Assignment 2

Siddharth Nayak EE16B073

22nd September 2018

1 Taxi Driver Dilemma

The given problem can be modelled as a DP problem with 'N' stages.
States $\rightarrow \{A, B, C\}$

$$\text{Actions} \rightarrow \begin{cases} 0 : \text{Cruise the streets looking for a passenger.} \\ 1 : \text{Go to the nearest taxi stand and wait in line.} \\ 2 : \text{Wait for a call from the dispatcher (this is not possible in town B because of poor reception).} \end{cases}$$

The transition probabilities and the single stage rewards are given in the form of a matrix.
We have to find the optimal actions for number of stages N=10 and N=20.

1.1 Case 1: N=10

Setting the value of N=10 in the program we get the following results:

$$\begin{bmatrix} A & B & C \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

In the above matrix each column represents the optimal action to be taken in that state. So the above matrix shows us that it is optimal to take the action '0': where the driver goes and waits at the nearest taxi stand and wait in line at each stage except the last stage where it is optimal to take action '1': which is cruise the streets and look for a passenger. Also in the second last stage we have action '0' if in state 'A'.

$$\begin{bmatrix} A & B & C \\ 123.01 & 136.85 & 124.19 \\ 109.67 & 123.50 & 110.84 \\ 96.32 & 110.16 & 97.50 \\ 82.98 & 96.81 & 84.16 \\ 69.63 & 83.47 & 70.81 \\ 56.29 & 70.12 & 57.47 \\ 42.96 & 56.77 & 44.13 \\ 29.66 & 43.42 & 30.90 \\ 17.75 & 29.93 & 17.87 \\ 8 & 16 & 7 \\ 0 & 0 & 0 \end{bmatrix}$$

In the above matrix the expected costs are printed for each stage and each state.

1.2 Case 2: N=20

Setting the value of N=20 in the program we get the following results:

$$\begin{bmatrix} A & B & C \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

In the above matrix each column represents the optimal action to be taken in that state. So the above matrix shows us that it is optimal to take the action '0': where the driver goes and waits at the nearest taxi stand and wait in line at each stage except the last stage where it is optimal to take action '1': which is cruise the streets and look for a passenger. Also in the second last stage we have action '0' if in state 'A'.

$$\begin{bmatrix} A & B & C \\ 256.46 & 270.29 & 257.63 \\ 243.11 & 256.95 & 244.29 \\ 229.77 & 243.60 & 230.95 \\ 216.42 & 230.26 & 217.60 \\ 203.08 & 216.91 & 204.26 \\ 189.73 & 203.57 & 190.91 \\ 176.39 & 190.22 & 177.57 \\ 163.05 & 176.88 & 164.22 \\ 149.70 & 163.53 & 150.88 \\ 136.36 & 150.19 & 137.53 \\ 123.01 & 136.85 & 124.19 \\ 109.67 & 123.50 & 110.84 \\ 96.32 & 110.16 & 97.50 \\ 82.98 & 96.81 & 84.16 \\ 69.63 & 83.47 & 70.81 \\ 56.29 & 70.12 & 57.47 \\ 42.96 & 56.77 & 44.13 \\ 29.66 & 43.42 & 30.90 \\ 17.75 & 29.93 & 17.87 \\ 8 & 16 & 7 \\ 0 & 0 & 0 \end{bmatrix}$$

In the above matrix the expected costs are printed for each stage and each state.

If the driver chooses to take action '1':where the driver goes and waits at the nearest taxi stand and wait in line at each stage then it is not optimal as in the last stage, in the optimal actions we have to take action '0'.Also in the second last stage in state 'A' we have to take action '0'. Therefore it is not optimal to just take action'1' always in all stages.Also it makes sense to search for a passenger instead of waiting for a passenger at the last stage.

2 Gridworld

In the given problem we have a gridworld with a 10x10 state-space.There are two goals: Goal 1 and Goal 2. In one case the Terminal state is Goal 1 and in another case we have the terminal state as Goal 2.
This problem can be modelled as a Stochastic Shortest Path Problem(SSP) with the terminal state being the goal state.
Actions $\rightarrow$ {up,right,down,left}
As the transitions are stochastic in nature we have to apply the 'T' operator to get the optimal actions(policy) and the optimal cost for each of the states and available actions.

2.1 When to stop?

So the value iteration loop can go on till infinity. So we have to decide when to stop. This decision can be made with the help of the graph of $\delta_i = max(|J_{i+1} - J_i|)$ vs the number of iterations.
So we can decide a particular threshold for $\delta_i (= 0.01)$ so that if it's value reduces below that then we can stop the iterations as it would have converged.

2.2 Plot of $\delta_i = max(|J_{i+1} - J_i|)$ vs the number of iterations

The graph of δ_i vs the number of iterations has an exponential decrease which is consistent with the fact proved in the class that the value iteration(Repeated application of the 'T' operator) has an geometric decrease.

2.2.1 Goal 1:

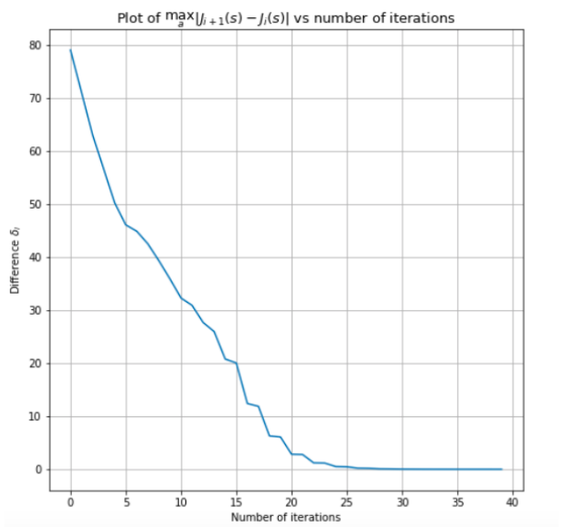

Figure 1: $\max(|J_{i+1} - J_i|)$ vs the number of iterations.

The value of δ_i goes below the value of 0.01 after 33 iterations. Thus it is ideal to stop the loop after 33 iterations as we have a converged value.

2.2.2 Goal 2:

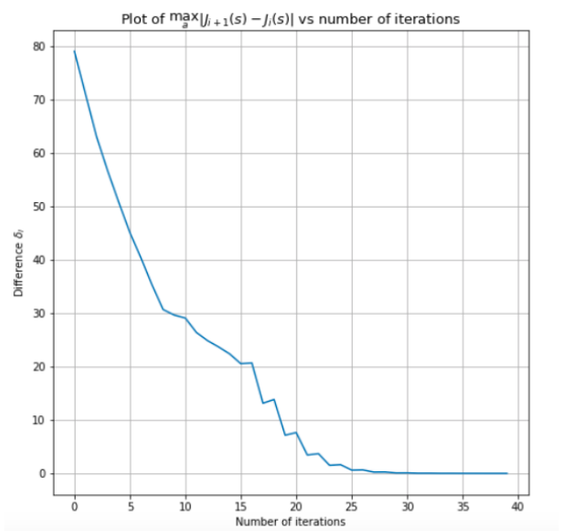

Figure 2: $\max(|J_{i+1} - J_i|)$ vs the number of iterations.

The value of δ_i goes below the value of 0.01 after 37 iterations. Thus it is ideal to stop the loop after 37 iterations as we have a converged value.

3 Goal 1:

The rewards can be explained as follows:

- The reward at the 'gray IN' cell is quite low as it takes us further away from 'GOAL 1'

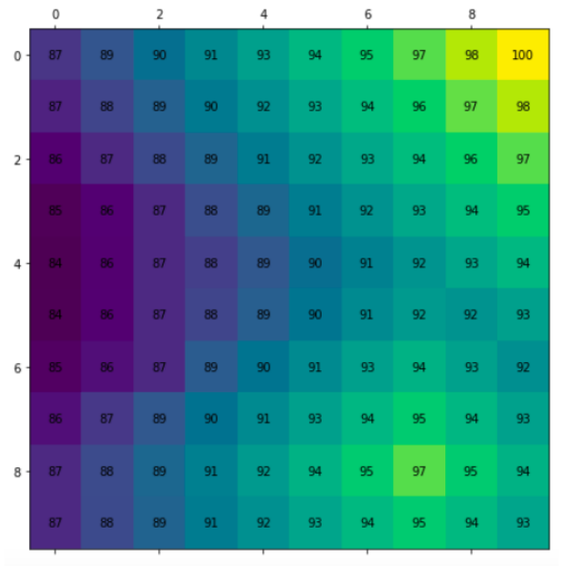

Figure 3: $J(s)$ after the termination of the loop

```
[['→' '→' '→' '→' '→' '→' '→' '→' '→' '↑']
 ['→' '→' '→' '→' '→' '→' '→' '→' '↑' '↑']
 ['→' '→' '→' '→' '→' '→' '→' '↑' '↑' '↑']
 ['→' '→' '→' '→' '→' '→' '→' '↑' '↑' '↑']
 ['→' '→' '→' '→' '→' '→' '→' '→' '↑' '↑']
 ['→' '→' '→' '→' '→' '→' '↓' '↓' '↑' '↑']
 ['↓' '↓' '→' '→' '→' '→' '↓' '↓' '↓' '←']
 ['→' '→' '→' '→' '→' '→' '↓' '↓' '↓' '←']
 ['→' '→' '→' '→' '→' '→' '→' '↓' '←' '←']
 ['→' '→' '→' '→' '→' '→' '→' '↑' '←' '←']]
```

Figure 4: $\Pi(s)$ after the termination of the loop

- The rewards decreases as we go further away from the 'GOAL 1' (taking into considerations the effects of the wormhole)
- The rewards at the 'orange IN' cell are higher than it's neighbours because these cells take us closer to the 'GOAL 1'

Note: The actions in the 'Goal' cell and the 'IN' cell are not valid as one cannot take action in those cells. The actions can be explained as follows:

- The actions around the 'orange IN' cell point into the cell as going into that cell takes to the 'orange OUT' which is nearer to 'GOAL 1'
- The actions around the 'GOAL 2' point into the goal which is quite obvious.
- Starting in any state we will always land up in 'GOAL 1' which means that there are no improper policies.

4 Goal 2:

The rewards can be explained as follows:

- The reward at the 'orange IN' cell is quite low as it takes us further away from 'GOAL 2'
- The rewards decreases as we go further away from the 'GOAL 2'
- The rewards at the 'gray IN' cell are higher than it's neighbours because these cells take us closer to the 'GOAL 2'
- The reward is lowest at the top right most corner and highest at the 'GOAL 2'

Note: The actions in the 'Goal' cell and the 'IN' cell are not valid as one cannot take action in those cells. The actions can be explained as follows:

- The actions around the 'orange IN' cell do not point into the cell as going into that cell takes to the 'orange OUT' which is further away from the 'GOAL 2'
- The actions around the 'gray IN' cell point towards the 'gray IN' cell as it will takes us to the 'gray OUT' which is near the 'GOAL 2'
- The actions around the 'GOAL 2' point into the goal which is quite obvious.
- Starting in any state we will always land up in 'GOAL 2' which means that there are no improper policies.

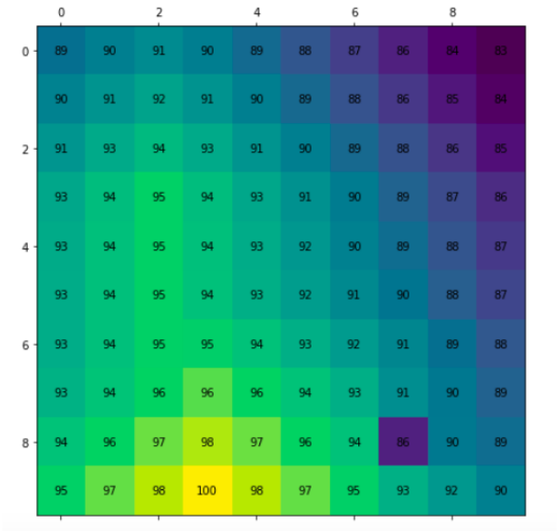

Figure 5: $J(s)$ after the termination of the loop

```
[['↓' '↓' '↓' '↓' '←' '←' '←' '←' '←' '←']
 ['↓' '↓' '↓' '↓' '↓' '←' '←' '←' '←' '←']
 ['↓' '↓' '↓' '↓' '←' '←' '←' '←' '←' '←']
 ['→' '→' '↓' '←' '←' '←' '←' '←' '←' '←']
 ['→' '→' '↑' '←' '←' '←' '←' '←' '←' '←']
 ['→' '→' '↑' '←' '←' '←' '←' '←' '←' '←']
 ['→' '→' '↓' '↓' '↓' '↓' '←' '←' '←' '←']
 ['→' '→' '↓' '↓' '↓' '↓' '←' '←' '←' '←']
 ['→' '→' '↓' '↓' '↓' '←' '←' '←' '↓' '↓']
 ['→' '→' '→' '↓' '←' '←' '←' '←' '←' '←']]
```

Figure 6: $\Pi(s)$ after the termination of the loop

5 Plots after 10 iterations

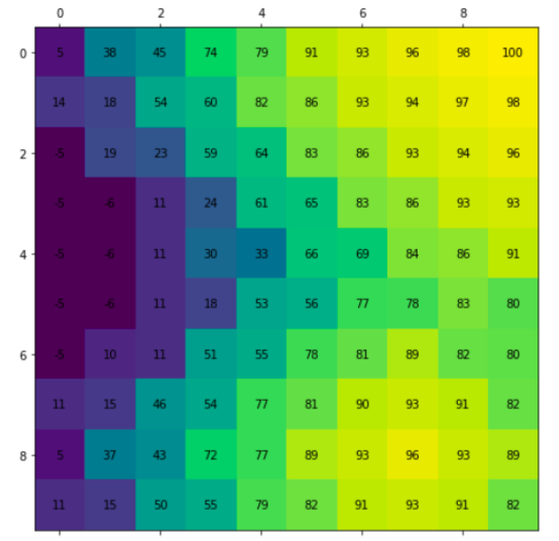

Figure 7: $J(s)$ after 10 iterations of the loop for GOAL 1

```
[['→' '→' '→' '→' '→' '→' '→' '→' '→' '↑']
 ['→' '→' '→' '→' '→' '→' '→' '→' '↑' '↑']
 ['→' '→' '→' '→' '→' '→' '→' '→' '↑' '↑']
 ['←' '↑' '→' '→' '→' '→' '→' '↑' '↑' '↑']
 ['←' '→' '→' '→' '→' '→' '→' '→' '↑' '↑']
 ['←' '→' '→' '→' '→' '→' '→' '↓' '↑' '↑']
 ['↓' '↓' '→' '→' '→' '→' '↓' '↓' '↓' '←']
 ['→' '→' '→' '→' '→' '→' '↓' '↓' '↓' '←']
 ['→' '→' '→' '→' '→' '→' '→' '↓' '←' '←']
 ['→' '→' '→' '→' '→' '→' '→' '↑' '←' '←']]
```

Figure 8: $\Pi(s)$ after 10 iterations of the loop for GOAL 1

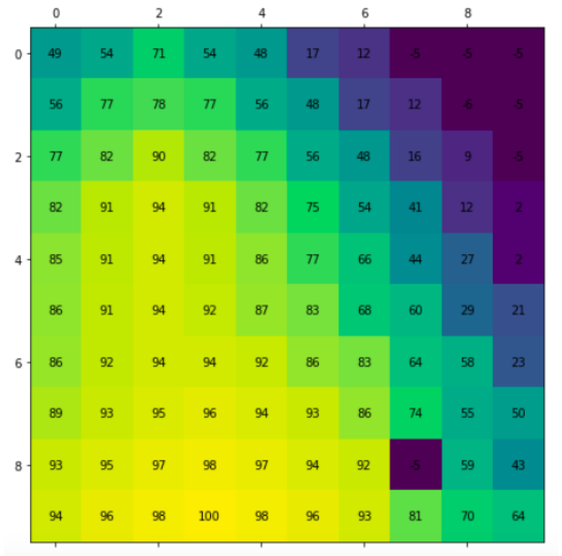

Figure 9: $J(s)$ after 10 iterations of the loop for GOAL 2

```
[['↓' '↓' '↓' '↓' '↓' '←' '←' '←' '↑' '↑']
 ['↓' '↓' '↓' '↓' '↓' '←' '←' '←' '←' '→']
 ['↓' '↓' '↓' '↓' '←' '←' '←' '←' '←' '←']
 ['→' '→' '↓' '←' '←' '←' '←' '←' '←' '←']
 ['→' '→' '↑' '←' '←' '←' '←' '←' '←' '←']
 ['→' '→' '↑' '←' '↓' '←' '←' '←' '←' '←']
 ['→' '→' '↓' '↓' '↓' '↓' '←' '←' '←' '←']
 ['→' '→' '↓' '↓' '↓' '↓' '←' '←' '←' '←']
 ['→' '→' '↓' '↓' '↓' '←' '←' '←' '↓' '↓']
 ['→' '→' '→' '↓' '←' '←' '←' '←' '←' '←']]
```

Figure 10: $\Pi(s)$ after 10 iterations of the loop for GOAL 2

6 Plots after 25 iterations

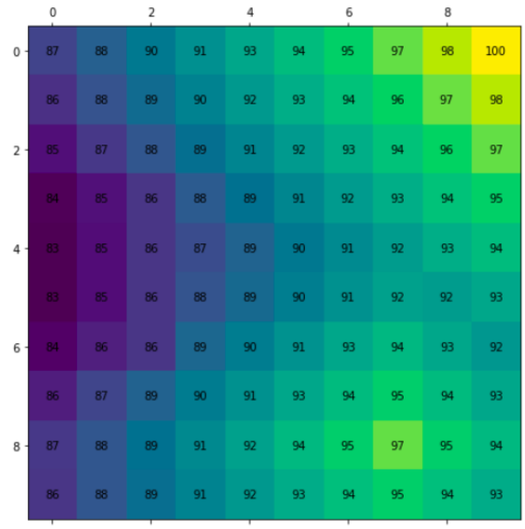

Figure 11: $J(s)$ after 25 iterations of the loop for GOAL 1

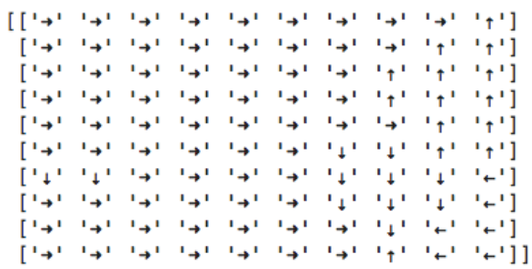

Figure 12: $\Pi(s)$ after 25 iterations of the loop for GOAL 1

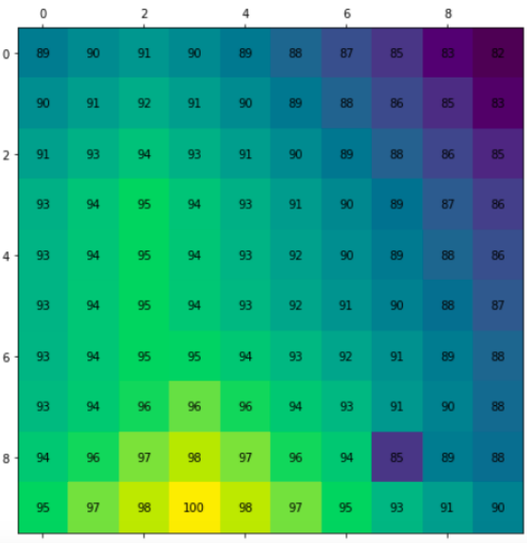

Figure 13: $J(s)$ after 25 iterations of the loop for GOAL 2

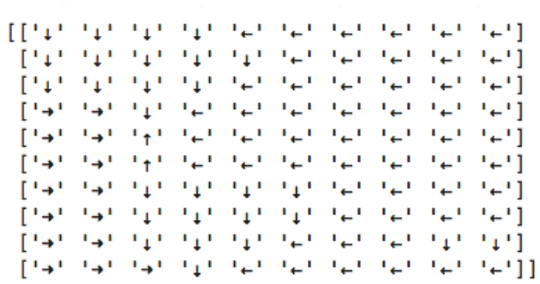

Figure 14: $\Pi(s)$ after 25 iterations of the loop for GOAL 2

7 References:

- Course Notes
- Dynamic Programming and Optimal Control by D.Bertsekas.