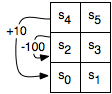
Tiny reinforcement learning problem

Yu Jiang 6/22/2017

This demo shows how to use rltool to solve a simple reinforcement learning problem.

In the following tiny grid, there are 6 states, s0, s1, s2, …, s5. At each state, four actions can be taken, they are up, down, left, and right. At state s4, if action “left” is taken, the next state will be s0, and the reward of this movement is 10. At state s2, if action “left” is take, the next state will still be s2, with a reward of -100. Other movements will generate a reward of -1 at each time. If there are no states along the direction of the movement, the next state will be the same state.

Apparently, to maximize the reward, the optimal policy should drive the state quickly converging to the loop of s0 -> (up) -> s2 -> (up) -> s4 -> (left) -> s0…



To solve the problem with rltool, we first create the state grid and the action grid

>> stategrid = rlstate(0:5);

>> actiongrid = actiongrid(1:4) % 1 – up, 2 – right , 3 – left, 4 – down.

Then, create the Q table

>> qt = qtable(stategrid, actiongrid);

We can set the three parameters, Epsilon, LearningRate, and DiscountingFactor. They are properties of qt.

Also, we need to initialize the initial state, which can be done randomly.

>> s = 3;

Next, we can repeat the following:

>> a = qt.getAction(s);

>> [snew, r] = env.step(a); % env is an object describing the environment. It takes an action, and produce a new state and the corresponding reward.

>> qt.update(s, a, snew, r); % Key equation for q table update

>> s = snew; % save a copy of the state for next step use.

Run demo\_tiny\_rl.m to see the animation.