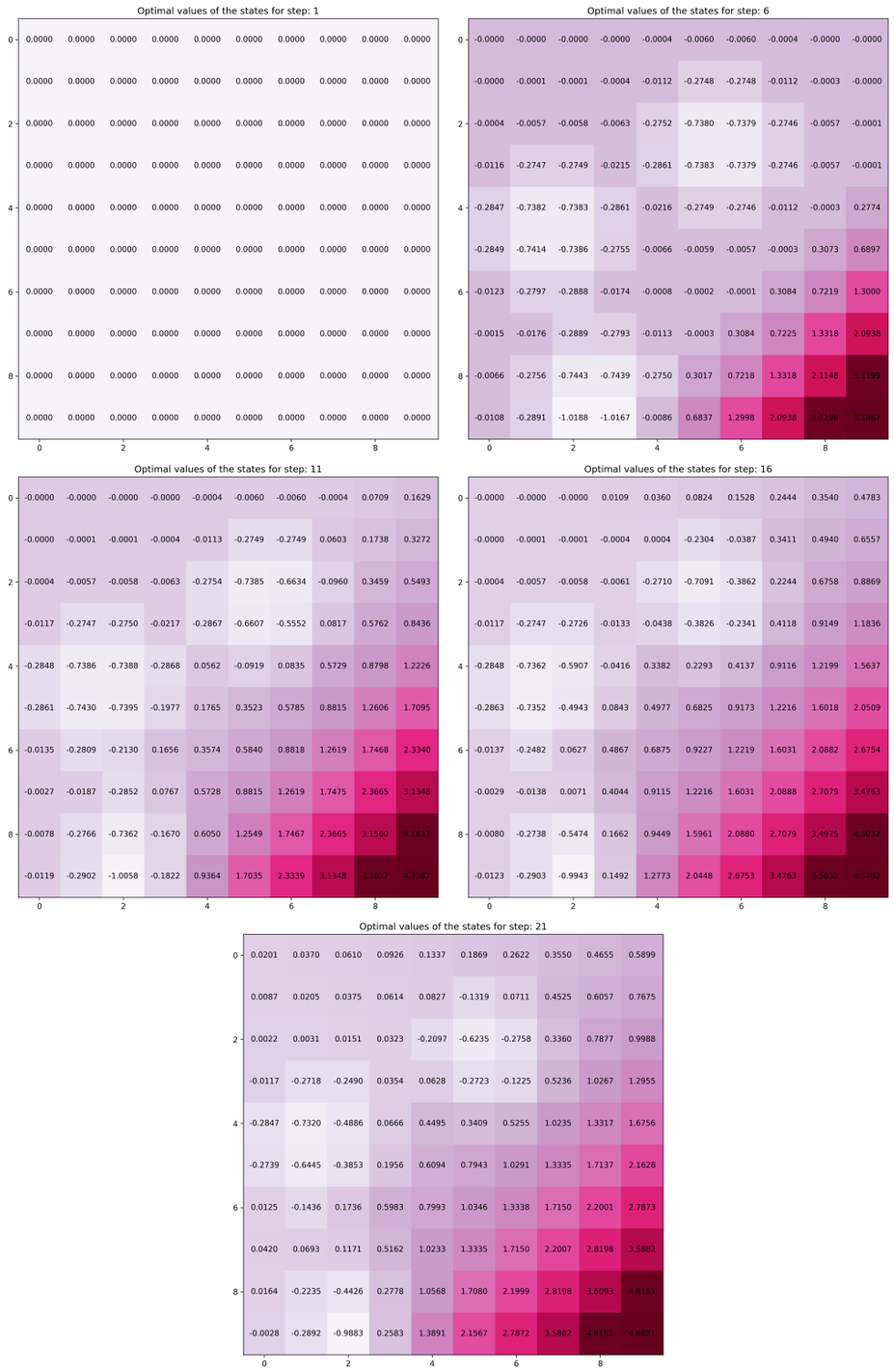


Figure 4: Snapshots of state values in value iteration for steps 1, 6, 11, 16 and 21

Figures 3 and 4 show that state 99 (the state in the lower right corner) has the greatest optimum value, which is not surprising given that state 99 offers the maximum reward for reward function 1. As one gets away from state 99, the ideal value of each state gradually declines. Furthermore, the blocks of states that produce negative rewards have negative optimum values. Neighboring states near these blocks have a lower ideal value than neighboring states near the state with the highest reward. In other words, as an agent goes towards states with low reward, the optimum values of the nearby states decline gradually, and as an agent advances towards desirable situations with more reward, the ideal values of the states it encounters increase gradually.

In addition, from the snapshots in Figure 4, most of the states have a value of 0 during the earlier steps, yielding a sparse $V(s)$ matrix. However, as the number of steps increases, the state values near state 99 (the state with the largest reward) gradually grow, while the state values near the states with negative rewards gradually drop. Because state 99 receives the biggest reward, the values of state 99 and its nearby states rise more quicker. As the algorithm approaches convergence, the sparsity in $V(s)$ fades, with most of the states awarded a non-zero optimum value, with states near state 99 assigned very high and positive values, and states near blocks with negative reward allocated negative values. We can also see that the rate of change of state values is faster in the early steps, then follows a logarithmic rate of convergence until $\Delta < \epsilon$.

Question 3:

Figure 5 depicts the heatmap of optimal state values derived in Question 2 throughout the 2D grid for reward function 1.

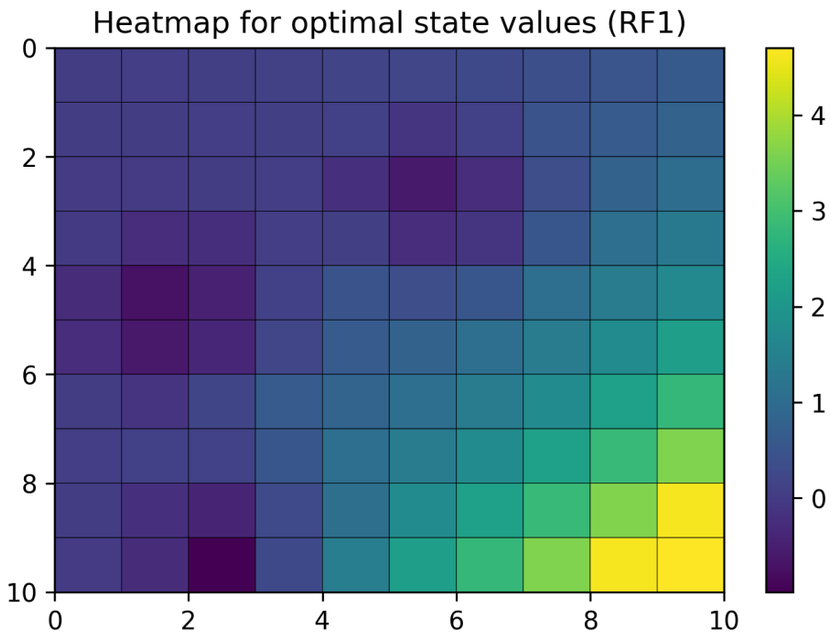


Figure 5: Heatmap visualization for optimal state values (reward function 1)

Question 4:

Considering the notion of the expected discounted cumulative benefit for the agent in a particular state in the heat map above, we can see that states in the bottom right have the greatest values, because the maximum reward is in that region. When the number of steps to each stage grows, the value lowers. For example, states at the highest reward have the highest values among all states since it just takes one step to get there, but states in the upper left have the lowest values because it takes so many steps to get there. Another thing to notice is that three regions outside the top left corner have the lowest value, which is because they have the lowest rewards according to the Reward Function 1, and the neighbors of these three regions will also have low values, but notice that some of them are close to the highest reward, so they will have slightly higher values than the neighbors who are further from the highest reward, which makes sense.

We can make several inferences from the heatmap in Figure 5. Note that the darker regions correspond to states with low optimal values, while the brighter regions correspond to states with high optimal values. State 99 has the highest reward and highest optimal value.

- The ideal value of each state gradually declines as one gets away from state 99, which provides the largest reward. Furthermore, surrounding states near the blocks giving negative rewards have a lower ideal value than neighboring states near the state yielding the maximum reward. In other words, as an agent goes towards states with low reward, the optimum values of the nearby states decline gradually, and as an agent advances towards desirable situations with more reward, the ideal values of the states it encounters increase gradually. The greater the distance between a state and the state with the highest reward, the lower its value. Mathematically:

$$V^*(s) = \max_{a \in A} \mathbb{E}[r_t + \gamma V^*(s_{t+1}) | s_t = s, a_t = a] = \max_{a \in A} P^a_{s',s}[R^a_{s',s} + \gamma V^*(s')]$$

  From the above equation, we see that $V(s) \propto R^a_{S',S}$.

- The ideal values surrounding the condition with the highest reward show an obvious, progressive, and steady deterioration. This is due to the Bellman equation's discount component, which reduces future benefits.

- The patterns apparent in Figure 2's heatmap of reward function 1 are generally observable in the heatmap of optimum values. Figure 2 shows three blocks of negative rewards for reward function 1. The heatmap of optimum values inside the same predicted areas in the state space pretty clearly shows the same three blocks. Furthermore, the zone near the state with the greatest award shines brighter. This gives us the impression that we can extract the reward function by monitoring how the environment or an expert behaves. This is basically what inverse reinforcement learning does (IRL).

Question 5:

Figure 6 shows the optimal actions taken by the agent in the 2D grid for reward function 1, shown using arrows.

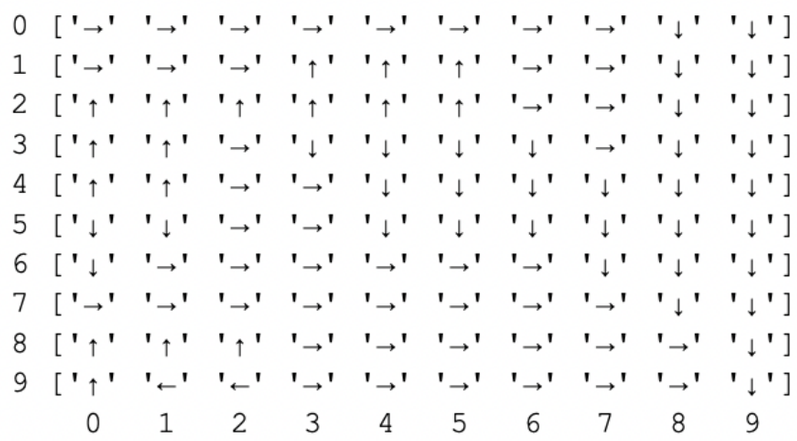


Figure 6: Optimal actions undertaken by the agent for reward function 1, shown using arrows

We can make several observations from Figure 6:

- All of the arrows eventually lead to state 99, which provides the largest prize of any state. Furthermore, when confirmed by the heatmap in Figure 4, the arrow in a given state tends to go in the direction of the state that would eventually maximize the expected reward. Furthermore, the agent prefers to migrate away from places with negative rewards (darker parts on the heat map), and the arrows in Figure5 lead to brighter locations on the heat map. This corresponds to our understanding since the agent's purpose is to maximize the total cumulative benefit it can earn within a finite horizon, and hence should strive for state 99 because the rewards from other states are either negative or zero.

- By monitoring the optimal values of its nearby states, the agent may compute the ideal action to execute at each state. This becomes clear when we examine the direction the arrow is heading given the optimum values of nearby states in Figure 3. At each stage, the arrow always points to the nearby state with the highest optimum value among all neighbors. As a consequence, even if the agent is uninformed of future ideal values, the agent may still construct an optimal policy by monitoring the local best values inside each state's neighborhood. The best policy for each state is found mathematically via the value iteration process as follows:

$$\pi^*(s) \rightarrow \underset{a \in A}{argmax} P^a_{s',s}[R^a_{s',s}, \gamma V^*(s')]$$

  Apart from the discount factor (which is xed in our example), transition matrix, and rewards, the following equation shows that optimum policy is determined by the optimal values of the nearby states. As a result of seeing local optimal values, value iteration causes the agent to take an action that maximizes the expected payoff.

Question 6:

Figure 7 shows the optimal values for all the states found using value iteration for reward function 2.

Figure 7: Optimal values of the states for reward function 2

Figure 7 shows that position 99 (in the lower right corner) has the largest ideal value, which is obvious because state 99 produces the biggest reward for reward function 2. As one gets away from state 99, the ideal value of each state gradually declines. Furthermore, the category of states with negative rewards has negative optimum values. Neighboring states near these blocks have a lower ideal value than neighboring states near the state with the highest reward. In other words, as an agent goes towards states with low reward, the optimum values of the nearby states decline gradually, and as an agent advances towards desirable situations with more reward, the ideal values of the states it encounters increase gradually.

Question 7:

Figure 8 shows the heatmap of optimal state values across the 2D grid for reward function 2 obtained in Question 6.

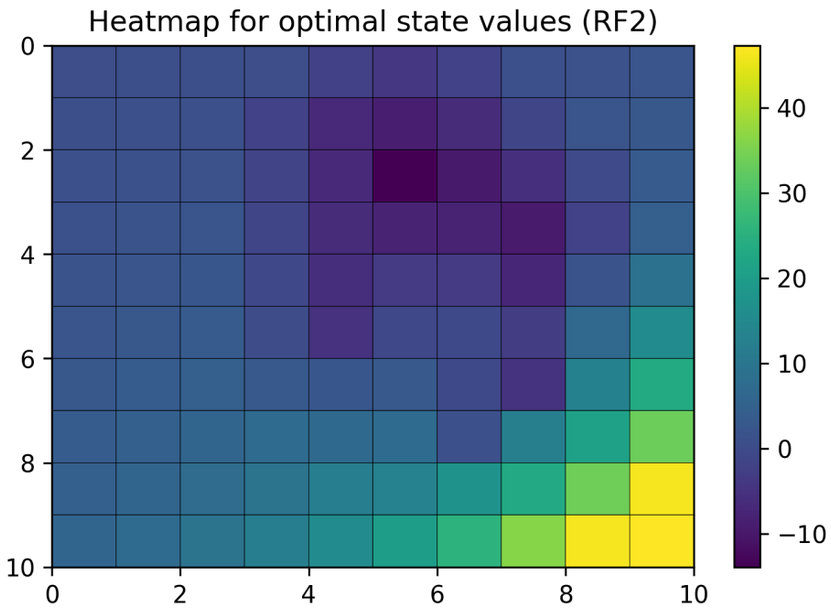


Figure 8: Heatmap visualization for optimal state values (reward function 2)

We can make several inferences from the heatmap in Figure 7. Note that the darker regions correspond to states with low optimal values, while the brighter regions correspond to states with high optimal values. State 99 again has the highest reward and highest optimal value.

- The ideal value of each state gradually declines as one gets away from state 99, which provides the largest reward. Furthermore, surrounding states near those states giving negative rewards have a lower ideal value than bordering states near the state yielding the highest benefit. In other words, as an agent goes towards states with low reward, the optimum values of the nearby states decline gradually, and as an agent advances towards desirable situations with more reward, the ideal values of the states it encounters increase gradually. The greater the distance between a state and the state with the highest reward, the lower its value. Mathematically:

$$V^*(s) = \max_{a \in A} \mathbb{E}[r_t + \gamma V^*(s_{t+1}) | s_t = s, a_t = a] = \max_{a \in A} P^a_{s',s}[R^a_{s',s} + \gamma V^*(s')]$$

  From the above equation, we see that $V(s) \propto R^a_{S',S}$.

- The patterns apparent in Figure 2's heatmap of reward function 2 are roughly observable in Figure 8's heatmap of optimum values. According to Figure 2, the negative incentives create a snake-like chain in the state space. The heatmap shows this chain roughly, with the darkest spots corresponding to the states with the lowest payouts. Furthermore, the zone near the state with the greatest award shines brighter. This gives us the impression that we can extract the reward function by monitoring how the environment or an expert behaves. This is approximately what happens in real life.

- State 52 has the lowest optimum value and is reflected in the heatmap as the darkest. This is due to the fact that state 52 is surrounded on three sides by states with negative rewards, with just one path that does not punish the agent. Thus, value iteration pushes the agent to avoid this state since the likelihood of landing in a state with negative reward is the largest of any state if the agent falls on it.

- Even if some of the negative-reward states are closer to state 99 than others, the distant states have greater optimum values than the states along the negative-reward chain. This suggests that being adjacent to a state with a high reward does not guarantee that a state will have a high ideal value if the majority of its neighbors have negative rewards. This is especially noticeable for states in the top left corner, which have a greater ideal value than ones in the negative-reward chain.

- The ideal values surrounding the condition with the highest reward show an obvious, progressive, and steady deterioration. This is due to the Bellman equation's discount component, which reduces future benefits.

Question 8:

Figure 9 shows the optimal actions taken by the agent in the 2D grid for reward function 2, shown using arrows.

```
0 ['↓' '↓' '↓' '←' '←' '→' '→' '→' '→' '↓']
1 ['↓' '↓' '↓' '←' '←' '↑' '→' '→' '→' '↓']
2 ['↓' '↓' '↓' '←' '←' '↓' '→' '→' '→' '↓']
3 ['↓' '↓' '↓' '←' '←' '↓' '↓' '↑' '→' '↓']
4 ['↓' '↓' '↓' '←' '←' '↓' '↓' '↓' '→' '↓']
5 ['↓' '↓' '↓' '←' '←' '↓' '↓' '←' '→' '↓']
6 ['↓' '↓' '↓' '↓' '↓' '↓' '←' '←' '→' '↓']
7 ['↓' '↓' '↓' '↓' '↓' '↓' '←' '↓' '↓' '↓']
8 ['→' '→' '→' '↓' '↓' '↓' '↓' '↓' '↓' '↓']
9 ['→' '→' '→' '→' '→' '→' '→' '→' '→' '↓']
    0   1   2   3   4   5   6   7   8   9
```

Figure 9: Optimal actions undertaken by the agent for reward function 2, shown using arrows

We can make several observations from Figure 9:

- All of the arrows eventually lead to state 99, which provides the largest prize of any state. Furthermore, when confirmed by the heatmap in Figure 8, the arrow in a given state tends to go in the direction of the state that would eventually maximize the expected reward. Furthermore, the agent prefers to migrate away from places with negative rewards (darker parts on the heatmap), and the arrows in Figure 8 indicate to brighter locations on the heatmap. This corresponds to our understanding since the agent's purpose is to maximize the total cumulative benefit it can earn within a finite horizon, and hence should strive for state 99 because the rewards from other states are either negative or zero.

- We can also see how the agent tries to migrate away from the snake-like negative reward chain and toward places that are geodesically further away from state 99. This is because the agent's purpose is to maximize the predicted cumulative reward inside the nite horizon, not to reach the state with the highest reward in the quickest path. The agent attempts to avoid incurring negative reward throughout its journey, even if it means choosing a longer but more optimum (in terms of total reward) path.

- By monitoring the optimal values of its nearby states, the agent may compute the ideal action to execute at each state. This becomes clear when we examine the direction the arrow is heading given the optimum values of nearby states in Figure 6. At each stage, the arrow always points to the nearby state with the highest optimum value among all neighbors. As a consequence, even if the agent is uninformed of future ideal values, the agent may still construct an optimal policy by monitoring the local best values inside each state's neighborhood. The best policy for each state is found mathematically via the value iteration process as follows:

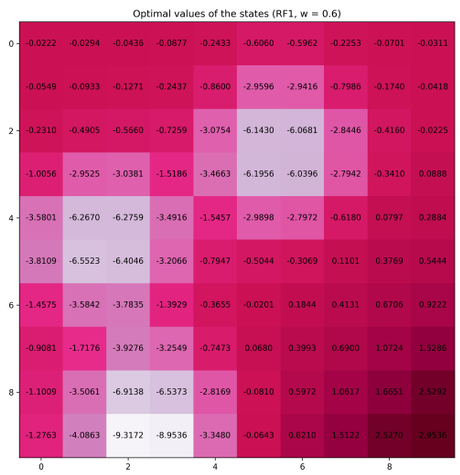
$$\pi^*(s) \rightarrow \underset{a \in A}{argmax} P^a_{s',s}[R^a_{s',s}, \gamma V^*(s')]$$

  Apart from the discount factor (which is fixed in our example), transition matrix, and rewards, the following equation shows that optimum policy is determined by the

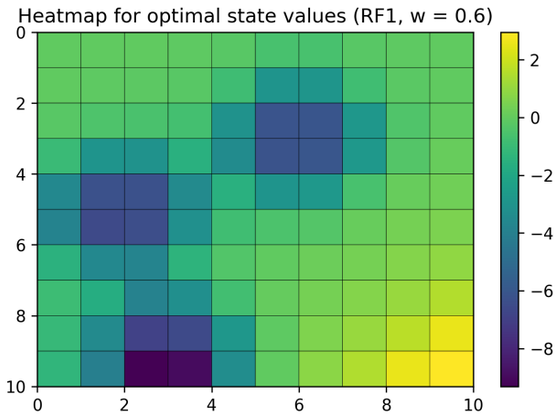
optimal values of the nearby states. As a result of seeing local optimal values, value iteration causes the agent to take an action that maximizes the expected payoff.

Question 9:

In this question, we are requested to alter from 0.1 to 0.6 and depict the effects on the learnt policy. Figure 10 depicts the grid of optimum values, the heatmap plot of optimal values, and the plot of optimal actions (represented by arrows) for reward function 1, while Figure 11 depicts the same for reward function 2.



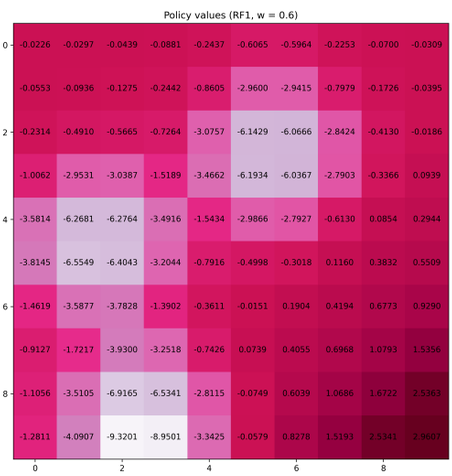
(a) Optimal values of the states for reward function 1 for $w = 0.6$



(b) Heatmap visualization for optimal state values (reward function 1) for $w = 0.6$

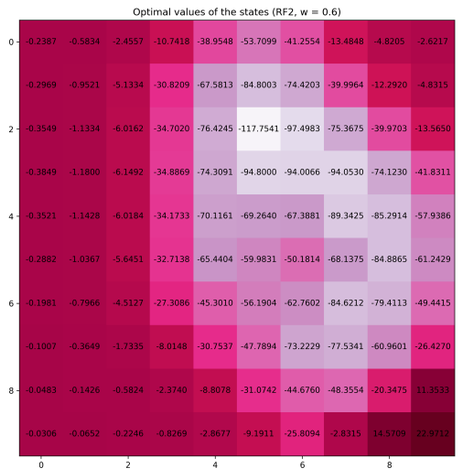
	0	1	2	3	4	5	6	7	8	9
0	↑	←	←	←	←	←	→	→	→	↑
1	↑	↑	↑	↑	↑	↑	↑	→	→	↓
2	↑	↑	↑	↑	←	↑	→	→	→	↓
3	↑	↑	↑	↑	←	↓	→	→	→	↓
4	↑	↑	↑	↑	↓	↓	↓	↓	↓	↓
5	↓	↓	→	→	↓	↓	↓	↓	↓	↓
6	↓	←	→	→	→	→	→	↓	↓	↓
7	←	←	←	→	→	→	→	→	↓	↓
8	↑	←	←	→	→	→	→	→	→	↓
9	↑	←	←	→	→	→	→	→	→	↓

(c) Optimal actions undertaken by the agent for reward function 1, shown using arrows for $w = 0.6$



(d) Policy values undertaken by the agent for reward function 1 for $w = 0.6$

Figure 10: Plots for Reward function 1 for $w = 0.6$

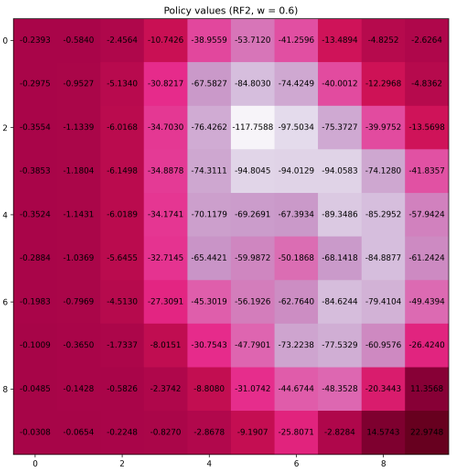


(a) Optimal values of the states for reward function 2 for $w = 0.6$

(b) Heatmap visualization for optimal state values (reward function 2) for $w = 0.6$

0 ['↑' '←' '←' '←' '←' '←' '→' '→' '→' '↑']
1 ['↑' '←' '←' '←' '←' '↑' '→' '→' '↑' '↑']
2 ['↑' '←' '←' '←' '←' '↓' '→' '→' '↑' '↑']
3 ['↓' '←' '←' '←' '←' '↓' '↓' '↑' '↑' '↑']
4 ['↓' '←' '←' '←' '←' '↓' '↓' '←' '→' '↑']
5 ['↓' '←' '←' '←' '←' '→' '←' '←' '→' '↓']
6 ['↓' '←' '←' '←' '←' '↓' '↑' '←' '→' '↓']
7 ['↓' '←' '←' '←' '←' '←' '←' '↓' '↓' '↓']
8 ['↓' '←' '←' '←' '←' '←' '↓' '↓' '↓' '↓']
9 ['↓' '←' '←' '←' '←' '←' '→' '→' '→' '↓']
0 1 2 3 4 5 6 7 8 9

(c) Optimal actions undertaken by the agent for reward function 2, shown using arrows for $w = 0.6$



(d) Policy values undertaken by the agent for reward function 2 for $w = 0.6$

Figure 11: Plots for Reward function 2 for $w = 0.6$

We can make several observations from all the plots in Figure 10 and 11:

- The optimal values for the states adjoining state 99 (state with the highest reward) have declined, as has the value for state 99 itself. The negative value of states with negative incentives has been heightened. In summary, instead of only surrounding states with positive and negative rewards, more states take on higher or lower values, providing the agent more options for where to migrate and, eventually, lowering the

likelihood of landing in the state with the highest reward.

- The heatmap and value plots for reward function 2 show that the road to the state with the highest optimum value has been cut-o owing to the rise in adversarial influence by the negative-reward chain over a longer state-space. This is because negative rewards are amplified while good rewards are attenuated.

- The policy plots show that increasing the inuence of negative incentives causes the agent to frequently leave the grid. Furthermore, there are fewer pathways for both reward functions that might lead to the agent reaching state 99. This occurred because the states next to state 99 do not give any positive or negative reward to the agent (0 reward), causing the values for those states to be overridden by the states with negative rewards.

- In the policy plots, we can see several local optima where the agent simply oscillates between two states. This is known as a stalemate circumstance.

- The exploration vs exploitation conundrum can be blamed for all of the aforementioned causes. We can conceive of it as the likelihood of exploration. Exploitation is making the best option feasible (by maximizing future benefit), whereas exploration is performing an immediately poor action to gain knowledge. While the poor action will always result in a lower reward in the short run, it may help the agent to acquire new tactics that will allow policy improvement in the long term. A poor sample complexity would result if there is no balance between exploration and exploitation. In this scenario, since is quite high, the balance has not been struck, causing the agent to explore more suboptimal behaviors at each stage rather than fairly leveraging knowledge of optimal values. As a result, the extracted policy is suboptimal.

The appropriate value of w is 0.1, since the agent is more likely to reach the state with the highest reward while having the choice of doing limited exploration and is less likely to become caught in local optima or be blown off the grid. Furthermore, when w is 0.1, the radius of hostile influence of negative reward chains/blocks is decreased, and more pathways point towards condition 99.