Exercise Set 1 - Reinforcement Learning

MDPs and Dynamic Programming

Giulio Starace - 13010840

September 13, 2022

Homework: Coding Assignment - Dynamic Programming

1. Coding answers have been submitted on codegra under the group "stalwart cocky sawfly".
2. Comparing policy iteration and value iteration algorithms:
   - I expect a single iteration of value iteration to run faster than a single iteration of policy iteration. This is because an iteration of policy iteration involves policy evaluation, which itself is an iterative process requiring several sweeps across the set of states. On the other hand, an iteration in value iteration "truncates" policy evaluation by stopping after a single sweep.
   - I expect policy iteration to take fewer iterations than value iteration. This is because **TODO**

Homework: Dynamic Programming

1. The value $v^\pi(s)$ of a state s under policy π is simply the expected action-value function at s over all actions a under pi. It can hence be expressed in terms of π and $q^\pi(s,a)$ as:

$$\begin{aligned} v^\pi(s) &= \mathbb{E}_\pi \left[q^\pi(S_t, A_t) | S_t = s, A_t = a \right] \\ v^\pi(s) &= \sum_a \pi(s|a) q^\pi(s,a) \end{aligned} \tag{1}$$

   in the stochastic case and

$$v^\pi(s) = q^\pi(s, \pi(s)) \tag{2}$$

   in the deterministic case.

2. The value iteration update can be re-written in terms of Q-values as such:

$$q_{k+1}(s,a) = \sum_{s',r} p(s', r|s, a) \left[r + \gamma \max_{a'} q_k(s', a') \right], \tag{3}$$

   obtaining the Q-value iteration update. This is done by turning the Bellman optimality equation for $q_*(s,a)$ into an update rule, as is done for the value iteration update.

3. A new policy evaluation update in terms of $Q^\pi(s, a)$ instead of $V^\pi(s)$ can be obtained by writing the following, making use of equation (1):

$$Q^\pi(s, a) \leftarrow \sum_{s', r} p(s', r|s, a) \left[r + \gamma \sum_{a'} \pi(a|s) Q^\pi(s', a') \right] \tag{4}$$

4. hello world.

5. On page 74 (it's actually on page 75), policy evaluation involves a summation over actions, with the policy taking each action being multiplied by the sum over states and rewards. On page 80, the summation over actions is not present, and so we compute the action in the probability term by using the policy previously computed in the policy improvement step. The reason for this is **todo**