Assignment 4

Lilian Kourti
CME 241

Problem 1

We perform two iterations of Value Iteration for the given MDP assuming γ=1. It is underlined that no actions are taken and no rewards are gained from s_3, thus $v(s_3)$ and $q(s_3)$ remain zero and are omitted from the following calculations

- k=1:

$$\begin{aligned}
q_1(s_1, a_1) &= R(s_1, a_1) + P(s_1, a_1, s_1)v_0(s_1) + P(s_1, a_1, s_2)v_0(s_2) + P(s_1, a_1, s_3)v_0(s_3) \\
&= 8.0 + 0.2 * 10.0 + 0.6 * 1.0 + 0.2 * 0.0 \ = \ 10.6 \\
q_1(s_1, a_2) &= 10.0 + 0.1 * 10.0 + 0.2 * 1.0 \ = \ 11.2 \\
v_1(s_1) &= max(q_1(s_1, a_1), q_1(s_1, a_2)) \ = \ 11.2 \\
\pi_1(s_1) &= a_2
\end{aligned}$$

$$\begin{aligned}
q_1(s_2, a_1) &= R(s_2, a_1) + P(s_2, a_1, s_1)v_0(s_1) + P(s_2, a_1, s_2)v_0(s_2) + P(s_2, a_1, s_3)v_0(s_3) \\
&= 1.0 + 0.3 * 10.0 + 0.3 * 1.0 + 0.4 * 0.0 \ = \ 4.3 \\
q_1(s_2, a_2) &= -1.0 + 0.5 * 10.0 + 0.3 * 1.0 \ = \ 4.3 \\
v_1(s_2) &= max(q_1(s_2, a_1), q_1(s_2, a_2)) \ = \ 4.3 \\
\pi_1(s_2) &= a_1
\end{aligned}$$

- k=2:

$$\begin{aligned}
q_2(s_1, a_1) &= R(s_1, a_1) + P(s_1, a_1, s_1)v_1(s_1) + P(s_1, a_1, s_2)v_1(s_2) + P(s_1, a_1, s_3)v_1(s_3) \\
&= 8.0 + 0.2 * 11.2 + 0.6 * 4.3 + 0.2 * 0.0 \ = \ 12.82 \\
q_2(s_1, a_2) &= 10.0 + 0.1 * 11.2 + 0.2 * 4.3 \ = \ 11.98 \\
v_2(s_1) &= max(q_2(s_1, a_1), q_2(s_1, a_2)) \ = \ 12.82 \\
\pi_2(s_1) &= a_1
\end{aligned}$$

$$\begin{aligned}
q_2(s_2, a_1) &= R(s_2, a_1) + P(s_2, a_1, s_1)v_1(s_1) + P(s_2, a_1, s_2)v_1(s_2) + P(s_2, a_1, s_3)v_1(s_3) \\
&= 1.0 + 0.3 * 11.2 + 0.3 * 4.3 + 0.4 * 0.0 \ = \ 5.65 \\
q_2(s_2, a_2) &= -1.0 + 0.5 * 11.2 + 0.3 * 4.3 \ = \ 5.89 \\
v_2(s_2) &= max(q_2(s_2, a_1), q_2(s_2, a_2)) \ = \ 5.89 \\
\pi_2(s_2) &= a_2
\end{aligned}$$

If we were to continue choosing the same policy through the Value iteration for $k > 2$, i.e. if $\pi_k()$ is the same as $\pi_2()$, this would require:

$$q_k(s_1, a_1) - q_k(s_1, a_2) > 0,\ q_k(s_2, a_2) - q_k(s_2, a_1) > 0$$

Let's check that:

$$\begin{aligned} q_k(s_1, a_1) - q_k(s_1, a_2) &= 8 + 0.2v_{k-1}(s_1) + 0.6v_{k-1}(s_2) - 10 - 0.1v_{k-1}(s_1) - 0.2v_{k-1}(s_2) \\ &= -2 + 0.1v_{k-1}(s_1) + 0.4v_{k-1}(s_2) \\ &\geq -2 + 0.1v_2(s_1) + 0.4v_2(s_2) \\ &\geq -2 + 0.1 * 12.82 + 0.4 * 5.89 \ = \ 1.63 \ > \ 0 \end{aligned}$$

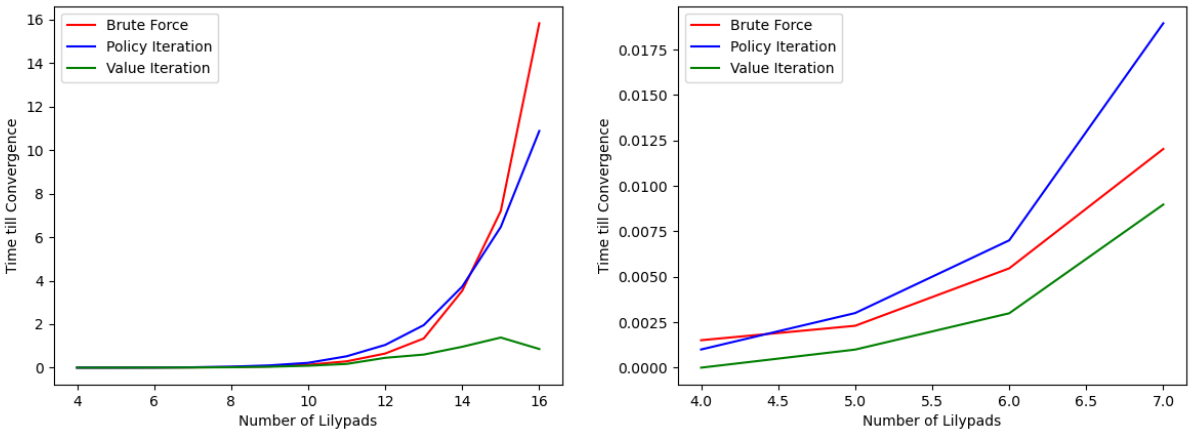
Similarly:

$$\begin{aligned} q_k(s_2, a_2) - q_k(s_2, a_1) &= -1 + 0.5v_{k-1}(s_1) + 0.3v_{k-1}(s_2) - 1 - 0.3v_{k-1}(s_1) - 0.3v_{k-1}(s_2) \\ &= -2 + 0.2v_{k-1}(s_1) \\ &\geq -2 + 0.2v_2(s_1) \\ &\geq -2 + 0.2 * 12.82 \ = \ 2.56 \ > \ 0 \end{aligned}$$

Hence, we conclude that: $\pi_k(s_1) = a_1,\ \pi_k(s_2) = a_2\ \forall k > 2$.

Problem 2

In the following plot, we compare the Brute Force approach (implemented in Assignment 3), the Policy Iteration and the Value Iteration by juxtaposing the time until convergence of each the three algorithms as a function of the number of lilypads:



As expected, Value Iteration outperforms the other two. In general, Value iteration is faster than Policy Iteration for Finite MDPs because at each iteration it only performs a partial Policy Evaluation. Additionally, we observe that Brute Force becomes slow exponentially fast, which is reasonable since it explicitly compares 2^{n-1} different policies to find the optimal, while the other two algorithms don't have such a deterministic dependence that grows exponentially to the number of states.

Problem 3

At the beginning of any day, the worker can be:

- employed (E) with job i. In this case, there's no action to be taken, i.e. the current job i is accepted (A). Or
- unemployed (U) with an offer for job i. In this case, the job offer can either be accepted (A) or declined (D)

Therefore, we observe that the worker's state is depended on the job index (i), but also the job status (U or E). Also, the actions are depended on the state of the worker. So we formulate the problem as follows:

- State Space: $\mathcal{S} = \{(i,k)|1 \leq i \leq n,\ k \in \{E,U\}\}$
- Action Space: $\mathcal{A}((i,E)) = \{A\},\ \mathcal{A}((i,U)) = \{A,D\},\ 1 \leq i \leq n$
- Transition Function for $1 \leq i \leq n,\ k \in \{E,U\}$:

$$P[(i',k')|(i,k),A] = \begin{cases} 1-\alpha, & i'=i,\ k'=E \\ \alpha p_{i'}, & 1 \leq i' \leq n,\ k'=U \\ 0, & o.w. \end{cases}$$

$$P[(i',k')|(i,U),D] = \begin{cases} p_{i'}, & 1 \leq i' \leq n,\ k'=U \\ 0, & o.w. \end{cases}$$

- Reward Function for $1 \leq i \leq n,\ k \in \{E,U\}$:

$$R((i,k),A) = U(w_i)$$

$$R((i,U),D) = U(w_0)$$

- Bellman Optimality Equation for $1 \leq i \leq n$:

$$V^*((i,U)) = max\left\{U(w_0) + \gamma \sum_{i'=1}^{n} p_{i'} V^*((i',U)),\ V^*(i,E)\right\}$$

$$V^*(i,E) = U(w_i) + \gamma\left(\alpha \sum_{i'=1}^{n} p_{i'} V^*((i',U)) + (1-\alpha)V^*(i,E)\right)$$

See RL-book/_lkourti/Assign4/prob_4.3.py for the implementation and the calculation of the Optimal Value Function and the Optimal Policy.