Reinforcement Learning Tutorial 7, Week 8

—

Reward expection, Prioritisation, and Uncertainty*

Michael Herrmann

March 2024

Problem 1 – R-learning

R-learning[1] is similar to $\mathcal{Q}$-learning, in particular for non-discounted, non-episodic problems. It is based on the average reward $\rho = \lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} E\left[r_t\right]$, and considers the current rewards in comparison to this accumulating reward averge towards the value:

$$V\left(s_t\right) = \sum_{k=1}^{\infty} E\left[r_{t+k} - \rho | s_t = s\right]$$

$$\mathcal{Q}\left(s_t, a_t\right) = \sum_{k=1}^{\infty} E\left[r_{t+k} - \rho | s_t = s, a_t = a\right]$$

In this relative value function (relative to the average), ρ is slowly adapted as a measure of success. In this way a different concept of optimality is implied in particular for non-episodic tasks. As an algorithm, R-learning works as follows

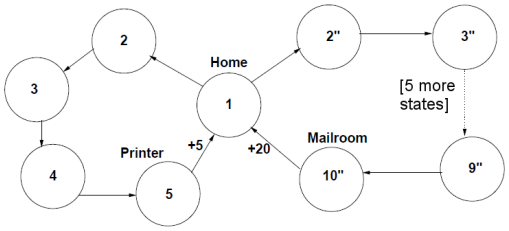
1. Initialise ρ and $\mathcal{Q}\left(s, a\right)$
2. Observe s_t and choose a_t (e.g. ε-greedy), execute a_t
3. Observe r_{t+1} and s_{t+1}
4. Update
$$\mathcal{Q}_{t+1}(s_t, a_t) = (1 - \eta)\, \mathcal{Q}_t(s_t, a_t) + \eta\Big(r_{t+1} - \rho_t + \max_a \mathcal{Q}_t(s_{t+1}, a)\Big)$$
5. If $\mathcal{Q}\left(s_t, a_t\right) = \max_a \mathcal{Q}\left(s_t, a\right)$ then
$$\rho_{t+1} = (1-\alpha)\, \rho_t + \alpha\Big(r_{t+1} + \max_a \mathcal{Q}_t(s_{t+1}, a) - \max_a \mathcal{Q}_{t+1}(s_t, a)\Big)$$

[*We would choose $\eta \gg \alpha$, because, otherwise, for $r = 0$, $\mathcal{Q}$-value may cease to change and the agent may get trapped in a suboptimal limit cycle.*]

*with special thanks to Adam Jelley

[1] A. Schwartz (1993) A reinforcement learning method for maximizing undiscounted rewards. 10th ICML. (You don't need to know R-learning for the exam.)

Compare R-learning and Q-learning in the following simple example, where only one decision needs to be taken: The agent moves either to nearby printer ("o.k.") or to distant mail room ("good").



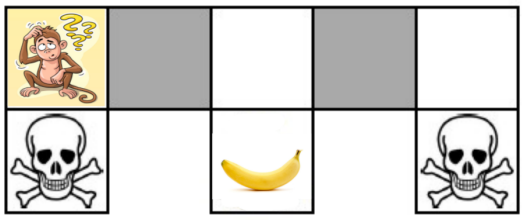
Is it possible that R learning finds the optimal solution quicker than Q-learning? How does the result for Q-learning depend on the discount factor γ?

Problem 2 – Prioritised sweeping[2]

While Dyna[3] agents select state-action pairs uniformly at random from the previously experienced pairs, it might be more efficient to use a non-uniform probability distribution. Why? Which state-action pairs should be preferred? Discuss the role of a goal states in this context.

Problem 3 – Ambiguous state information

Discuss the aliased gridworld example[4], where the agent, which is here, as always, a monkey, cannot distinguish between the two densely overgrown swampy parts of the environment (shown here as grey grid cells). This means that for the two states that are shown in grey the same entry of the policy $\pi(\cdot, \text{grey})$ has to be used. Actions are: N, W, S, E. Rewards and states as shown in the figure, where we naturally assume that $r(\textit{skull}) \ll r(\textit{banana})$.



Compare the optimal deterministic policy for the example with the optimal stochastic policy. How could an algorithm find the stochastic policy?

[2] Sutton & Barto, Sect. 8.4
[3] The Dyna-Q algorithm is one example of Dyna, see Lecture 7, Slides 7ff
[4] adapted from David Silver's lecture 7