CS533-Assignment4-Report

Chunxiao Wang

1 Part I: Designing the MDPs

We specify our MDP using suggested structure. To make the location L more explicit, we further split it into column $C \in \{A, B\}$ and row $R \in \{1, 2, \ldots, n\}$ with n being the number of parking rows. Therefore each state is a 4-tuple (C, R, O, P). Here we set n as 10.
We select two sets of parameters. The only difference is in the assumption of the negative reward that represents the cost of driving. For the first set, we use the -1 representing the cost of driving and for the second set, we use the -20 representing the cost of driving. The other parameters remain the same. Intuitively, the average reward of the MDP created by parameter set 1 should be larger than it of the MDP created by parameter set 2 after running many trials of a specific policy such as random parking policy or safe parking policy since the punishment not parking is much higher in parameter set 2. We will verify this in Part II. The other qualitative characteristics are as follows,

- The reward for parking as a function of being in row $r \in \{2, 3, \ldots, n\}$ is $(n-(r-1)) \cdot 10$, where 10 is the reward coefficient which can be changed. The driver will get more reward if they park closer to the store except the handicap spots. The high cost for parking in the handicap spots is -1000 and for collision is -10000. The reward for the terminal state is 1. The terminal state has no transitions.
- After a DRIVE action, if the next spot is a handicap spot $(A[1], B[1])$, the probability that they are occupied is a small value fixed at 0.01. If the next spot is not a handicap spot, the probability that the next spot is occupied is a linear function of the row r: $(n-(r-1))/n$. As the row gets closer to the store, the occupied probability increases.

2 Policy Simulation

2.1 Policy Design

I use three policies here, they are specified as follows,

- The random parking policy. For this policy, the agent selects the PARK action with probability 0.1 and DRIVE action 0.9.

- The safe parking policy. For this policy, the agent selects the DRIVE action whenever O is true and when O is false, the agent selects PARK action with probability 0.1 and DRIVE action with probability 0.9.
- The optimal policy. This policy is derived from the policy iteration for MDP1 and MDP2 under infinite horizon case.

2.2 Result and Discussion

After 1000 simulations, the average rewards of each policy for MDP1 and MDP2 are as follows,

Table 1: policy simulation

	random parking	safe parking	optimal
MDP1	-4032.96	-3614.83	86.54
MDP2	-4237.84	-3917.33	58.96

From the result, for both MDPs, we could see the optimal policy gives the highest average rewards, the safe parking policy gives the intermediate average rewards and the random parking gives the lowest rewards. And since the not parking punishment is higher in parameter set 2, the average rewards for MDP2 under each policy is smaller than it of MDP1 under each policy.

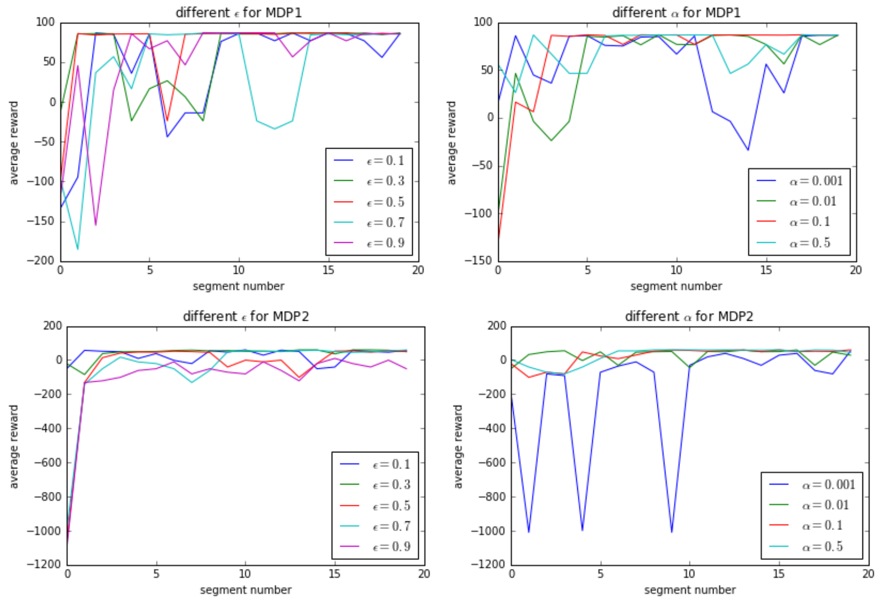
3 Reinforcement Learning

3.1 Design

We implemented a model-free reinforcement learning agent for the environment using Q-learning. The goal is to update the states and actions in the Q-table. For the explore/exploit policy, with probability ϵ, we choose a random action from the available actions for exploration, otherwise we exploit using the current Q-table. We explore different ϵs with fixed $\alpha = 0.01$ and different αs with fixed $\epsilon = 0.1$ in the experiments. The reverse trajectory update is used in the learning trial.

3.2 Result

To see significant change of the policy learning, we run $N = 100$ learning trials to perform Q-learning, then run $1,000$ trials without learning to get the average total reward and we repeat this 20 times. The plots of results of ϵ and α exploration for MDP1 and MDP2 are as follows,



3.3 Discussion

Firstly, we discuss the effect of ϵ. Overall the plot indicates that the Q-learning algorithm is successfully learning the optimal policy through Q-tables since the average total reward converges to it derived the optimal policy for all the cases. For MDP1, we see in upper left figure that for $\epsilon = 0.1, 0.3, 0.5$ the average reward eventually converges at segment 10(there is some oscillation for $\epsilon = 0.1$) to the average reward derived from the optimal policy. For $\epsilon = 0.7$ and $\epsilon = 0.9$, the convergence is slow. The average reward converges to the average reward derived from the optimal policy at segment 14 for $\epsilon = 0.7$ and 16 for $\epsilon = 0.9$. For MDP 2, from the left lower plot, we see the average reward converges after segment 15 for all the cases except $\epsilon = 0.9$. The slow convergence for larger ϵ makes sense since a larger ϵ means higher probability of exploration. Larger probability of exploration is good at the beginning when we have no information. However as we gain more information, exploitation is more preferable.

Then we discuss the effect of learning rate α. For MDP1, from the upper right plot, for $\alpha = 0.001, 0.01$ the average reward grows slower and will probably converges to the average reward derived from the optimal policy after another few segments. For $\alpha = 0.5$, it converges at first, but then oscillates to lower values. For $\alpha = 0.1$, the average reward converges at segment 4. For MDP2, from the lower right plot, the average reward converges fast after at segment 7 to the average reward derived from the optimal policy for $\epsilon = 0.1, 0.5$. For $\epsilon = 0.01$, there is still small oscillation, but for $\epsilon = 0.001$, the average reward grows slower and will probably converges to the average reward derived from the optimal policy after more segment. The result matches with the basic learning formula. Overall, the lower the α is, the more slowly until convergence. The higher the α, the faster the training but a very high α may cause instability issue.