Solution for Assignment 2, Markov Decision Processes
IFT6135-H2018

Shagun Sodhani (McGill ID: 260844972, UdeM ID:20111009),
Vardaan Pahuja (McGill ID: 260844959, UdeM ID:20081093)

February 6, 2018

1 Mandatory Questions

1.1 Bellman Optimality Equations

The finite horizon optimality equations may be expressed as

$$v_{n+1}(s) = sup_{a \in A_s}(r(s,a) + \sum_{j \in S} \lambda p(j|s,a)v_n(j))$$

For an optimal policy (in the limit):

$$v(s) = sup_{a \in A_s}(r(s,a) + \sum_{j \in S} \lambda p(j|s,a)v(j))$$

The above system of equations is called Bellman Equations.
In order to write the Bellman optimality equations in vectorized form, we define non-linear operator $\mathcal{L}$ and linear operator L on V as

$$\mathcal{L}v = sup_{d \in D^{MD}} \{r_d + \lambda P_d v\}$$
$$Lv = max_{d \in D^{MD}} \{r_d + \lambda P_d v\}$$

where the notation used is defined as below:
d: policies
D_{MD}: policies which are Markovian and Deterministic
v: vector of value function (each entry in column vector for a particular state)
λ: discount factor
r_d: reward function (each entry in column vector for a particular state)
P_d: transition probability matrix The solution to Bellman optimality equations can be expressed as:

$$v = \mathcal{L}v$$

To show that $\mathcal{L}$ is a contraction mapping
Assumption: The set of states S is discrete, $\mathcal{L}$ maps $V \to V$. An operator $T : U \to U$ is a contraction mapping if $\exists$ λ, $0 \leq \lambda < 1$ such that

$$\|Tv - Tu\| \leq \lambda \|v - u\|$$

$\forall$ u and v in U.
Let $u, v \in V$, fix $s \in S$, let $Lv(s) \geq Lu(s)$ and let

$$a_s^* = argmax_{a \in A_s} \left\{ r(s,a) + \sum_{j \in s} \lambda p(j|s,a)v(j) \right\}$$

$$
\begin{aligned}
0 &\leq Lv(s) - Lu(s) \\
&\leq r(s, a_s^*) + \sum_{j \in S} \lambda p(j|s, a_s^*)v(j) \\
&- r(s, a_s^*) - \sum_{j \in S} \lambda p(j|s, a_s^*)u(j) \\
&= \lambda \sum_{j \in S} p(j|s, a_s^*)[v(j) - u(j)] \\
&\leq \lambda \sum_{j \in S} p(j|s, a_s^*) \left\| v - u \right\|_\infty \\
&= \lambda \left\| v - u \right\|_\infty
\end{aligned}
$$

$$
\begin{aligned}
\implies Lv(s) - Lu(s) &\leq \lambda \left\| v - u \right\|_\infty \\
\text{Repeating this argument in the case that } Lu(s) &\geq Lv(s) \\
Lu(s) - Lv(s) &\leq \lambda \left\| v - u \right\|_\infty \\
\implies |Lv(s) - Lu(s)| &\leq \lambda \left\| v - u \right\|_\infty \\
\text{Taking sup. on both sides,} \\
sup_s \{|Lv(s) - Lu(s)|\} &\leq sup_s \lambda \left\| v - u \right\|_\infty \\
\left\| \mathcal{L}v - \mathcal{L}u \right\|_\infty &\leq \lambda \left\| v - u \right\|_\infty \\
\implies \mathcal{L} \text{ is a contraction mapping}
\end{aligned}
$$

$$
\begin{aligned}
0 &\leq \left\| \mathcal{L}v^* - v^* \right\|_\infty \\
&\leq \left\| \mathcal{L}v^* - v^n \right\|_\infty + \left\| v^n - v^* \right\|_\infty \\
&= \left\| \mathcal{L}v^* - \mathcal{L}v^{n-1} \right\|_\infty + \left\| v^n - v^* \right\|_\infty \\
&\leq \lambda \left\| v^* - v^{n-1} \right\|_\infty + \left\| v^n - v^* \right\|_\infty
\end{aligned}
$$

$lim_{n \to \infty} \left\| v^n - v^* \right\| = 0$, both quantities on RHS can be made arbitrarily small by choosing n large enough.

$$
\implies \mathcal{L}v^* = v^*
$$

Thus, the sequence of v^n obtained by applying the operator $\mathcal{L}$ converges, and there exists a solution to the Bellman optimality equations

1.2 Policy Iteration

Policy Iteration algorithm has two steps

1. Policy Evaluation
2. Policy Improvement

Let us denote our policy at time n as μ_n and the state value as v_n.

In **Policy Evaluation** step, we have

$v_n(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)(r + \gamma v_n(s'))$

We can rewrite this equation in the matrix form as

$$
v_n = r_{\mu_n} + \gamma P_{\mu_n} v_n \tag{1}
$$

where r_{μ_n} and P_{μ_n} are terms that depend on the current policy μ_n.

r_{μ_n} corresponds to the immediate rewards and P_{μ_n} corresponds to the transition matrix.

In **Policy Improvement** step, we have a new policy μ_{n+1} (policy at time $n+1$) such that

$r_{\mu_{n+1}} + \gamma P_{\mu_{n+1}} v_n \geq r_{\mu_n} + \gamma P_{\mu_n} v_n$

$\Rightarrow r_{\mu_{n+1}} + \gamma P_{\mu_{n+1}} v_n \geq v_n$ (Using equation 1)

Rearranging the terms, we get

$$r_{\mu_{n+1}} \geq (I - \gamma P_{\mu_{n+1}}) v_n \tag{2}$$

Now, when we run the policy evaluation step, we get

$v_{n+1} = r_{\mu_{n+1}} + \gamma P_{\mu_{n+1}} v_{n+1}$

Rearranging the terms, we get

$(I - \gamma P_{\mu_{n+1}}) v_{n+1} = r_{\mu_{n+1}}$.

Substituting value from (2), we get

$(I - \gamma P_{\mu_{n+1}}) v_{n+1} \geq (I - \gamma P_{\mu_{n+1}}) v_n$

$\Rightarrow v_{n+1} \geq v_n$

This implies that the state values are always either better or as good as the previous state values.
We would terminate the Policy Iteration algorithm when $\mu_{n+1} = \mu_n$. (Please note that in practice, we may have to check that if the difference between the two policies is less than a given threshold) Now given a finite MDP, with $|S|$ number of states and $|A|$ number of actions, we can not have more than $|A|^{|S|}$ number of policies. Each policy is basically a choice of what action to take in a given state. Given the finite MDP, we can choose $|A|$ actions in the first state, then $|A|$ actions in the second state and so on. The total number of possible choices comes out to be $(|A| * |A| \cdots)|S|$ times and is equal to $|A|^{|S|}$. In general, not all the actions may be possible in all the states so the actual number of possible policies could be less than $|A|^{|S|}$ but it can not be more than $|A|^{|S|}$. This argument proves that policy iteration must terminate under a finite number of steps.

Given all the possible policies, we can group the policies into bins $\Pi_1, \Pi_2, \cdots, \Pi_N$ such that

$\forall i \in \{1, 2, \cdots N\}$, all policies in Π_i are equal and
$\forall i, j \in \{1, 2, \cdots N\}$, all policies in Π_j are better than all policies in Π_i if $j > i$.
Here N is the total number of bins that can be formed and is bounded by $|A|^{|S|}$ (as we can not have more policies than the bins themselves).

This way, policy iteration guarantees that after each step, we move from one bin i to another j such that $j > i$. Once we reach the N^{th} bin, we terminate. Let us assume that at termination, policy iteration has converged to Π. We will prove by contradiction that Π is the optimal policy. Let us assume that the optimal policy is not Π but Π^* such that $\Pi^* \neq \Pi$.

If this is the case, there is at least one state s such that
$v_{\Pi^*}(s) > v_{\Pi}(s)$
$\Rightarrow (I - \gamma P_{\Pi^*})v_{\Pi^*} > (I - \gamma P_{\Pi^*})v_{\Pi}$
$\Rightarrow r_{\Pi^*} > (I - \gamma P_{\Pi^*})v_{\Pi}$
$\Rightarrow r_{\Pi^*} + \gamma P_{\Pi^*} v_{\Pi} > r_{\Pi} + \gamma P_{\Pi} v_{\Pi}$

This implies that the policy iteration algorithm would not have terminated at policy Π. This contradicts our assumption that the policy iteration algorithm terminated at a suboptimal policy. Thus the policy Π is indeed the optimal policy.

2 Track-2

The policy iteration can be written in recursive form as

$$Bv = max_{d \in D}(r_d + (\lambda P_d - I)v)$$

where the operator $B : V \to V$
Thus, $Bv = Lv - v$ where L is defined as

$$Lv = max_{d \in D^{MD}} \{r_d + \lambda P_d v\}$$

The Bellman optimality equation can the be expressed as

$$Bv = 0$$

For $u \in V$, let D_v denote the set of policy improvement rules. That is, $d_v \in D_v$ if

$$d_v \in argmax_{d \in D} \{r_d + (\lambda P_d - I)v\}$$

$$Bv = r_{d_v} + (\lambda P_{d_v} - I)v$$

The derivative of Bv w.r.t. v can be written as

$$\frac{d(Bv)}{dv} = (\lambda P_{d_v} - I)$$

$$v^{n+1} = r_{d_{v^n}} + \lambda P_{d_{v^n}} v^{n+1}$$

Re-arranging terms, we get,

$$v^{n+1} = (I - \lambda P_{d_{v^n}})^{-1} r_{d_{v^n}} - v^n + v^n$$

$$= v^n - (\lambda P_{d_{v^n}} - I)^{-1}[r_{d_{v^n}} + (\lambda P_{d_{v^n}} - I)v^n]$$

$$= v^n - \left[\frac{d(Bv)}{dv}\right]^{-1}(Bv)$$

This resembles Newton-Kantorovich iteration approach (in vector form) for finding the zero of the equation $Bv = 0$

The geometric intuition behind the observation is the following:

Consider a linear operator G defined from $V \to V$ such that $Gv = r + (\lambda P - I)v$ for each $v \in V$ and Bv is the maximum value of the operator for each v. Note that this way Bv defines the upper envelope of the G operator and is a convex function. Then, the policy iteration process corresponds to finding the zero of this convex function Bv. Hence the resulting equation looks very similar to Newton method.