Q-Learning with a Multi-state Markov Decision Process *

Jiarong Jiang, Hal Daumé III
{*jiarong,hal*}*@umiacs.umd.edu*
University of Maryland, College Park

In standard Markov decision process (MDP)[Bel57] setting, an MDP consists of a (typically finite) *state space* S, an *action space* A, a (possibly stochastic) *transition function* T, and a (possibly stochastic) *reward function* R. An agent observes the current state $s \in S$ and chooses an action $a \in A$. The environment responds by moving the agent to another state $s' \in S$, which is sampled from the MDP's transition distribution $T(s' \mid s, a)$, and by giving the agent a reward r, which is sampled from the MDP's reward distribution $R(r \mid s, a)$ (For simplicity, here we assume reward function only depends on the resulting state and the action taken). The agent's goal is to maximize its total reward over time. An agent's *policy* π describes how the agent chooses an action based on its current state.

However, in real world applications, there are cases which are computationally easier to allow the agent occupy multiple states at the same time and solve a MDP based on sets of states. More specifically, denote the standard MDP $M = (S, A, T, R)$.

Define the **multi-state MDP** based M as $M^+ = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}\}$ where

1. The state in M^+: $\tilde{s} \in \mathcal{S} = 2^S$ which is a set of states in M.
2. The action $\tilde{a} = (s, a) \in \mathcal{A}, s \in S, a \in A(s)$. In M^+, the action is defined as choosing a state $s \in S$ in M and one of its corresponding actions $a \in A(s)$.
3. The transition distribution $\mathcal{T}(\tilde{s}' \mid \tilde{s}, \tilde{a}) = T(\tilde{s}' \backslash \tilde{s} | s, a)$ where $\tilde{a} = (s, a)$.
4. The reward function $\mathcal{R}(\tilde{r} \mid \tilde{s}, \tilde{a}) = R(r \mid s, a)$ for $\tilde{a} = (s, a)$.

Compared to M, M^+ works as follows: An agent observes the current state $\tilde{s} \in \mathcal{S}$ where $\tilde{s} \subseteq S$. It choose a substate $s \in \tilde{s}$ and take action $a \in A(s)$. The environment responds by a new state s' through choosing a from s in M and expanding the agent to the state $\tilde{s}' = \tilde{s} \cup s'$, which is sample from $\mathcal{T}$. The environment also gives the agent a reward $\tilde{r}$ which is sample from $\tilde{R}$. The agent's goal is still to maximize its total reward over the time.

Take a chart parser as an example: all the actions and rewards are deterministic and all the states are non-revisitable. For M, $s \in S$ is a current partial tree and $a \in A(s)$ to choose one of the possible following constituents to build on top of the partial tree. The accuracy is evaluated when there is no more constituent to add (or a parse tree is found). For M^+, now the state is all the existing partial trees, and the action is to choose one of the partial tree and add a constituent on top of it. This is exactly what agenda-based parser does. For the convenience, we add backpointers for each $s \in \tilde{s}$ to remember the expansion of the states for the final reward. If we assume the cost for each expansion is positive, the final goal is to learn to trade off speed and accuracy.

In this work, we use Q-learning[Wat89] to solve this MDP problem and want to show that there exists a decomposition of the value such that

*Topic: Reinforcement Learning

1. The Q-learning algorithm gives $\tilde{Q}(\tilde{s}, \tilde{a})$ converges on M^+ to $\tilde{Q}^*(\tilde{s}, \tilde{a})$.
   (This is analogous to the proof in [JJS94]).

2. If we run Q-learning algorithm on both M and M^+ with some given initial state, $\tilde{Q}^*(\tilde{s}, \tilde{a}) = Q^*(s, a)$ if $\tilde{a} = (s, a)$ and s, a are on the path from the intimal state to the final state in the optimal policy and $\tilde{Q}^*(\tilde{s}, \tilde{a}) \leq Q^*(s, a)$ otherwise.

3. The Q-learning algorithm on M^+ converges faster (or equal) than the Q-learning algorithm on M.

References

[Bel57] R. Bellman. A markovian decision process. In *Journal of Mathematics and Mechanics 6*, 1957.

[JJS94] Tommi Jaakkola, Michael I. Jordan, and Satinder P. Singh. Convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6:1185–1201, 1994.

[Wat89] C.J.C.H Watkins. Learning from delayed rewards. In *Cambridge University*, 1989.