Reinforcement learning
Derivation from Bellman Equation

Bert Kappen

Reinforcement learning

We consider a first order Markov process that assigns a probability to the transition of x to x' under action u: $p_0(x'|x,u)$.

Reinforcement learning considers an infinite time horizon and rewards are discounted.

We introduce a reward that depends on our current state and action $R(x,u)$.

We define a *policy* $\pi(u|x)$ as the conditional probability to take action u given that we are in state x. Given the policy π and given that we start in state x_0, the probability to be in state x_t at time $t > 0$ is given by

$$p_\pi(x_t|x_0;t) \quad = \sum_{u_{0:t-1},x_{1:t-1}} \prod_{s=0}^{t-1} p_0(x_{s+1}|x_s,u_x)\pi(u_s|x_s)$$

The *expected future discounted reward* in state x is defined as:

$$J_\pi(x) = \sum_{s=0}^{\infty} \sum_{x',u'} \pi(u'|x') p_\pi(x'|x;s) R(x',u') \gamma^s \qquad p_\pi(x'|x,0) = \delta_{x,x'}$$

with $0 < \gamma < 1$ the discount factor. J_π is also known as the value function for policy π. The objective of reinforcement learning is to find the policy π that maximizes J for all states.

We can write a recursive relation for J_π in the same way as we did in the previous section.

$$\begin{aligned} J_\pi(x) &= \sum_u \pi(u|x) R(x,u) + \sum_{s=1}^{\infty} \sum_{x',u'} \pi(u'|x') p_\pi(x'|x;s) R(x',u') \gamma^s \\ &= \sum_u \pi(u|x) \left(R(x,u) + \gamma \sum_{x'} p_0(x'|x,u) J_\pi(x') \right) \end{aligned}$$

Solving for $J_\pi(x)$ by fixed point iteration is called *policy evaluation*.

The idea of policy improvement is to construct a better policy from the value of the previous policy. Once we have computed J_π, we construct a new deterministic policy

$$\pi'(u|x) = \delta_{u,u(x)}, \qquad u(x) = \arg\max_u R(x,u) + \gamma \sum_{x'} p_0(x'|x,u) J_\pi(x') \quad (1)$$

It can be shown that the solution for $J_{\pi'}$ is as least as good as the solution J_π in the sense that

$$J_{\pi'}(x) \geq J_\pi(x), \forall x$$

Thus

$$\pi^0 \rightarrow J_{\pi^0} \rightarrow \pi^1 \rightarrow J_{\pi^1} \rightarrow \pi^2 \ldots$$

One can show, that this procedure converges to a fixed point $J^*(x)$.

TD learning and actor-critic networks

The above procedures assume that the environment $p_0(x'|x,u), R(x,u)$ in which the automaton lives is known.

When the environment is not known one can either first learn a model and then a controller or use a so-called model free approach, which yields the well-known TD(λ) and Q-learning algorithms.

When p_0 and R are not known, one can replace the Bellman equation by a sampling variant

$$J_\pi(x) = J_\pi(x) + \alpha(r + \gamma J_\pi(x') - J_\pi(x)). \tag{2}$$

with x the current state of the agent, x' the new state after choosing action u from $\pi(u|x)$ and r the actual observed reward.

To verify that this stochastic update equation gives a solution, look at its fixed point:

$$J_\pi(x) = R(x,u) + \gamma J_\pi(x').$$

and take expectation wrt to $\pi(u|x)$ and $p_0(x'|x,u)$.

Eq. 2 is the TD(0) algorithm. In principle, one should require full convergence of the TD algorithm under the policy π before a new policy is defined.

Actor-critic idea: Interleave Eq. 2 with policy changes Eq. 1.

Q learning

A mathematically more elegant way to compute the optimal policy in a model free way is given by the Q learning algorithm (Watkins). Denote $Q(x,u)$ the optimal expected value of state x when taking action u and then proceeding optimally. That is

$$Q(x,u) = R(x,u) + \gamma \sum_{x'} p_0(x'|x,u) \max_{u'} Q(x',u') \tag{3}$$

and $J^*(x) = \max_u Q(x,u)$.

Its stochastic, on-line, version is

$$Q(x,u) = Q(x,u) + \alpha(R(x,u) + \gamma \max_{u'} Q(x',u') - Q(x,u)) \tag{4}$$

As before, one can easily verify that by taking the expectation value of this equation with respect to $p_0(x'|x,u)$ one recovers Eq. 3.

Note, that for this approach to work not only all states should be visited a sufficient number of times (as in the TD approach) but all state-action pairs. On the other hand, Q-learning does not require the policy improvement step and the repeated

computation of value functions. Also in the Q-learning approach it is tempting to limit actions to those that are expected to be most successful, as in the TD approach, but this may again result in a suboptimal solution.