RL Assignment1 Report

520030910342 Jiyu Liu

1. Convergence of Policy Iteration

Please refer to the writing proof in page 2.

2. Grid World

2.1. Value Iteration

In this section, I have implemented a value iteration agent in class **ValueIterationAgent**.

Firstly, I define the function **computeQValueFromValues** by initializing the value to 0 and iterate through all possible states then computing Q-values from self.value.

Secondly, in the function **computeActionfromvalues**, I construct a **Counter** named Q_values and calculate the corresponding Q-value of every action, then I use the function **argMax** to return the action that brings best Q-value.

Finally, in the function **runValueIteration**, I perform a series of iterations of value updating process. In each iteration, the function updates the value for every non-terminal state to the maximum Q-value. Plus, if the maximum change among the states is less than self.epsilon, terminate the function early.

2.2. Policy Iteration

In the funtion **runPolicyIteration**, I perform a series of iterations of policy evaluation and policy improvement. In the policy evaluation process, the function computes the V-values with a fixed policy by iterating until they converge. In the policy improvement process, the function updates the policy to an action that makes highest Q-value with fixed values. Moreover, the policy evaluation step stops when the maximum change is less than self.epsilon and the policy iteration ends when the updated policy is the same as the original one.

2.3. Convergence Speed

I plot the V-value of each state in each iteration until convergence and save them to the Plot folder, which contains V-value images in value iteration and policy iteration respectively.

It indicates that policy iteration converges faster than value iteration (in terms of DiscountGrid, the value iteration needs 11 iterations to converge while the policy iteration only needs 7 iterations), the reason of which might be that policy function tends to converge faster than value function due to its simpler structure (the policy function can only choose from four directions in a certain state).

1.

Denote the original policy as π_i and the new policy derived from policy improvement step as π_{i+1}

1) $\pi_{i+1}(s) \in \arg\max_a Q^{\pi_i}(s,a)$

2) $Q^{\pi_i}(s, \pi_{i+1}(s)) \geq V^{\pi_i}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s') \mid \pi_i, s_0 = s\right] = \sum_a P(a|s) \cdot Q^{\pi}(s,a)$

3) $V^{\pi_i}(s) \leq Q^{\pi_i}(s, \pi_{i+1}(s)) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi_{i+1}, s_t = s\right]$

$= E[R(s_t, a_t, s_{t+1}) + \gamma V^{\pi_i}(s_{t+1}) \mid \pi_{i+1}, s_t = s]$

$\leq E[R(s_t, a_t, s_{t+1}) + \gamma Q^{\pi_i}[s_{t+1}, \pi^{i+1}(t+1))] \mid \pi_{i+1}, s_t = s]$

$= E[R(s_t, a_t, s_{t+1}) + \gamma E[R(s_{t+1}, a_{t+1}, s_{t+2}) + \gamma V^{\pi_i}(s_{t+2}) \mid \pi_{i+1}, s_{t+2} = s] \mid \pi_i, s_t = s]$

$= E[R(s_t, a_t, s_{t+1}) + \gamma R(s_{t+1}, a_{t+1}, s_{t+2}) + \gamma^2 V^{\pi_i}(s_{t+2}) \mid \pi_{i+1}, s_t = s]$

$\vdots$

$\leq E[R(s_t, a_t, s_{t+1}) + \gamma R(s_{t+1}, a_{t+1}, s_{t+2}) + \gamma^2 R(s_{t+2}, a_{t+2}, s_{t+3}) \cdots \mid \pi_{i+1}, s_t = s]$

$= E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi_{i+1}, s_0 = s\right]$

$= V^{\pi_{i+1}}(s)$

So a policy improvement step will always produce a new policy at least as good as the original one.

The equation holds if and only if $V^{\pi_i}(s_n) = Q^{\pi_i}(s_n, \pi_{i+1}(s_n))$ for every n that is larger than or equal to t, which means the policy is optimal. Therefore, the new policy is strictly better than the original one before it becomes optimal.

Plus, since there are only finite policies in a given MDP, policy iteration can always converges to an optimal policy.