Chapter 03 – Policy Iteration

Author: Gianmarco Scarano

[*gianmarcoscarano@gmail.com*](mailto:gianmarcoscarano@gmail.com)

1. Policy Iteration

Policy Iteration outputs policies at every iteration: $\{\pi_0, \pi_1, \pi_2, \dots, \pi_T\}$, which is different from Value Iteration that was outputting numerical values. The general concept is that we start from a random policy, compute the value and then we'll take policy improvement steps.

Procedure:

1. We start from a random guess π_0 (deterministic or stochastic)
2. For $t = 0, \dots, T$:
    a. Do *policy evaluation* and compute $Q^{\pi t}$ for all s, a. (Of course, let's remember that $Q^{\pi}(s_t, a)$ is the Q-Function which choose the best action in the current state and then follows policy π starting from next state s'). Let's forget about approximation errors, so we want to use the Exact Policy Iteration.
    b. Do *policy improvement* as $\pi_{t+1} = argmax_a Q^{\pi t}(s, a)$ for all s.

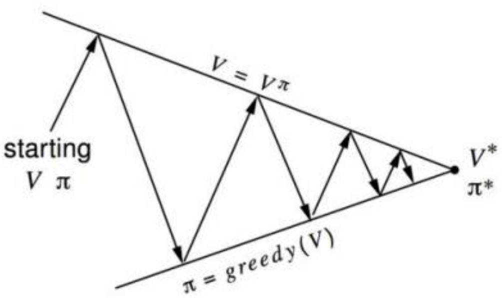
This algorithm only makes progress since the performance of the policy is monotonic.

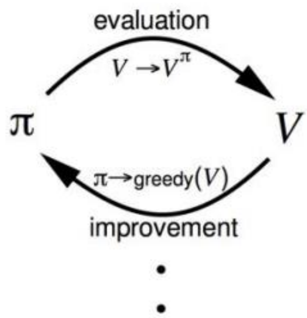
In fact, this is a property of this algorithm.

- **Monotonic improvement:** $Q^{\pi t+1} \geq Q^{\pi t}$ *for all* s, a
- **Convergence:** $\|V^{\pi i} - V^*\| \leq \gamma^i \, \|V^{\pi 0} - V^*\|$ *(Same inequality as contraction mapping)*

If the policy stays the same, it means that we reached the optimal policy.

From this figure on the right, we can see that the difference between the V function and the policy, as we iterate, will get smaller and smaller until we reach a point where we find V^* and π^* (optimal Value function).



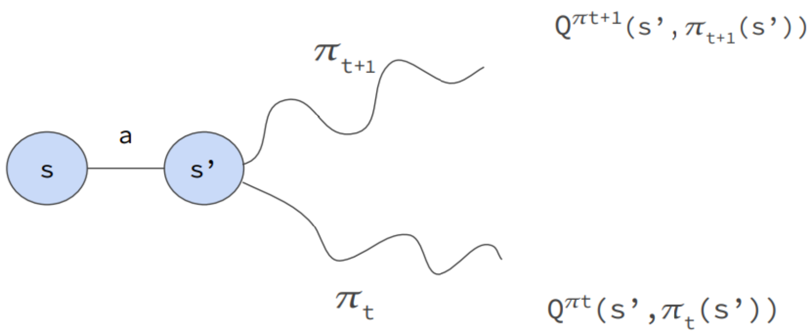


Now we'll go into detail about the monotonic improvement step:

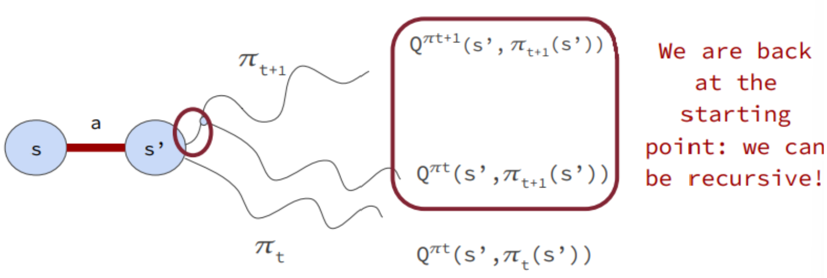
- **Demonstrate:** $Q^{\pi t+1} \geq Q^{\pi t}$ *for all* s, a.
- **Remember:** $\pi_{t+1} = argmax_a Q^{\pi t}(s, a)$ -> thanks to Optimal Policy

So, on the right image is explicated the initial situation. We have our initial state s and through action a we go to state s'. This step is the EXACT same for both $Q^{\pi t+1}$ and $Q^{\pi t}$.

Starting then from s', we follow two different policies at different time steps (t and $t+1$).

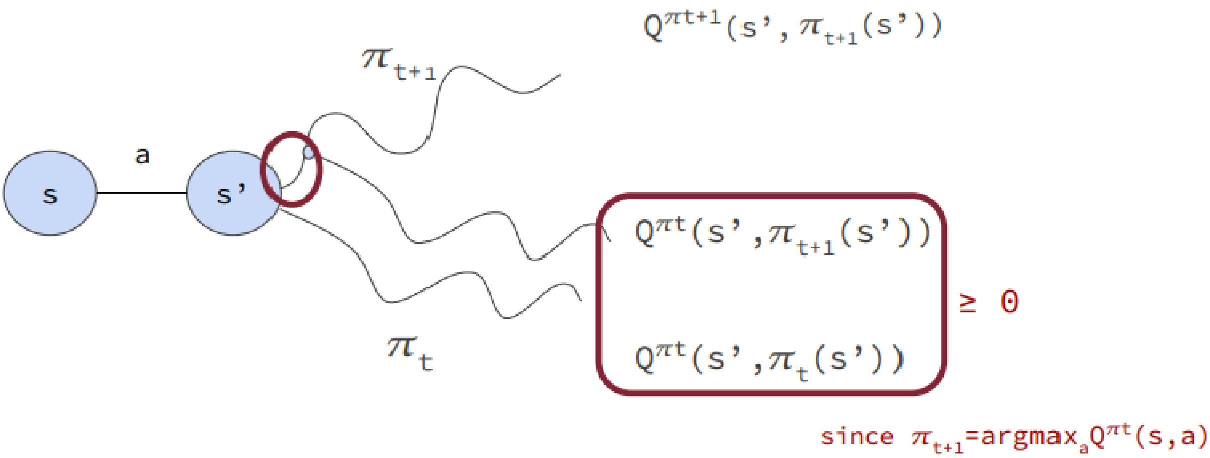


From here, we can definitely see how we can arrive to the next state after s' (little blue dot) and from there choosing to follow policy π_{t+1} and following the policy π_t (the one below).



From this, we can simply state that $Q^{\pi t+1} \geq Q^{\pi t}$ (also due to the 2nd bullet point **REMEMBER**).

We can better visualize it through this last plot below:



As for the proof of this, we want to look at the theorem by expanding the definitions of $Q^{\pi t+1}$ and $Q^{\pi t}$:

$$Q^{\pi^{t+1}}(s,a) - Q^{\pi^t}(s,a) = \gamma \mathbb{E}_{s' \sim P(s,a)} \left[Q^{\pi^{t+1}}(s', \pi^{t+1}(s')) - Q^{\pi^t}(s', \pi^t(s')) \right]$$

On the formula above, we simply expanded the definitions, brought out the Expectations and γ and remove the rewards. We pass from (s,a) to $(\pi^{t+1}(s'))$ due to the fact that that policy at timestep $t+1$ is the argmax of the action a over the Q-function at timestep t.

Now we can do a mathematical trick which adds and subtract a certain quantity (so in principle we are not doing anything).

add and subtract the Q of our intermediate policy

$$= \gamma \mathbb{E}_{s' \sim P(s,a)} \left[Q^{\pi^{t+1}}(s', \pi^{t+1}(s')) \underline{- Q^{\pi^t}(s', \pi^{t+1}(s')) + Q^{\pi^t}(s', \pi^{t+1}(s'))} - Q^{\pi^t}(s', \pi^t(s')) \right]$$

It's easy to demonstrate that the last two terms $(Q^{\pi^t}(s', \pi^{t+1}(s')) - Q^{\pi^t}(s', \pi^t(s'))$ is a quantity ≥ 0.

This is due to the last plot we have above. At the end, this is $\geq$ than the previous quantity and by iterating over and over we say that in the end it is $\geq -\gamma^{\infty}(1-\gamma)$ which is equal to 0.

The maximum number of iterations of policy iteration is $|A|^{|S|}$ since it's the maximum number of policies we could have. We can stop if the policy does not change anymore for any state.

1.1 Dynamic Programming

Problem	Bellman Equation	Algorithm
Prediction	Bellman Expectation Equation	Iterative Policy Evaluation
Control	Bellman Expectation Equation + Greedy Policy Improvement	Policy Iteration
Control	Bellman Optimality Equation	Value Iteration

We applied Dynamic Programming for **planning** as we assumed to know the MDP transition probabilities.