EL2805 Reinforcement Learning

Exercise Session 3

November 19, 2018

Department of Automatic Control
School of Electrical Engineering
KTH Royal Institute of Technology

3 Exercises

Some of these exercises have been inspired by, or taken from, [1–4]. If you want to solve more exercises, see any of the books.

3.1

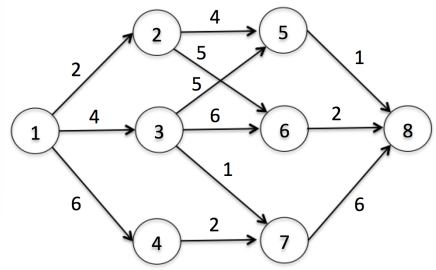

Figure 1: Subway system.

In Figure 3.1, a schematic overview of a town's subway system is shown. The nodes represent subway stations and the number on each arrow the (expected) time it takes to travel from one station to the other. Assume that whenever you arrive to a station, there is a train leaving. Your home is located in node 1, and your work in node 8. You want to find the best travel path so as to get to work as fast as possible. Note that it is only possible to take trains in the directions indicated by the arrows.

(a) Model the problem as an MDP.

(b) Solve the MDP.

3.2

Consider a production machine on a factory floor that can be in two states: functional or broken. When operating the machine, it has a probability θ of breaking down. The factory owner can choose to replace the machine at a cost R. If it is broken, then he acquires a cost c for not being able to produce new products. He wants to find an optimal policy for T time-steps that minimizes his expenses. Assume that the cost at the end of the horizon is zero, regardless of state.

(a) Model the problem as an MDP.

(b) For $\theta = 0.1$, $R = 10$, $c = 5$, $T = 3$, solve the MDP *by hand*. That is, compute the optimal cost-to-go and the optimal policy.

3.3

You have to sell your apartment within N days, and want to maximize your profit. Every evening, you receive an offer w_t, which you have to either accept or reject the next day. Assume that all offers are multiples of 10 000 SEK, and that they are i.i.d., positive and bounded. Their distribution is known to you from previous sales in your neighbourhood. Once you sell your apartment, you receive a fixed daily interest rate $\rho > 0$ on the money that land in your bank account. Model this as an MDP, and derive the optimal policy.

3.4

Given a sequence of matrix multiplications

$$M_1 M_2 \dots M_k M_{k+1} \dots M_N,$$

with each matrix M_k has dimension $n_k \times n_{k+1}$. The order in which multiplications are carried out can make a difference. For example, if $n_1 = 1$, $n_2 = 10$, $n_3 = 1$, and $n_4 = 10$, the calculation $((M_1 M_2)M_3)$ requires 20 scalar multiplications, whereas $(M_1(M_2 M_3))$ requires 200 scalar multiplications.

Derive a dynamic programming algorithm for finding the optimal multiplication order. Solve the problem for $N = 3$, $n_1 = 2$, $n_2 = 10$, and $n_4 = 1$.

3.5

Derive a dynamic programming algorithm to solve the following knapsack problem:

$$\begin{aligned} \max_x \quad & \sum_{i=1}^{N} c_i x_i \\ \text{subject to :} \quad & \sum_{i=1}^{N} w_i x_i \leq W, \\ & x_i \in \{0, 1\}, \quad \forall i. \end{aligned}$$

Here, c_i for all i, are positive numbers and w_i for all i and W are positive integers.

3.6

An unemployed worker receives a job offer at each time period, which she may accept or reject. The offered salary takes one of n possible values $w_1, \dots, w_n$ with given probabilities independently of preceding offers. If she accepts the offer, she must keep the job for the rest of her life. If she rejects the offer, she receives unemployment compensation c for the current period and is eligible to accept future offers. Assume that income is discounted by a factor $\lambda < 1$.

a) Show that there is a threshold $\overline{w}$ such that it is optimal to accept an offer if and only if its salary is larger than $\overline{w}$, and characterize $\overline{w}$.

b) Now assume that the worker will be fired from her job with a given probability p_i (when her salary is w_i). If she is fired, she will try to find a new job. Show that the result of part (a) holds if $p_i = p$ for all i.

3.7

A person has an umbrella that she takes from home to office and vice versa. There is a probability p of rain at the time she leaves home or office independently of earlier weather. If the umbrella is in the place where she is and it rains, she takes the umbrella to go to the other place (this involves no cost). If there is no umbrella and it rains, there is a cost of W for getting wet. If the umbrella is in the place where she is but it does not rain, she may take the umbrella to go to the other place incurring an inconvenience cost U, or she may leave the umbrella (hence no cost). There is a discount factor $\lambda < 1$. Formulate this as an infinite horizon MDP and characterize the optimal policy.

3.8

Consider an MDP problem with state space $\{0, 1, \ldots, M\}$, where in each state there are two possible actions, i.e., $A_s = \{-1, +1\}$ for all s. In each state s, $P(s'|s, a) = \mathbb{1}\{s' = s + a\}$, that is $P(s'|s, a)$ puts probability mass 1 on state $s' = s + a$. The reward function is specified as below: $r(s, a) = 0$ when $s = 0, M$. For states $s = 1, 2, \ldots, M - 2$, we have $r(s, -1) = -1$ and $r(s, +1) = -2$. The states $s = 0$ and $s = M$ are zerocost absorbing states regardless of the action taken. Finally, $r(M - 1, -1) = -1$ and $r(M - 1, +1) = 2M$.

Starting from $V(s) = 0$ for all s, how long does it take for Policy Iteration to find the optimal policy?

3.9

Consider an MDP with state space $\{0, 1, 2\}$. At state 0, action 1 results in a zero cost and a transition to state 1, whereas action 2 results in a cost of $\lambda^2/(1 - \lambda)$ and a transition to state 2. States 1 and 2 are absorbing states with respective costs of 1 and 0.

Starting from $V_0(s) = 0$ for all s, and assuming a discount factor $\lambda < 1$, derive the time it takes for Value Iteration to identify the optimal policy.

3.10

An unscrupulous innkeeper charges a different rate for a room as the day progresses, depending on whether has has many of few vacancies. His objective is to maximize his expected total income during the day. Let x be the total number of empty rooms at the start of the day, and let y be the number of customers that will ask for a room in the course of the day. We assume (somewhat unrealistically) that the innkeeper knows y with certainty, and upon arrival of a customer, quotes one of m prices r_i, $i = 1, \ldots, m$, where $0 < r_1 \leq r_2 \leq \ldots \leq r_m$. A quote of a rate r_i is accepted with probability p_i and is rejected with probability $1p_i$, in which case the customer departs, never to return during that day.

(a) Show that the maximal expected income as a function of x and y satisfies:

$$V(x, y) = \max_{i=1,\ldots,m} \left[p_i(r_i + V(x - 1, y - 1)) + (1 - p_i)V(x, y - 1)\right],$$

for all $x \geq 1$ and $y \geq 1$ with $V(x, 0) = V(0, y) = 0$ for all x and y.

(b) Show that $V(x - 1, y) \leq V(x, y)$ for all $x \geq 1$ and $y \geq 0$.

(c) Assume that

$$p_1 r_1 \leq p_2 r_2 \leq \ldots \leq p_m r_m$$
$$p_1 \geq p_2 \geq \ldots \geq p_m.$$

Show that the innkeeper should always charge the highest rate r_m.

3.11

Suppose that at each epoch, a production machine is inspected and its condition and state is noted. States are $\{0, 1, 2, \ldots\}$ with state 0 being "perfectly new". With each state i, an operating cost of $C(i)$ is incurred at each epoch where $C(\cdot)$ is assumed to be an increasing function in i. After inspecting the state, at each epoch a decision is made: $a = 0$ (not replace the machine) or $a = 1$ (replace the machine). If the decision is to replace, a cost $R > 0$ is immediately incurred. This then moves the state of the machine to 0 for the next epoch. If the decision is not to replace then the condition of the machine evolves randomly according to the probabilities P_{ij}, namely machine goes to state j if not replaced. We assume the following about P_{ij} : For each k, $\sum_{j=k}^{\infty} P_{ij}$ is an increasing function of i. This means that if T_i is a random variable representing the next state visited after i (assuming no replacement) then,

$$\mathbb{P}(T_{i+1} > k) \geq \mathbb{P}(T_i > k).$$

This is known as a stochastic order and is equivalent to:

$$\mathbb{E}[f(T_{i+1})] \geq \mathbb{E}[f(T_i)]$$

for all increasing functions f.

a) Pose the problem as an MDP with discount factor $\lambda < 1$ and derive Value Iteration for it.

b) Use the Value Iteration, starting from $V_0(s) = C(s)$ for all s, to prove that $V(i)$ is an increasing function.

c) Prove that there exists an i_{th} such that an optimal policy is to replace when $i \geq i_{\text{th}}$ and to keep if $i < i_{\text{th}}$.

4 Solutions

Solution to Problem 3.1

See Lecture 2 (Part 2), slides 25-27.

Solution to Problem 3.2

Part a)

- **State space:** $\mathcal{S} = \{\text{Functional}, \text{Broken}\}$.
- **Actions:** $\mathcal{C}$ - Continue, $\mathcal{R}$ - Repair.
- **Rewards:** Terminal: $r_T(\cdot) = 0$.
  Non-terminal:
  - $r_t(s = F, a = \mathcal{C}) = 0$,
  - $r_t(s = B, a = \mathcal{C}) = -c$,
  - $r_t(s = F, a = \mathcal{R}) = -R$,
  - $r_t(s = B, a = \mathcal{R}) = -R$.
- **Transitions:** $P(\mathcal{C}) = \begin{bmatrix} 1-\theta & \theta \\ 0 & 1 \end{bmatrix}$ and $P(\mathcal{R}) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$.
- **Time-horizon and objective:** Finite-horizon $T < \infty$, $\mathbb{E}\{\sum_{t=0}^{T-1} r(s_t, a_t) + r_T(s_T)\}$.

Part b)

$u* =$

F	-1.45	-0.5	0	0
B	-10.5	-10.0	-5.0	0

$\pi* =$

F	$\mathcal{C}$	$\mathcal{C}$	$\mathcal{C}$	$\cdot$
B	$\mathcal{R}$	$\mathcal{C}/\mathcal{R}$	$\mathcal{C}$	$\cdot$

Solution to Problem 3.3

The MDP model was derived in the previous exercise session. However, to simplify, assume that we always talk in units of 10 000 SEK, so that the state-space is $\mathcal{S} = \{0, 1, \ldots, b_{\max}\} \cup \{\text{Sold}\}$. By performing backward induction on the value function, one obtains the following threshold policy: If we recive an offer b at time t, then we should

$$\mathcal{A}\text{ccept the offer } b \text{ if } b > \alpha_t,$$
$$\mathcal{R}\text{eject the offer } b \text{ if } b \leq \alpha_t,$$

where

$$\alpha_t = \frac{\sum_{i=0}^{b_{\max}} \Pr\{w_t = i\} u_{t+1}^*(i)}{(1+\rho)^{N-t}}.$$

Solution to Problem 3.4

State space for the corresponding MDP is

$$S = \{(1,1), (1,2), \ldots, (1,N), (2,2), (2,3), \ldots, (2,N), (3,3), \ldots, (N,N)\}.$$

At each state (i,j), we have $A_{(i,j)} = \{i, i+1, \ldots, j-1\}$. For each state $s = (i,j) \in S$, we let $V(i,j)$ be the minimum number of multiplications needed to compute $M_i M_{i+1} \ldots M_j$. Then, using dynamic programming, we have the following recursion:

Algorithm 1 Algorithm for solving Knapsack problem

Set $V(0,u) = 0$ for all $0 \leq u \leq W$.
for $i = 1 \dots n$ **do**
  for $u = 0 \dots W$ **do**
    if $(w_i \leq u)$ `AND` $\Big(c_i + V(i-1, u-w_i) > V(i-1,u)\Big)$ **then**
      $V(i,u) = c_i + V(i-1, u-w_i)$
      $K(i,u) = 1$
    else
      $V(i,u) = V(i-1,u)$
      $K(i,u) = 0$
    end if
  end for
end for
Set $R = W$.
for $i = n \dots 1$ **do**
  if $K(i,R) = 1$ **then**
    $x_i^\star = 1$
    $R \leftarrow R - w_i$
  else
    $x_i^\star = 0$
  end if
end for

$$V(i,j) = \min_{i \leq k < j} (V(i,k) + V(k+1,j) + n_{i-1} n_k n_j), \quad i < j$$
$$V(i,i) = 0.$$

This recursion follows easily from the fact that $A_i A_{i+1} \dots A_j = (A_i \dots A_k)(A_{k+1} \dots A_j)$.

The minimal multiplication cost will be $V(1,N)$, which can be computed recursively. To recover the optimal chain for multiplication, it would be enough to save at each step the index k which minimizes the relation for $V(i,j)$.

For the provided example, the optimal ordering for the given problem is $M_1(M_2M_3)$.

Solution to Problem 3.5

We define the state space $S = \{0, 1, \dots, n\} \times \{0, 1, \dots, W\}$. For all $1 \leq i \leq n$ and $0 \leq u \leq W$:

$$V(0,u) = 0,$$
$$V(i,u) = \max\{V(i-1,u), (c_i + V(i-1, u-w_i))\mathbb{1}\{u \geq w_i\}\}.$$

This recursion yields $V(n,W)$, which corresponds to the optimal value of the knapsack problem. The pseudo-code of this recursive procedure is shown in Algorithm **??**. Clearly, the time complexity of this procedure is $\mathcal{O}(nW)$.

Solution to Problem 3.6

Solution. *(Part a).)* The state space is $S = \{s_1, s_2, \dots, s_n, s_1', \dots, s_n'\}$, where for each i, s_i corresponds to the case where the worker is unemployed and being offered a salary w_i, and s_i' corresponds to the case where she is employed at a salary level w_i. Let q_i be the probability of an offer at salary level w_i at any one period.

We have $A_{s_i} = \{C, \overline{C}\}$ for all i, where C denotes the action corresponding to accepting an offer ($\overline{C}$ rejecting the offer). Furthermore, $A_{s_i'} = \{X\}$ for all i, where X indicates continuation of the job.

Rewards are given by: For all i, $r(s_i, C) = w_i$, $r(s_i, \overline{C}) = c$, and $r(s'_i, X) = w_i$.
Transition probabilities are given by:

$$p(s'_i|s'_i, X) = 1, \quad p(s'_i|s_i, C) = 1, \quad p(s_j|s_i, \overline{C}) = q_j, \quad \forall i.$$

Bellman's equations are:

$$V(s_i) = \max\Big\{c + \lambda \sum_{j=1}^{n} q_j V(s_j), w_i + \lambda V(s'_i)\Big\}, \quad i = 1, \ldots, n,$$
$$V(s'_i) = w_i + \lambda V(s'_i), \quad i = 1, \ldots, n.$$

Simplifying, we obtain:

$$V(s'_i) = \frac{w_i}{1-\lambda}, \quad \forall i,$$
$$V(s_i) = \max\Big\{c + \lambda \sum_{j=1}^{n} q_j V(s_j), \frac{w_i}{1-\lambda}\Big\}, \quad \forall i.$$

Observe that $c + \lambda \sum_{j=1}^{n} q_j V(s_j)$ is independent of i. Thus, there exists a threshold $\overline{w}$ such that it is optimal to accept an offer w_i if

$$w_i \geq (1-\lambda)\Big(c + \lambda \sum_{j=1}^{n} q_j V(s_j)\Big) := \overline{w}.$$

Part b). In this case, Bellman's equations become:

$$V(s_i) = \max\Big\{c + \lambda \sum_{j=1}^{n} q_j V(s_j), w_i + \lambda\Big((1-p_i)V(s'_i) + p_i \sum_{j=1}^{n} q_j V(s_j)\Big)\Big\}, \quad i = 1, \ldots, n,$$
$$V(s'_i) = w_i + \lambda\Big((1-p_i)V(s'_i) + p_i \sum_{j=1}^{n} q_j V(s_j)\Big), \quad i = 1, \ldots, n.$$

Thus, for any i,

$$V(s'_i) = \frac{w_i + \lambda p_i \sum_{j=1}^{n} q_j V(s_j)}{1 - \lambda(1-p_i)}.$$

Assume, without loss of generality, that $w_1 < w_2 < \ldots < w_n$. Now, if $p_i = p$, we have that

$$V(s'_i) = \frac{w_i + \lambda p \sum_{j=1}^{n} q_j V(s_j)}{1 - \lambda(1-p)},$$

so it follows that

$$V(s'_1) < V(s'_2) < \ldots < V(s'_n).$$

Therefore, the second term in the maximization for $V(s_i)$ is $w_i + V(s'_i)$, which is monotonically increasing in i. Observing that the first term is independent of i, we deduce that there is a salary threshold above which the offer is accepted.

Solution to Problem 3.7

Define the set of states $S = \{s, \overline{s}, o\}$, where

$s \equiv$ the umbrella and the person are in the same locations and it rains,
$\overline{s} \equiv$ the umbrella and the person are in the same locations and it doesn't rain,
$o \equiv$ the umbrella and the person are in diffetent locations.

In state $\overline{s}$, the person has two actions (to take the umbrella or not) $A_{\overline{s}} = \{T, \overline{T}\}$, where T and $\overline{T}$ respectively correspond to 'take the umbrella' and 'leave the umbrella'. In state s she takes the umbrella so that $A_s = \{T\}$. Finally, we have $A_o = \{\overline{T}\}$.

Transition probabilities are given by:

$$\begin{aligned} &p(s|\overline{s}, T) = p, \quad p(\overline{s}|\overline{s}, T) = 1 - p, \quad p(o|\overline{s}, \overline{T}) = 1, \\ &p(s|s, T) = p, \quad p(\overline{s}|s, T) = 1 - p, \\ &p(s|o, \overline{T}) = p, \quad p(\overline{s}|o, \overline{T}) = 1 - p. \end{aligned}$$

The rewards are given by:

$$\begin{aligned} &r(s, T) = 0, \\ &r(\overline{s}, T) = -U, \quad r(\overline{s}, \overline{T}) = 0, \\ &r(o, \overline{T}) = -pW. \end{aligned}$$

Bellman's equations yield:

$$\begin{aligned} V(o) &= pW + \lambda p V(s) + \lambda(1-p)V(\overline{s}) \\ V(s) &= \lambda p V(s) + \lambda(1-p)V(\overline{s}) \\ V(\overline{s}) &= \min\{\lambda V(o), U + \lambda p V(s) + \lambda(1-p)V(\overline{s})\}. \end{aligned}$$

Clearly, $V(\overline{s}) = \frac{1-\lambda p}{\lambda(1-p)} V(s)$. Then, $V(o) = pW + V(s)$ and $V(\overline{s}) = \min\{\lambda V(o), U + V(s)\}$. Hence,

$$\frac{1-\lambda p}{\lambda(1-p)} V(s) = \min\{\lambda V(o), U + V(s)\}.$$

We focus on the Bellman's equation for state $\overline{s}$, i.e., the above minimization. She takes the umbrella in state $\overline{s}$ (i.e., she takes action T) if

$$\lambda V(o) > U + V(s). \tag{1}$$

Under the above condition, $\frac{1-\lambda p}{\lambda(1-p)} V(s) = U + V(s)$ and hence,

$$V(s) = \frac{U\lambda(1-p)}{1-\lambda}.$$

Substituting the expression for $V(s)$ into (**??**), and solving for p, we deduce that the optimal policy is to take the umbrella whenever possible if:

$$p > \frac{U}{U+W} \cdot \frac{1+\lambda}{\lambda}.$$

Solution to Problem 3.8

Starting from the initial value function $V_0(s) = 0$ for all s, we see that the optimal policy π_0 implied by this starting choice is $\pi_0(s) = -1$ for all $s = 1, \ldots, M-2$ and $\pi_0(M-1) = +1$. The choice of policy is irrelevant at $s = 0, M$. When $\lambda = 1$, the value V_1 generated by the first PI step is:

$$V_1(s) = \begin{cases} 0 & \text{if } s = 0, M, \\ -s & \text{if } s = 1, 2, \ldots, M-2, \\ 2M & \text{if } s = M-1. \end{cases} \tag{2}$$

For the second step, π_1 differs from π_0 in only one state, $s = M-2$, where it is now optimal to choose $a = +1$ at $s = M-2$. Moreover, after each succeeding PI step, the updated decision rule π_n differs from the previous policy π_{n-1} in state $s = M-n-1$, flipping the optimal action from left to right. This process continues for $M-1$ PI step until the optimal policy $\pi^\star(s) = +1$ for all s is achieved. The optimal value function $V^\star$ associated to this policy is then given by

$$V^\star(s) = \begin{cases} 0 & \text{if } s = 0, M, \\ 2(s+1) & \text{if } s = 1, 2, \ldots, M-1. \end{cases} \tag{3}$$

Solution to Problem 3.9

Starting from $V_0(s) = 0$ for all s, Value Iteration gives:

$$\begin{aligned} V_n(0) &= \min\{\lambda V_{n-1}(1), \lambda^2/(1-\lambda) + \lambda V_{n-1}(2)\}, \\ V_n(1) &= 1 + \lambda V_{n-1}(1), \\ V_n(2) &= 0 + \lambda V_{n-1}(2). \end{aligned}$$

Hence, starting from $V_0(s) = 0$ for all s, yields

$$\begin{aligned} V_n(1) &= 1 + \lambda + \lambda^2 + \ldots + \lambda^{n-1} = \frac{1-\lambda^n}{1-\lambda}, \\ V_n(2) &= 0. \end{aligned}$$

Assuming that VI converges after N iterations, Value Iteration continues to choose the suboptimal action at state 0 until iteration N where $\lambda V_{N-1}(1) > \lambda^2/(1-\lambda) + \lambda V_{N-1}(2)$. Hence, N satisfies

$$\frac{\lambda(1-\lambda^{N-1})}{1-\lambda} > \frac{\lambda^2}{1-\lambda},$$

and hence, $N > \frac{\log(1-\lambda)}{\log(\lambda)} + 1$.

Solution to Problem 3.10

(Part a).) Let states be equal to the number of free rooms $S = \{0, 1, \ldots\}$. As state $x \geq 1$, the innkeeper has $A_x = \{1, 2, \ldots, m\}$, where action i corresponds to quoting a rate r_i. We further have $A_0 = \emptyset$.

Transition probabilities are given below:

$$p(x-1|x,i) = p_i, \quad p(x|x,i) = 1 - p_i, \quad \forall x \geq 1.$$

Furthermore, the innkeeper's reward function is: $r(x,i) = p_i r_i$ for all i and $x \geq 1$, and $r(0,\cdot) = 0$. It is assumed that p_i is monotonically nondecreasing in i.

Dynamic programming for this problem yiealds:

$$\begin{aligned} &V(x,0) = 0, \quad \forall x \geq 0, \\ &V(x,y) = \max_{i=1,\ldots,m} [p_i(r_i + V(x-1,y-1)) + (1-p_i)V(x,y-1)], \quad \forall x \geq 0, \\ &V(0,y) = 0, \quad \forall y. \end{aligned}$$

Part b). We prove by induction on y that $V(x,y) \geq V(x-1,y)$ for all $x \geq 1$ and all y. Indeed this is true for $y = 0$. Assuming that this is true for for a given y, we will prove that

$$V(x,y+1) \geq V(x-1,y+1), \quad \forall x \geq 1.$$

This relation holds for $x = 1$ since $r_i > 0$. For $x \geq 2$, using the dynamic programming recursion, this relation is written as:

$$\begin{aligned} &\max_{i=1,\ldots,m} [p_i(r_i + V(x-1,y)) + (1-p_i)V(x,y)] \\ &\qquad \geq \max_{i=1,\ldots,m} [p_i(r_i + V(x-2,y)) + (1-p_i)V(x-1,y)]. \end{aligned}$$

By the induction hypothesis (i.e., $V(x,y) \geq V(x-1,y)$ for all x), each of the terms on the left-hand side is no less than the corresponding term on the right-hand side, and so the above relation holds.

Part c). The optimal rate is the one that maximizes in the dynamic programming, or equivalently the maximizer of

$$p_i r_i - p_i(V(x,y-1) - V(x-1,y-1)).$$

The highest rate r_m simultaneously maximizes $p_i r_i$ and minimizes p_i. Since $V(x-1,y-1) \leq V(x,y-1) \leq 0$ (as proved above), we see that the highest rate maximizes the above relation.

Solution to Problem 3.11

Part a). We define the state space $S = \{0, 1, 2, \ldots\}$ and at each state s, we have $A_s = A = \{0, 1\}$, where 0 indicates to keep the machine, whereas 1 corresponds to replacement. Transition probabilities are:

$$p(j|s,a) = \begin{cases} P_{sj} & a = 0, \\ 1 & a = 1, j = 0, \\ 0 & j \neq 0. \end{cases} \tag{4}$$

Costs (the opposite of rewards) $r(s, 0) = C(s)$ and $r(s, 1) = C(s) + R$.

The value function takes the form:

$$V(i) = C(i) + \min\Big\{\lambda \sum_{j=0}^{\infty} P_{ij} V(j), R + \lambda V(0)\Big\}, \quad i \in S.$$

Part b). Let $V_n(i)$ be the value function at the n-th iteration of the VI algorithm, with $V_0(i) = C(i)$. We first prove by induction that $V_n(i)$ is an increasing function in i for all $n \geq 0$.

For $n = 0$, $V_0(i) = C(i)$ is increasing in i (per problem definition). Assume that $V_k(i)$ is an increasing function in i. Next, we show that $V_{k+1}(i)$, expressed below, is an increasing function in i:

$$V_{k+1}(i) = C(i) + \min\Big\{R + \lambda V_k(0), \lambda \sum_{j=0}^{\infty} P_{ij} V_k(j)\Big\}.$$

Observe that $\sum_{j=0}^{\infty} P_{ij} V_k(j) = \mathbb{E}[V_k(j)|i] = \mathbb{E}[V_k(T_i)]$. Since $V_k(i)$ is increasing in i, by the stochastic ordering property, $\mathbb{E}[V_k(T_i)]$ is increasing in i, too. Hence, for $\lambda \in (0, 1)$, $V_{k+1}(i)$ is the sum of increasing function $C(i)$, and the minimum of a constant and an increasing function. Hence, it is increasing and thus, $V_n(i)$ is increasing in i for all $n \geq 0$. Finally, since $V(i) = \lim_{n\to\infty} V_n(i)$, we deduce that $V(i)$ is increasing in i.

Part c). We would like to show that there exists i_{th} such that $\pi^\star(i) = \mathbb{1}\{i \geq i_{\text{th}}\}$. Let k denote the earliest time that replacement happens $k = \min\{i \in S : \pi^\star(i) = 1\}$. We have that:

$$R + \lambda V(0) \leq \lambda \sum_{j=0}^{\infty} P_{kj} = \lambda \mathbb{E}[V(T_k)].$$

Since $V(\cdot)$ is increasing, we have for all $\ell > k$ that:

$$\begin{aligned} R + \lambda V(0) &\leq \lambda \mathbb{E}[V(T_k)] \\ &\leq \lambda \mathbb{E}[V(T_\ell)] \quad \text{(by stochastic ordering property)} \\ &= \lambda \sum_{j=0}^{\infty} P_{\ell j} V(j). \end{aligned}$$

Hence, $\pi^\star(\ell) = 1$.

References

[1] D. Bertsekas, *Dynamic programming and optimal control*, vol. 1. Athena Scientific, 1995.

[2] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

[3] M. S. Santos and J. Rust, "Convergence properties of policy iteration," *SIAM Journal on Control and Optimization*, vol. 42, no. 6, pp. 2094–2115, 2004.

[4] M. L. Littman, T. L. Dean, and L. P. Kaelbling, "On the complexity of solving markov decision problems," in *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pp. 394–402, Morgan Kaufmann Publishers Inc., 1995.