Programming Assignment 3

Purvam Jain EE20B101, Martin Reich ME23F201

November 13, 2023

Contents

1 SMDP Q-learning

1.1 Code Structure

Options are defined as : $goal = 0 : [0,0], 1 : [0,4], 2 : [4,0], 3 : [4,3]$ // And the intermediate reward is defined as:

```
move: number of time steps
    reward_bar += reward*(gamma**move)
```

We compress the Q-Value table from 500 X 10(6 actions and 4 options) to 20 X 4 (Passenger locations X Drop Locations and 4 options). The passenger locations are the four pickup locations and a in taxi state.

```
    nX = 5; nY = 5; nPas = 5; nDrop = 4
    x,y,pas,drop = env.decode(state)
    subState = nDrop*pas + drop
```

Only the four options are considered since primitive actions are negligibly updated compared to options.
Q-value Update rule:

```
    subprev: state
    move: number of timesteps
    q_values_SMDP[subPrev, option] += alpha*(reward_bar + (gamma**move)*
    np.max(q_values_SMDP[subState, :]) - q_values_SMDP[subPrev, option])
```

1.2 HyperParameter

Parameter	**value**
Policy	$\epsilon - Greedy$
Epsilon	0.5
Epsilon Decay rate	0.99
Alpha(learning rate)	0.1
Episodes	1500

Table 1: List of hyperparameters

1.3 Reward Plots and Q-Values

The environment is considered solved if the average reward for the previous 100 episodes is more than 9.7.

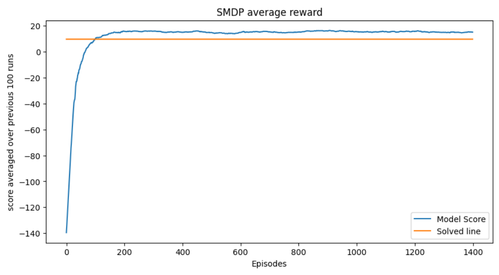
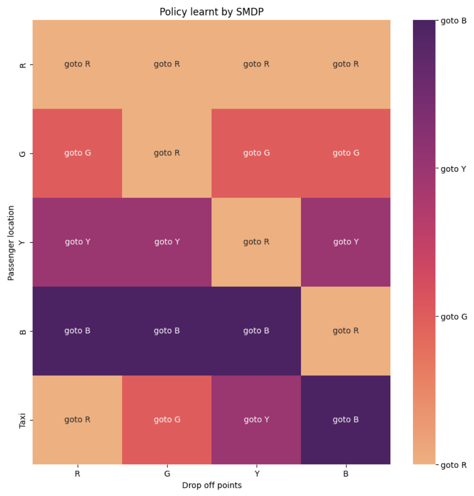


Figure 1: Running average of total reward over 100 episodes



	R	G	Y	B
R	goto R	goto R	goto R	goto R
G	goto G	goto R	goto G	goto G
Y	goto Y	goto Y	goto R	goto Y
B	goto B	goto B	goto B	goto R
Taxi	goto R	goto G	goto Y	goto B

Figure 2: Q-Value for SMDP specified as option

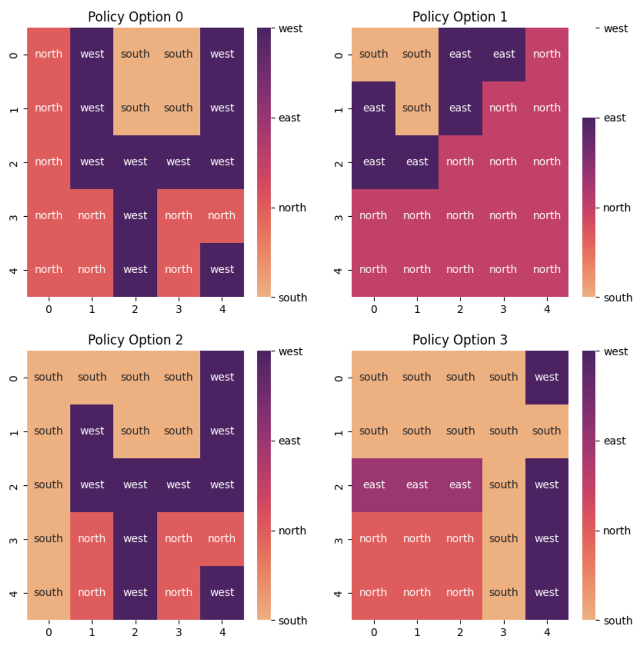


Figure 3: Q-Values for each option specified as option

Here, {0: Option R, 1: Option G, 2: Option Y, 3: Option B}

1.4 Policies

1. From looking at the Q-values for specific options, it is clear that the optimum policy is achieved such that they move towards their respective locations (R,Y,G,B).

2. From Figure 2, we can clearly see that as long as the passenger is not in the taxi, the correct option is chosen to go to the passenger's pick-up location.

3. When the passenger is in the taxi, it is clear the correct option is chosen depending on the drop-off location.

4. When the pick-up and drop-off locations are the same, the episode ends, and any random option can be chosen; as such, here, Option R is favoured due to initial Q-values and argmax function.

5. SMDP quickly learns the optimal policy in less than ≈ 150 episodes; the agent is able to learn the optimal policy to choose the correct option depending on the desired destination. This occurs because if a wrong option is selected, then it will always get a negative cumulative reward even if it reaches the correct destination according to the option definition.

6. Within the respective options, the policy is learnt to minimize the cumulative reward and reach the option destination as quickly as possible.

2 Intra Option Q-learning

2.1 Code Structure

The options and the states are the same as in 1.1. The updates of the Q-Values after each episode is the same as well. However, during an episode all the options that would have taken the same action are updated as well:

```
for o in range(number_options):
    optact_o,optdone_o = Option(env,state,Qopt_intra,o,eps[o])
    if optact_o == optact:
      eps[o] = max(eps_min,eps_decay*eps[o])
      _,_,pas_sub,drop_sub = env.decode(state)
      intra_opt_sub_state = nDrop * pas_sub + drop_sub
      if optdone_o:
        q_values_intra[intra_opt_sub_state, o] += alpha*(reward +
        gamma*np.max(q_values_intra[intra_opt_sub_state, :]) -
        q_values_intra[intra_opt_sub_state, o])
      else:
        q_values_intra[intra_opt_sub_state, o] += alpha*(reward +
        gamma*q_values_intra[intra_opt_sub_state, o] -
        q_values_intra[intra_opt_sub_state, o])
```

2.2 HyperParameter

Parameter	**value**
Policy	$\epsilon - Greedy$
Epsilon	0.5
Epsilon Decay rate	0.99
Alpha(learning rate)	0.1
Episodes	1500

Table 2: List of hyperparameters

2.3 Reward Plots and Q-Values

The environment is considered solved if the average reward for the previous 100 episodes is more than 9.7.

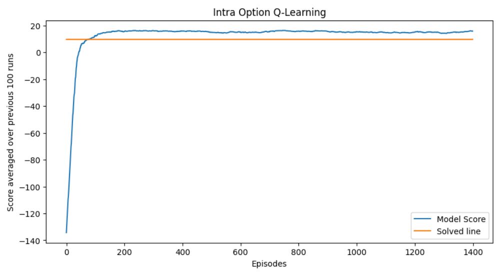
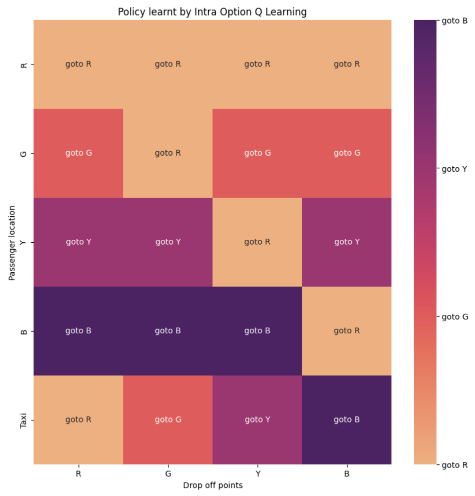


Figure 4: Running average of total reward over 100 episodes



	R	G	Y	B
R	goto R	goto R	goto R	goto R
G	goto G	goto R	goto G	goto G
Y	goto Y	goto Y	goto R	goto Y
B	goto B	goto B	goto B	goto R
Taxi	goto R	goto G	goto Y	goto B

Figure 5: Q-Value for Intra Option Q Learning encoded as option

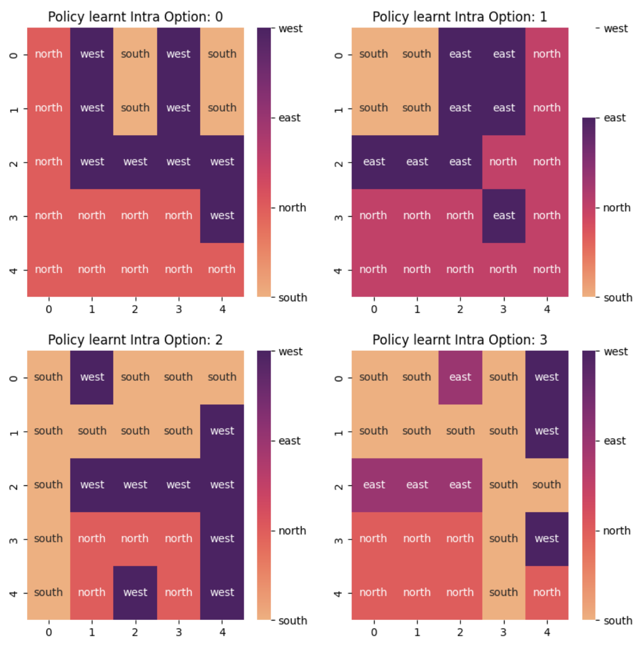


Figure 6: Q-Values for each option encoded as option

Here, {0: Option R, 1: Option G, 2: Option Y, 3: Option B}.

2.4 Policies

The learnt policies are almost similar to the ones learnt with SMDP in chapter 1.4. Only small differences can be spotted as for example in the policy learnt for going to Y (2), where in the top right corner the policy goes to the south first and then west afterwards. However, the Q-values are actually almost similar (-2.97379256 and -2.96552305) in that particular grid-cell, so going either way leads to the same rewards. This is due to the max function for plotting purposes and those small numerical differences are due to the exploration probabilities of moving in either direction.

3 Comparison

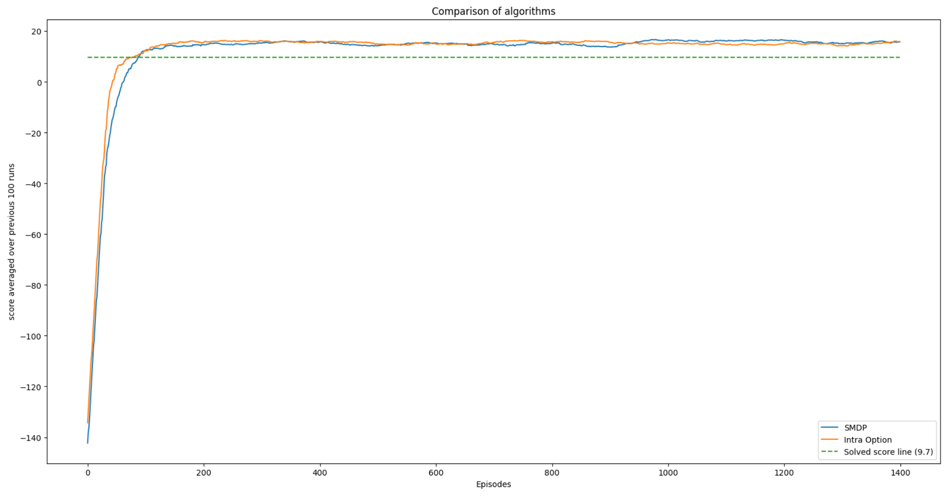


Figure 7: Comparison of the two algorithms

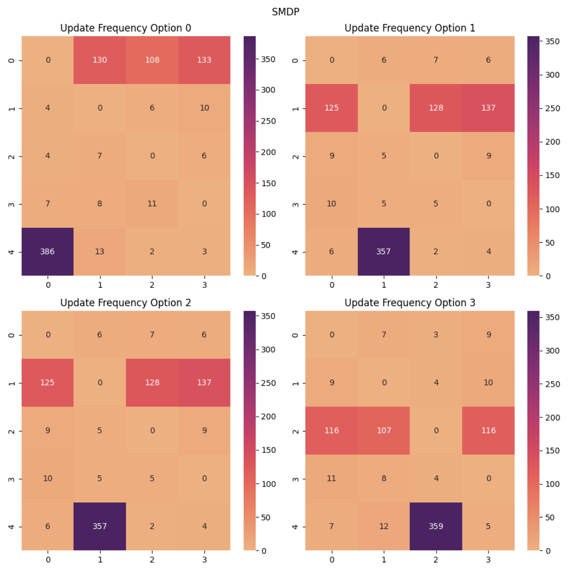


Figure 8: SMDP Update Frequency

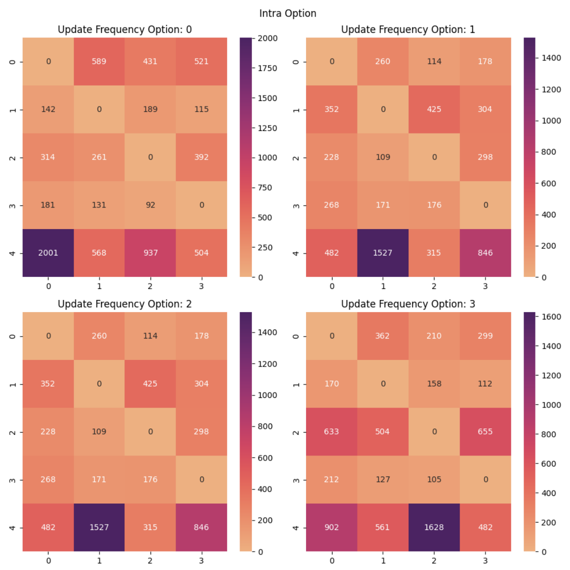


Figure 9: Intra Option Update Frequency

The two algorithms quickly learn the optimal policy, with Intra Option Q-Learning having a slightly higher average reward at the beginning (figure 7. This is due to the q-value updates received by the other options that would have taken the same action, as can be seen in figure 9 and 8. They both have a similar outcome in the long run. Intra-Option has significantly higher update frequencies for the same states compared to SMDP.

4 Alternate option set

Alternate Options: A mutually exclusive set of options could be: {0:Pick-up, 1:Drop-off}. Taxi must navigate to the expected location and pick up/drop off the passenger. This set is mutually exclusive to the original set as the original set doesn't include pickup/drop. 10

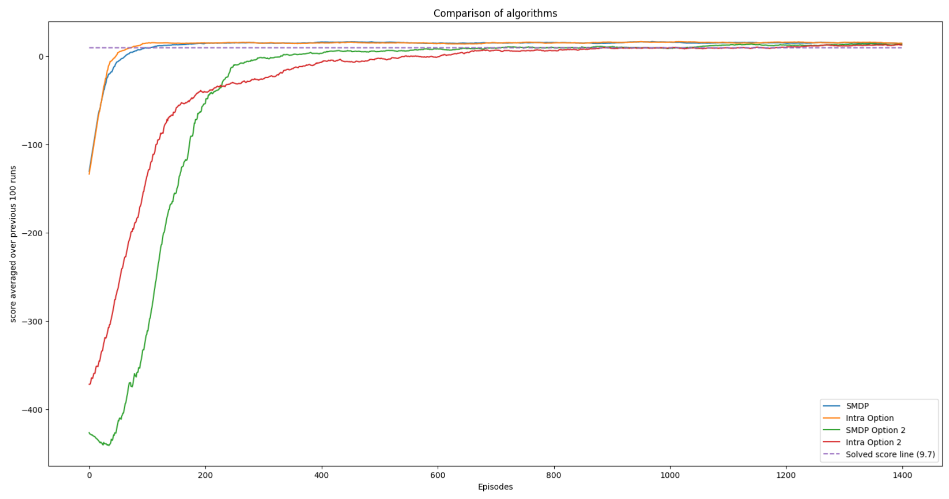


Figure 10: All options compared

This option set performs worse than the original set because in the original set, we decode all the passenger locations and drop locations through options which makes it easier just to learn option-specific policy, i.e., to navigate to the respective location and policy to select the best option. But here, it takes more iterations to learn the policy, as once an option is chosen which does not have the correct ending action (i.e. go to R and drop instead of pickup even though the passenger is not in the taxi), it is stuck with that and receives a penalty. The learning takes longer. This is avoided in the first option set where the pickup/drop is excluded.
Also, from this option set, we are better able to capture the difference between Intra-option learning and SMDP, as Intra-option performs better than SMDP since it receives updates at each step, unlike cumulative updates received in the case of SMDP. So initially, we receive large negative rewards in SMDP as the agent tries to learn the correct policy, and hence it performs worse than Intra Option.