EE5904 Part II

Project 1: Q-Learning for World Grid Navigation

Student Name: Xin Zhengfang

Student Number: A0206597U

Email Address: e0427425@u.nus.edu

Problem Statement

We need find optimal policy that makes robot traverse from initial state (0,0) to the final state (10,10) with maximum total reward on a 10x10 grid under giving reward function.

Q-learning

We use Q-learning to solve above problem. Q-function measures “worth” of state-action pair (s,a) in terms of rewards received when agent is executing a task. An optimal policy is one that maximizes (with respect to a given task) values of Q-function over all possible (s,a) pairs. To obtain this Q-function, we follow below steps:

1. **Input:** Discount factor γ; exploration probability ϵ_k; learning rate α_k
2. Initialize Q-function
3. Determine the initial state s_0
4. For time step k, select action α_k according to:

$$a_k = \begin{cases} a \in \arg\max_{\hat{a}} Q_k(s_k, \hat{a}) & \text{with probability } 1-\epsilon_k \\ \text{an action uniformly randomly} & \\ \text{selected from all other actions} & \\ \text{available at state } s_k & \text{with probability } \epsilon_k \end{cases}$$

5. Apply action a_k, receive reward r_{k+1}, then observe next state s_{k+1}
6. Update Q-function with:

$$Q_{k+1}(s_k, a_k) = Q_k(s_k, a_k) + \alpha_k(r_{k+1} + \gamma \max_{a'} Q_k(s_{k+1}, a') - Q_k(s_k, a_k))$$

7. Set k = k + 1 and repeat **for-loop** for the next time step
8. Break the loop if we back to an old state second time or reach the end state, and we count it as one trial
9. We will do 3000 trials in all tasks. The trial will early stop if Q-function has already converged

Task 1

Task 1 is to solve the problem we mentioned in Sec. Problem Statement. We will use the method illustrated in Sec. Q-learning with different setting. The reward function is given and specified in Task1.mat.

Moreover, the learning rate threshold is 0.005, tolerance error of Q-function is also 0.005.

In this task, ϵ_k and α_k are set to the same value. We will run 10 times of Q-learning procedure. Table 1 shows my result with specified setting.

Table 1. Parameters values and performance of Q-learning

$\epsilon_k = \alpha_k$	No. of goal-reached runs		Execution time (sec.)	
	$\gamma = 0.5$	$\gamma = 0.9$	$\gamma = 0.5$	$\gamma = 0.9$
$\frac{1}{k}$	0	0	N/A	N/A
$\frac{100}{100+k}$	0	9	N/A	2.7663
$\frac{1+\log(k)}{k}$	0	0	N/A	N/A
$\frac{1+5\log(k)}{k}$	0	3	N/A	3.3929

Maximum reward is 1861.9157. And the optimal policy is visualized in Figure 1.

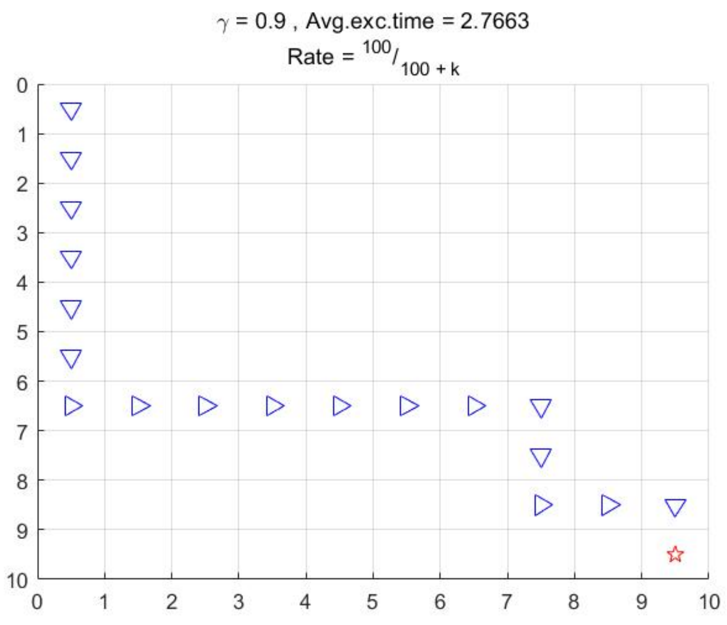

Figure 1. The visualization of optimal policy in task 1.

Comments

1. The result may be not reproducible because of exploration. Therefore, we need fix random seed. My report shows the result when rng(1) is used. The execution time will fluctuate every time because of computer status.
2. During the experiment, if the tolerance error becomes harsh, then the number of goal-reached runs may increase, but the execution time will spend more time on convergence. To balance efficiency and goal reached, we use tolerance error as 0.05.
3. As shown in the Table 1, there are no successful runs when discount rate equals to 0.5. Because the reward value drops so fast that there is no difference in each direction.
4. The learning rate is also significant. If the learning rate decreases so fast (such as $\frac{1}{k}$ and $\frac{1+\log(k)}{k}$), the knowledge learned is limited and it will explore more frequently.

In conclusion, we should use a stable learning rate and exploration rate that do not change so much or drop so quickly, and the discount rate is should be a little large like 0.9 or try some values that are larger than 0.9.

Task 2

With inspirations of task 1, we use another one learning rate function which is more stable comparing with the functions provided in the project. The learning rate function is shown in Eq. 1.

$$\gamma = \exp(-0.001 \times k) \tag{1}$$

And different learning rate functions are shown in Figure 2.

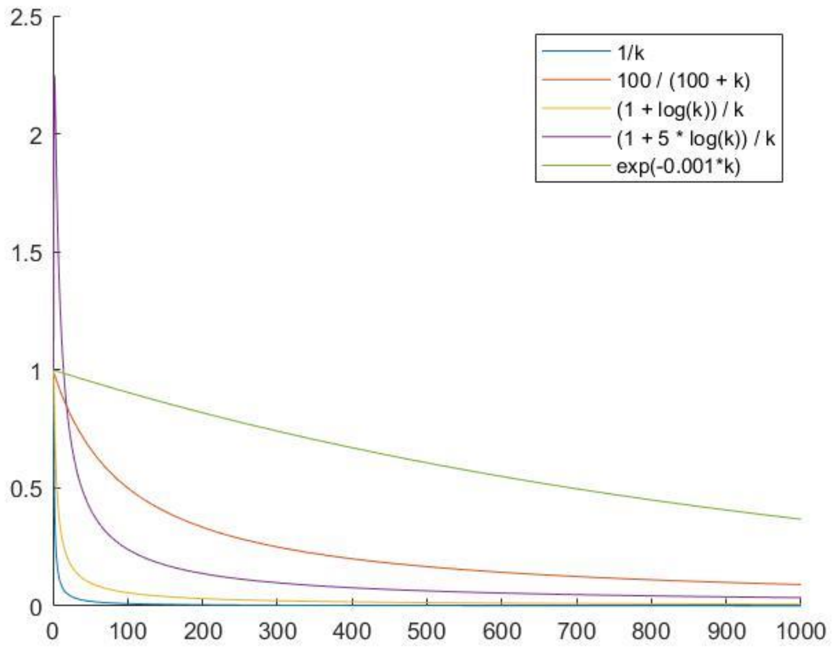

Figure 2. Comparison of different learning rate functions

As for discount ratio, we simply choose 0.95 for it. Because, times are all at the same magnitude and all can achieve 10 goal-reached runs on Task1.mat when $\gamma \geq 0.9$.