MS&E338 Reinforcement Learning Lecture 7 - April 27, 2020

Real-Time Value Iteration

Lecturer: Ben Van Roy *Scribe: Cindy Lu, Mohammadhossein Shafinia, Viraga Perera*

1 Example: Markov Decision Processes

Let us consider a simple Markov Decision Process (MDP) defined by,

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \widetilde{\rho}, \widetilde{P}, \widetilde{r}) \tag{1}$$

which models the behavior of an environment defined by,

$$\mathcal{E} = (\mathcal{O}, \mathcal{A}, \mathcal{H}, \rho) \tag{2}$$

Within this model, we assume that the termination is memory-less; each episode has a termination probability of $(1 - \gamma)$. Furthermore, we assume that all actions are deterministic.

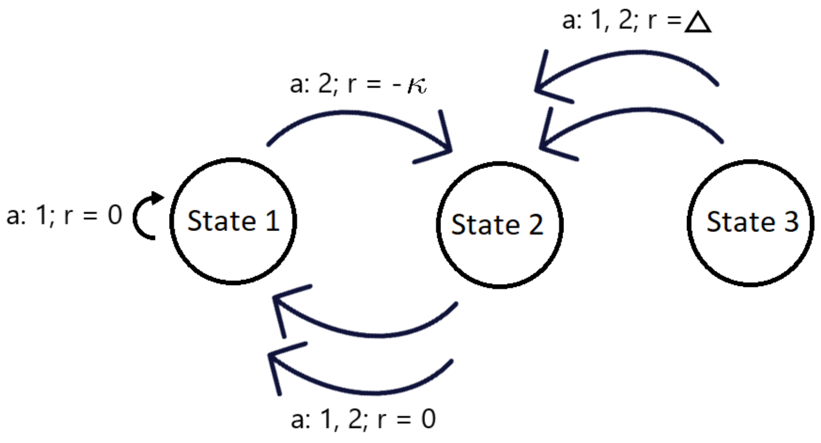

Figure 1: Example MDP

Here, the MDP has three states,

$$\mathcal{S} = \{1, 2, 3\} \tag{3}$$

In each state, it has two possible actions,

$$\mathcal{A} = \{1, 2\} \tag{4}$$

For simplicity, let us assume that each episode starts at State 3,

$$\widetilde{\rho}(3) = 1 \tag{5}$$

Each row of the transition matrix $\widetilde{P}$ sums to γ. Since there is only one non-zero entry within each row, that entry has a value of γ. The rewards related to the MDP, $\widetilde{r}$, can be observed within Figure 1.

Within this setup, the optimal strategy appears to be to start at State 3, transition to State 2, transition to State 1, and remain there until termination; this accrues a total of Δ rewards.

Now, let's consider a further compression of histories of the this Markov decision process such that there are only two states (as opposed to three); we define the resulting two agent states as,

$$\mathcal{S} = \{\alpha, \beta\} \tag{6}$$

where, α is defined as the same as State 1 in the example above, whereas β is defined as a combination of State 2 and State 3.

Let us assign relevance weights to the old states (with relation to the new states) as follows:

$$\nu(1) = 1 \tag{7}$$

$$\nu(2) = 1 - \epsilon \tag{8}$$

$$\nu(3) = \epsilon \tag{9}$$

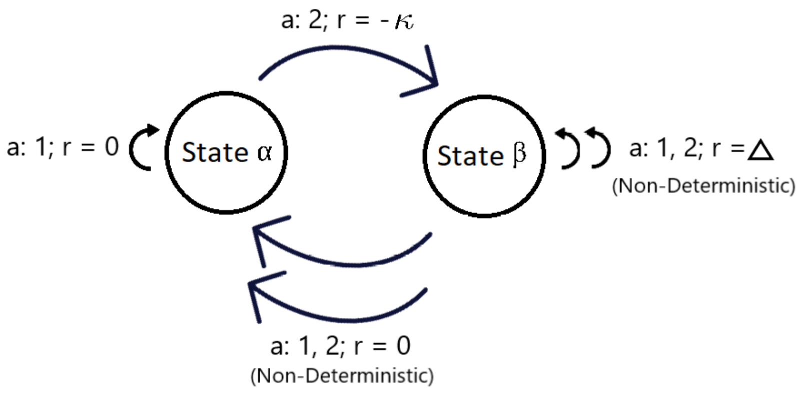

Figure 2: Further Compressed Example MDP

Here, State β behaves like State 2 with probability ϵ and like State 3 with probability $1 - \epsilon$. So, our actions are no longer deterministic.

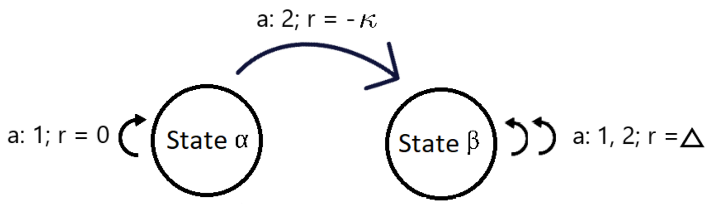

Figure 3: Further Compressed Example MDP with $\epsilon = 0$

Now let's consider an extreme case in which $\epsilon = 0$.

In this case,

$$\widetilde{\mathcal{Q}_*}(\beta, a) = (1/(1-\gamma))\Delta \tag{10}$$

(for a = 1, 2)

$$\widetilde{\mathcal{Q}_*}(\alpha, 2) = -\kappa + (\gamma/(1-\gamma))\Delta \tag{11}$$

Here, we are going to assume that the quantity on the right side of the equation is > 0; we do this by imposing the following requirement on κ:

$$\kappa < (\gamma/(1-\gamma))\Delta \tag{12}$$

Finally,

$$\widetilde{\mathcal{Q}_*}(\alpha, 1) = \gamma \max_{a \in \mathcal{A}} \widetilde{\mathcal{Q}_*}(\alpha, a) \tag{13}$$

Here, if the maximum is achieved by $a = 1$, then,

$$\widetilde{\mathcal{Q}_*}(\alpha, 1) = 0 \tag{14}$$

However, we know that when a = 2, $\widetilde{\mathcal{Q}_*} > 0$. So, the maximum must be achieved when $a = 2$. In this case,

$$\widetilde{\mathcal{Q}_*}(\alpha, 1) = \gamma(-\kappa + (\gamma/(1-\gamma))\Delta) \tag{15}$$

In this case, the agent is incentivized to go back to State 2, after reaching State 1, causing a looping between States 1 and 2 while incurring a cost of $-\kappa$ with each loop. This policy contrasts with that for the the original MDP that was discussed earlier, in which the optimal policy for the agent was shown to be to remain in State 1 after reaching it. This deviation from the original result is an error stemming from the agent's belief that States 2 and 3 are one state (State β), where State 3 is significantly more representative of the aggregate state than State 2. So, the agent (falsely) believes that it will get a reward Δ with each visit to State β, whereas, in reality, it ends up incurring a cost of $-\kappa$ with every visit.

It turns out to be the case that the policy discussed here satisfies the following worst-case condition:

$$\widetilde{\mathcal{V}_*} - \widetilde{\mathcal{V}}_\pi = 2\tilde{\tau}^2\Delta \tag{16}$$

In order to overcome these short-comings, we must utilize a method to assign more appropriate relevance weights (ν) to the states that are being aggregated.

2 The Real-Time Value Iteration Algorithm

In this next section, we explore one algorithm where we can induce the desired relevance weights. Otherwise, we cannot ensure that the policies generated by the algorithm have better performance than those generated by other approximations where the state sampling frequencies are not incorporated into such relevance weights. The main idea is to design a planning algorithm that plans by interacting with a simulator of the environment $\mathcal{E}$.
Given a simulator for the environment $\mathcal{E}$, we seek to come up with a good policy for that environment by allowing the agent to interact with this simulator. The interaction is similar to the process that takes place in reinforcement learning, as illustrated by Figure 4.

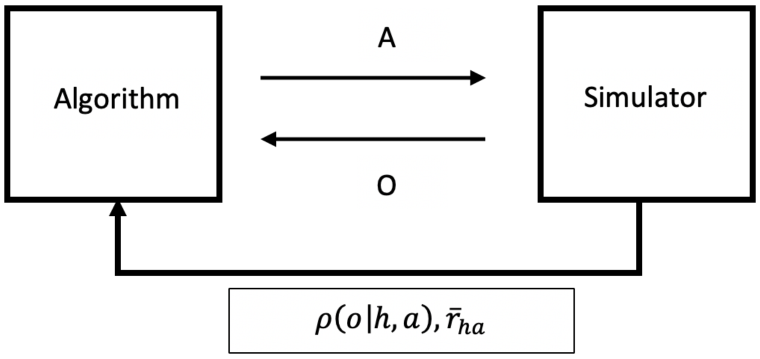

Figure 4: The Real-Time Valuation Iteration Algorithm (RTVI)

Under RTVI, the algorithm takes an action A and gets back an observation O from the simulator, over all episodes. The algorithm is not quite an RL algorithm, because it looks under the hood of the simulator and extracts the probabilities of observations given histories and actions, $\rho(o|h,a)$, and expected rewards $\bar{r}_{ha}$. We make two assumptions for this algorithm, which will be abstracted away later.

Assumption 1. *The compression function f is sufficient, i.e. Q^* values are constant over any cell of the partition of $\mathcal{H}$.*

Assumption 2. *The expected termination time is memoryless, i.e. there exists a deterministic scalar $\gamma \in [0,1)$ such that for all $t \in \mathbb{Q}_+, h \in \mathcal{H}$ of duration t, $\mathbf{P}(\tau = t+1|\mathcal{E}, H_t = h) = 1 - \gamma$.*

Algorithm 1 shows how the states are sampled by an agent from the simulator that at all times, the actions are decided on in a greedy manner based on the current action-value function. After executing the action selected and observing a new observation, we update the action-value function by taking the sum with the immediate expected rewards and the expected action-value $\tilde{Q}$ at what the next agent-state would be if maximized over actions.

Algorithm 1: Real-Time Value Iteration (RTVI)

```
initialize $\tilde{Q}(s,a) = M$ with $M$ large, $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$;
for $\ell = 1, 2, ...$ do
    observe $O$;
    $H \leftarrow (O), S \leftarrow f(O)$;
    while $S$ not terminated do
        $A \leftarrow \arg\max_{a \in \mathcal{A}} \tilde{Q}(s,a)$;
        execute $A$, observe $O$;
        $\tilde{Q}(S,A) \leftarrow \bar{r}_{ha} + \sum_{o \in \mathcal{O}} \rho(o|h,a) \max_{a' \in \mathcal{A}} \tilde{Q}(f(S,A,O),a')$;
        $H \leftarrow (H,A,O), S \leftarrow f(S,A,O)$
    end
end
```

Intuitively, we can imagine that in this algorithm, the agent interacting with the environment has an internal action-value function $\tilde{Q}$, which says how good the state-action pairs are, at each time step. It selects actions that are greedy with respect to $\tilde{Q}$. The update rule is based on a thought experiment of what is the range of possibilities that can occur as a consequence of executing this action a at this history h. These possibilities

are weighted by observation probabilities. At each time step, the agent only updates the value associated with an agent-state action pair. Note that the right hand side of the update rule in Algorithm 1 depends on the current history H. So for different H that yields the same agent-state S, the updated $\tilde{Q}$ value can be different.
The algorithm converges to an optimal policy. Later, it can be shown that the algorithm converges quickly. If we further relax the assumption that f is sufficient, then we can further generalize the real-time value iteration algorithm to accommodate for compression functions that allow Q values to differ across histories in the same partition. If an algorithm of this flavor converges, then it should converge in a manner where the episodes that its going through are sampling state-action pairs according to the policy it is converging to. These sampling frequencies are include relevance weights, resulting in better policies than if we made approximations that do not pay attention to relevance weights related to state frequencies.
It follows from our assumption that f is sufficient that there exists $\tilde{Q}_*$ such that for all $h \in \mathcal{H}, a \in \mathcal{A}$,

$$\tilde{Q}_*(f(h), a) = Q_*(h, a)$$

In other words, since f is a sufficient compression, we can always come up with such a function $\tilde{Q}$ that matches Q_*.

Next we further study the updating step in the RTVI algorithm. Letting $s = f(h)$, the updating operation can be written as

$$\tilde{Q}(s, a) = (G_h \tilde{Q})(s, a) \tag{17}$$

where the operator G_h is defined as

$$G_h \tilde{Q}(s, a) = \begin{cases} \overline{r}_{ha} + \sum_{o \in \mathcal{O}} \rho(o|h, a) \max_{a' \in \mathcal{A}} \tilde{Q}(f(s, a, o), a') & \text{if } s = s' \\ Q(s, a) & \text{otherwise} \end{cases}$$

In equation 17, note that on the right hand side, we have an operator that reminds us a lot of a Bellman operator, except it depends on the current history. It should be clear that G_h is monotonic, and further, $\tilde{Q}_*(s, a) = (G_h \tilde{Q})(s, a)$ for all $h \in \mathcal{H}, a \in \mathcal{A}$ such that $s = f(h)$. We now show that using this updating rule, the algorithm converges, and it converges to an optimal policy.

Proof: Initialize $\tilde{Q}(s, a) = M = \frac{1}{1-\gamma} \max \overline{r}_{ah}\ \forall (s, a)$, then we have:

$$\begin{aligned} (G_h \tilde{Q})(s, a) &= \overline{r}_{ah} + \sum_{o \in \mathcal{O}} \rho(o|h, a) \max_{a' \in \mathcal{A}} \tilde{Q}(f(s, a, o), a') \\ &\leq (1-\gamma)M + \gamma M \\ &= M \\ &= \tilde{Q}(S, a) \end{aligned} \tag{18}$$

So if we start with the defined initial value and apply the update we can only decrease the value. And since G_h is monotonic it always get smaller at each step. Thus the sequence of functions $\tilde{Q}$ is non-increasing. Also $\tilde{Q}_*$ is a fix point for G_h so again by using monotonicity of G_h we have $\tilde{Q} > \tilde{Q}_*$. These observations implies that $\tilde{Q}$ converges. (There is an issue with this part of the proof. Please see Lecture 9 for a fix.)

As it is converging there is a limit for it. Let $\tilde{Q}_\dagger$ be the limit of convergence and let $\tilde{\pi}_\dagger$ be greedy policy with respect to it. It remains to show that $\tilde{Q}_\dagger$ is indeed the optimal policy. $\forall (h, a)$ visited by $\tilde{\pi}_\dagger$ and $s = f(h)$, as $G_h \tilde{Q}_\dagger$ is the sequence converging limit, we have:

$$(G_h \tilde{Q}_\dagger)(s, a) = \tilde{Q}_\dagger(s, a) \tag{19}$$

This implies:

$$\begin{aligned}\tilde{Q}_\dagger(s,a) &= \bar{r}_{ah} + \sum_{o\in\mathcal{O}} \rho(o|h,a) \max_{a'\in\mathcal{A}} \tilde{Q}_\dagger(f(s,a,o),a') \\ &= \bar{r}_{ah} + \sum_{o\in\mathcal{O}} \rho(o|h,a) \sum_{a'\in\mathcal{A}} \tilde{\pi}_{\dagger h}(a')\tilde{Q}_\dagger(f(s,a,o),a')\end{aligned} \tag{20}$$

Let $\tilde{Q}_\dagger(h,a) = \tilde{Q}_\dagger(f(h),a)$

$$\tilde{Q}_\dagger(h,a) = \bar{r}_{ah} + \sum_{o\in\mathcal{O}} \rho(o|h,a) \sum_{a'\in\mathcal{A}} \tilde{\pi}_{\dagger h}(a')\tilde{Q}_\dagger(f(h,a,o),a') = (F_{\tilde{\pi}_\dagger}\tilde{Q}_\dagger)(h,a) \tag{21}$$

Thus we have $\tilde{Q}_\dagger(h,a) = Q_{\tilde{\pi}_\dagger}(h,a)$ $\forall(h,a)$ visited by $\tilde{\pi}_\dagger$. We know $\tilde{Q}_\dagger(h,a) \leq Q_*(h,a)$. Thus:

$$Q_*(h,a) \leq \tilde{Q}_\dagger(h,a) \leq Q_*(h,a) \tag{22}$$

We have $Q_*(h,a) = \tilde{Q}_\dagger(h,a)$ $\forall(h,a)$ visited by $\tilde{\pi}_\dagger$. Also we have $Q_*(h,a) \leq \tilde{Q}_\dagger(h,a)$ $\forall(h,a)$. The last thing to show is that the greedy policy with respect to $Q_\dagger$ has to be optimal. Consider a history $h \in \mathcal{H}$ which was visited by $\tilde{\pi}_\dagger$. Suppose $\tilde{\pi}_\dagger$ selects $a \in A$ then $Q_*(h,a) = Q_\dagger(h,a) = \max_{a'\in\mathcal{A}} Q_\dagger(h,a') \geq \max_{a'\in\mathcal{A}} Q_*(h,a')$ which shows a is optimal and so $\tilde{\pi}_\dagger$. This concludes the proof.