# Exercise Set 1 - Reinforcement Learning

MDPs and Dynamic Programming

Giulio Starace - 13010840

September 14, 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 total iterations than value iteration. This is because value iteration relies on the convergence to an optimal *value function*, despite ultimately being interested in the optimal *policy*. Since the convergence criterion is set to some arbitrarily small $\Delta$, and we only extract the optimal policy from the optimal value function after convergence, it can occur that the optimal policy is found before convergence. In policy iteration on the other hand, since we improve our policy at each iteration, we obtain a new policy at each iteration, which we can evaluate in the next iteration before checking for stability (and hence convergence). This potentially allows for more directed improvements since we are directly focusing on the policy rather than the value function. Furthermore this is a more frequent way of checking for *policy* optimality by directly evaluating policies. The more frequent and directness of the convergence check intuitively suggests faster convergence than value iteration.

## Homework: Dynamic Programming

1. The value $v^\pi(s)$ of a state $s$ under policy $\pi$ is simply the expected action-value function at $s$ over all actions $a$ under $pi$. It can hence be expressed in terms of $\pi$ 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. The policy improvement step of the policy iteration algorithm can be rewritten in terms of $Q^{\pi}(s, a)$ instead of $V^{\pi}(s)$ as follows:

$$\pi(s) \leftarrow \arg\max_{a} Q^{\pi'}(s, a), \tag{5}$$

where $\pi'$ is the old policy we are improving.

5. On page 75, policy evaluation involves a weighted summation over actions stochastically outputted by the policy multiplied by the sum over transition-weighted state rewards and state-values. On page 80, the summation over actions is not present, and we compute the action in the probability term deterministically by using the policy previously computed in the policy improvement step.

   We can make this distinction precisely because in one case we apply the policy stochastically (therefore requiring an expectation over actions, and hence summation) while in the other case we apply it deterministically. The usage of the policy in the policy evaluation step of policy iteration is deterministic because we have fixed the policy after greedily obtaining it via policy improvement and are now evaluating this policy specifically.