Homework 5

Branch Vincent

December 1, 2017

Problem 1

The MDP is represented in Figure 1.

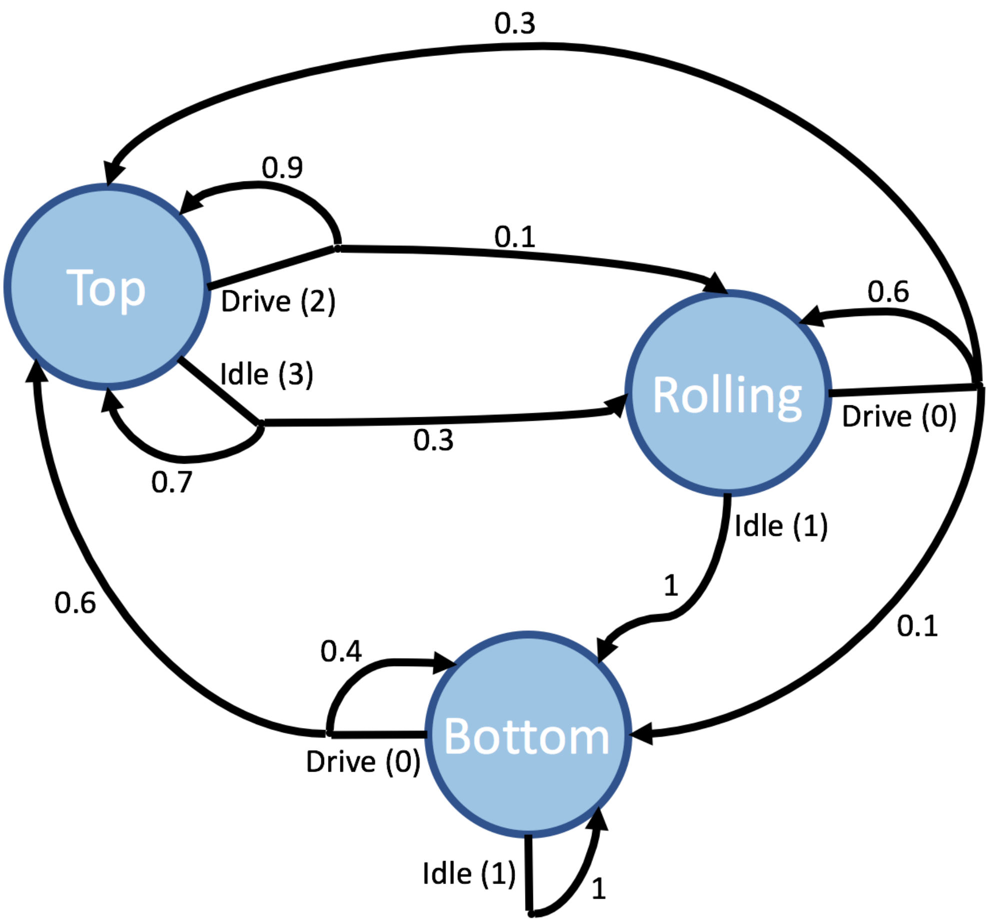

Figure 1: Markov Decision Process

Problem 2

Using value iteration and a discount factor $\delta = 0.8$, we arrive at the following optimal policy and values (given a state s).

$$\pi^*(s) = \begin{cases} \text{idle} & s = \text{top or rolling} \\ \text{drive} & s = \text{bottom} \end{cases}$$

$$v^*(s) \approx \begin{cases} 10.64 & s = \text{top} \\ 7.01 & s = \text{rolling} \\ 7.51 & s = \text{bottom} \end{cases}$$

Problem 3

Using policy iteration and $\delta = 0.8$, we arrive at the following optimal policy and values.

$$\pi^*(s) = \begin{cases} \text{idle} & s = \text{top or rolling} \\ \text{drive} & s = \text{bottom} \end{cases}$$

$$v^*(s) \approx \begin{cases} 10.64 & s = \text{top} \\ 7.01 & s = \text{rolling} \\ 7.51 & s = \text{bottom} \end{cases}$$

Problem 4

(i) Let $\delta = 0.5$. Then

$$\pi^*(s) = \begin{cases} \text{idle} & \forall\, s \end{cases}$$

This is because we now more heavily discount the future. Before at $s =$ bottom, the optimal policy was to drive in an attempt to reach the top, where we can collect a higher reward. However, since we now care less about the future, the optimal policy becomes collecting a smaller reward now.

(ii) Let the transition probability for $a =$ drive at $s =$ bottom be

$$T(\text{bottom, drive}) = \begin{cases} 0.1 & s' = \text{top} \\ 0.9 & s' = \text{rolling} \end{cases}.$$

Then

$$\pi^*(s) = \begin{cases} \text{idle} & \forall\, s \end{cases}$$

This is because, once at $s =$ bottom, we now are less likely to reach the top again. Thus, it is no longer worth trying to reach the top while collecting no reward. Instead, the optimal policy becomes collecting a smaller (but certain) reward by idling at the bottom.

(iii) Let the reward for $a =$ idle at $s =$ bottom be 3. Then

$$\pi^*(s) = \begin{cases} \text{idle} & \forall\, s \end{cases}$$

This is because the marginal benefit of driving to reach the top at $s =$ bottom is now zero, since we collect the same reward for idling at the top and at the bottom. Thus, the optimal policy becomes to always idle.

Appendix

The preceding problems were solved using the attached code.