Artificial Neural Networks (Gerstner). Solutions for week 3

Markov Decision Processes

Exercise 1. Optimal policies for finite horizon.

Create a Markov Decision Process where the optimal horizon-T policy depends on the time step, i.e. there is at least one state s and one pair of timesteps t and t' such that $\pi^{(t)}(a|s) \neq \pi^{(t')}(a|s)$.

Hint: You can choose $T = 2$ for simplicity.

Solution:

Consider the simple MDP in Figure 1, where we have three states s1, s2, and s3. There are 2 actions available at s1 and s2: Action a1 takes the agent from both states s1 and s2 to state s3, through which the agent recieves a deterministic reward of +2. Action a2 takes the agent from state s1 to s2 and from s2 to s1, while the agent recieves a deterministic reward of +1 through both transitions. State s3 is a terminal state with a dummy action a1 that keeps agents at state s3 (without any reward).

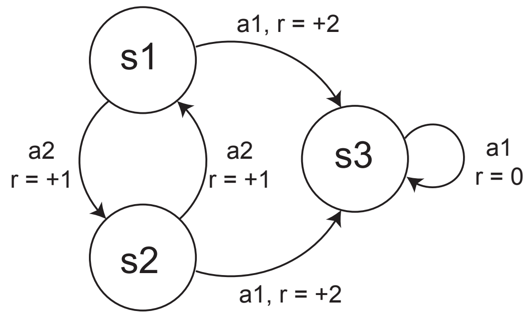

Figure 1: MDP of Exercise 1

For $T = 2$, it is easy to see that the optimal policy is given by

$$\pi^{(1)}(a|\text{s1}) = \delta_{a,\text{a2}} \quad \text{and} \quad \pi^{(1)}(a|\text{s2}) = \delta_{a,\text{a2}}$$
$$\pi^{(2)}(a|\text{s1}) = \delta_{a,\text{a1}} \quad \text{and} \quad \pi^{(2)}(a|\text{s2}) = \delta_{a,\text{a1}}.$$

Exercise 2. Shortest path search.

Let $\mathcal{S} = \{s_1, s_2, s_3, \dots\}$ denote a set of vertices (think of cities on a map) and let the vertices be connected by some edges $e_{s_i,s_j} \in (0, \infty]$ (think of distances between cities), where $e_{s_i,s_j} = \infty$ indicates that there is no direct connection between s_i and s_j. Dijkstra's algorithm for finding the shortest paths to some goal vertex g can be written in the following way (we show the lenght of the shortest path from vertex s to g by $V(s)$):

- For each vertex $s \in \mathcal{S}$, initialize all distances from g by $V(s) \leftarrow \infty$.
- Initialize the distance of g from itself by $V(g) \leftarrow 0$.
- Define and initialize $\tilde{\mathcal{S}} \leftarrow \mathcal{S}$.
- While $\tilde{\mathcal{S}}$ is not empty
  - $s_i \leftarrow \arg\min_{s\in\tilde{\mathcal{S}}} V(s)$
  - Remove s_i from $\tilde{\mathcal{S}}$
  - For each neighbor s_j of s_i still in $\tilde{\mathcal{S}}$: $V(s_j) \leftarrow \min(V(s_j), V(s_i) + e_{s_i,s_j})$.
- Return $V(s)$ for all $s \in \mathcal{S}$.

The output $V(s)$ of Dijkstra's algorithm is equal to the lenght of the shortest path from s to g. In this exercise, we formulate the problem of finding the shortest path as a dynamic programming problem.

a. What is the equivalent Markov Decision Process for the problem of finding the shortest paths to some goal state?
Hint: Define the goal state as an absorbing state and describe the properties of r_s^a and $p^a_{s_i \to s_j}$.

b. Compare the value iteration algorithm on the MDP of part a with Dijkstra's algorithm.

Solution:

a. We consider a deterministic MDP with the state space $\mathcal{S}$ and the following properties:

   (i) $\gamma = 1$.
   (ii) Available actions in each state $s \in \mathcal{S}$ are moving to one of the neighbouring states.
   (iii) The reward corresponding to moving from $s \in \mathcal{S}$ to $s' \in \mathcal{S}$ is equal to $-e_{s,s'}$.
   (iv) The goal vertix $g \in \mathcal{S}$ is the only terminal state.

   Since all rewards are negative, the optimal policy in this MDP is to get to the terminal state $g \in \mathcal{S}$ with largest cumilative reward which is equivalent to shortest distance. Hence, the negative optimal value $-V^*(s)$ is equal to the shortest distance from vertix s to the source g.

b. Dijkstra's algorithm is similar to value iteration, but it has some fundamental differences:

   (i) In Dijkstra's algorithm, the set of states whose values are updated in each iteration decreases by one after each iteration (s_i is removed from $\tilde{\mathcal{S}}$).
   (ii) In Dijkstra's algorithm, the arg max over all possible next actions is removed and replaced by a comparison between the current value of the state ($V(s_j)$) and the value of the action that takes the agent to s_i (i.e., $V(s_i) + e_{s_i,s_j}$).
   (iii) Dijkstra's algorithm uses the fact that transitions are deterministic and replace the averaging over next state s' in the value update directly by the value of the next state.

Exercise 3. Bellman operator.

Proof that the Bellman operator is a contraction.

Hint: Show the contraction with the infinity norm, i.e.

$$\|T_\gamma[X] - T_\gamma[Y]\|_\infty = \max_s |T_\gamma[X]_s - T_\gamma[Y]_s| \leq \gamma \|X - Y\|_\infty,$$

where the last inequality is to be proven. You can use the notation $Q^X_{sa} = r^a_s + \gamma \sum_{s' \in \mathcal{S}} p^a_{s \to s'} X_{s'}$ and the facts that $|\max_a Q^X_{sa} - \max_{a'} Q^Y_{sa'}| \leq \max_a |Q^X_{sa} - Q^Y_{sa}|$ and $\sum_{s' \in \mathcal{S}} p^a_{s \to s'} = 1$.

Solution:

We start with replacing the Bellman operators in the hint by their explicit definitions

$$\|T_\gamma[X] - T_\gamma[Y]\|_\infty = \max_s |T_\gamma[X]_s - T_\gamma[Y]_s| = \max_s \left| \max_a Q^X_{sa} - \max_{a'} Q^Y_{sa'} \right|.$$

We can now use the fact $|\max_a Q^X_{sa} - \max_{a'} Q^Y_{sa'}| \leq \max_a |Q^X_{sa} - Q^Y_{sa}|$ as well as the definition of Q^X_{sa} and write

$$\begin{aligned}
\|T_\gamma[X] - T_\gamma[Y]\|_\infty &\leq \max_s \max_a \left| Q^X_{sa} - Q^Y_{sa} \right| \\
&= \max_s \max_a \left| \left(r^a_s + \gamma \sum_{s' \in \mathcal{S}} p^a_{s \to s'} X_{s'} \right) - \left(r^a_s + \gamma \sum_{s' \in \mathcal{S}} p^a_{s \to s'} Y_{s'} \right) \right| \\
&= \max_s \max_a \left| \gamma \sum_{s' \in \mathcal{S}} p^a_{s \to s'} (X_{s'} - Y_{s'}) \right| \leq \gamma \max_s \max_a \sum_{s' \in \mathcal{S}} p^a_{s \to s'} |X_{s'} - Y_{s'}|,
\end{aligned}$$

where, for the last inequality, we used the fact that $|\sum_{s'} Z_{s'}| \leq \sum_{s'} |Z_{s'}|$ for any vector Z. In addition, we have

$$|X_{s'} - Y_{s'}| \leq \max_{s'} |X_{s'} - Y_{s'}| = \|X - Y\|_\infty .$$

Combining the last two inequalities, we have

$$\begin{aligned}\|T_\gamma[X] - T_\gamma[Y]\|_\infty &\leq \gamma \max_s \max_a \sum_{s' \in \mathcal{S}} p^a_{s \to s'} \|X - Y\|_\infty \\ &\leq \gamma \|X - Y\|_\infty \max_s \max_a \sum_{s' \in \mathcal{S}} p^a_{s \to s'},\end{aligned}$$

and, because $\sum_{s' \in \mathcal{S}} p^a_{s \to s'} = 1$, we have

$$\|T_\gamma[X] - T_\gamma[Y]\|_\infty \leq \gamma \|X - Y\|_\infty .$$

If $\gamma < 1$, then the last inequality implies that the operator T_γ is a contraction mapping.

Exercise 4. Coding exercise: Value and policy iteration.

Implement value and policy iteration in python to solve the MDP from Example 1 from the lecture.