Reinforcement Learning—HW 8 Solution

April 6, 2020

1. *Exercise 8.1* The nonplanning method looks particularly poor in Figure 8.3 because it is a one-step method; a method using multi-step bootstrapping would do better. Do you think one of the multi-step bootstrapping methods from Chapter 7 could do as well as the Dyna method? Explain why or why not.

   Solution:

   A multi-step bootstrapping method should do better than a 1-step method (should update values for state/actions far from the end state, just as Dyna method does). However, it's unlikely to do as well as the Dyna method because the Dyna method does many more updates of value functions. For an episode of length k, it'll do only k updates of values, whereas the Dyna method will do kn updates if doing n planning steps.

2. *Exercise 8.2* Why did the Dyna agent with exploration bonus, Dyna-Q+, perform better in the first phase as well as in the second phase of the blocking and shortcut experiments?

   Solution: The Dyna-Q+ agent performed better in the first phase because it did more exploration. Presumably, the Dyna-Q agent found *some* non-optimal path to the goal, and could rely only on the ϵ-greedy exploration to improve the path. The Dyna-Q+ agent did further exploration due to the exploration bonus, and found path improvement more quickly.

3. *Exercise 8.5* How might the tabular Dyna-Q algorithm shown on page 164 be modified to handle stochastic environments? How might this modification perform poorly on changing environments such as considered in this section? How could the algorithm be modified to handle stochastic environments and changing environments?

   Solution: To handle stochastic environments, the algorithm could keep associated with a state/action pair a reward/next state pair, along with a count. Then, when the model needs a next state/reward, it could sample from those reward/next state pairs weighted by the counts.

   This could work poorly in a changing environment because the old reward/next state pairs could weigh down the changing probabilities. One way to deal with that would be to weight more recent reward/next state pairs more heavily, perhaps by doing an exponential decay of the counts.