Solution of chapter 3rd exercises

* **Answer 3.1:**

Example1: Suppose reinforcement learning is used for automatic watering flower system.

The actions are to turn on/off the switches and to add the right fertilizers.

The states are the sensors’ readings, such as the temperature, the humidity and so on.

The rewards might be how good the flowers are, which can be judged by human beings.

Example2: Reinforcement learning is used for automatic problem query of customer service center.

The actions are to choose the right answer replying to the customers questions.

The states are the question asked by customers.

The rewards are if the question solve the customer’s question.

Example 3: Maybe we can use the reinforcement learning for the Traffic light dispatching

The actions are to control the traffic lights, namely choosing which one the light on and the lasting time.

The states are the numbers of vehicles on the road

The rewards are good or bad depending on the whole cars’ passing time.

* **Answer 3.2:**

As I can see, it seem the MDP is adequate to represent all goal-directed learning task.

* **Answer 3.3:**

I think the boundary between agent and environment is depended on what kind of issues. At this driving problem, it may need several agents to complete the whole process. Suppose I want a car drive itself to a place. At high level, there needs an agent to determine the optimal route. When it is on the road, it needs another agent to control the brake and wheels to run safely.

* **Answer 3.4:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| s | a | p(s’,r|s,a) | s’ | r |  |
| high | search | a | high | rsearch |  |
| high | wait | 1 | high | rwait |  |
| high | search | 1-a | low | rsearch |  |
| low | search | B | low | rsearch |  |
| low | search | 0 | high | -3 |  |
| low | wait | 1 | low | rwait |  |
| low | recharging | 1 | high | 0 |  |

* **Answer 3.5:**



Where S’ is the state set with terminal state.

* **Answer 3.6**

If it uses discounting with all rewards zero except for -I upon failure when treating pole-balancing as an episodic task, the return will be -1 for every episode. This return is different because the continuing formulation will add all the failure reward, but the episodic formulation will reset the return when failure happens.

* **Answer 3.7**

When you give a reward of +1 for escaping from the maze and tread it as a episodes task, the return of the episode task will always be 1 at each episode, which cannot improve the robot’s action.

You haven’t effectively communicated to the agent what you want it to achieve.

* **Answer 3.8**

The reward sequence is -1, 2, 6, 3, 2, so the returns can be calculated backwards.

G5 = 2;

G4 = 3 + 0.5\*G5 = 4

G3 = 6 + 0.5\*G4 = 8

G2 = 2 + 0.5\*G3 = 6

G1 = -1 + 0.5\*G2 = 2

* **Answer 3.9**

R1is followed by an infinite sequence of 7s, so G1 = 7\*/(1-0.9) = 70 because of (3.10),

Then G0 = 2 + 0.9\* G1 = 65

* **Answer 3.10**



(2) – (1)



So



* **Answer 3.11**



* **Answer 3.12**



* **Answer 3.13**



* **Answer 3.14**

 = 0.675

* **Answer 3.15**

Here we denote the new return as ,

then



Denote the new value of state as 



So



* **Answer 3.16**

For the episodic task, adding a constant c to all the reward may change the return with variable value which leads to the meaningless process.

For example, in a maze running, we set the wrong step as -1, and the right step as 1, the return will tell agent if this episode is good or not by compare the return. Once we add 1 to all the rewards, the wrong step will be 0, and the right step will be 2, which lead to every episode return to be the same. Another situation is that is each reward +2 = 1, then the solution for -1 will find the longest way.

* **Answer 3.17**



* **Answer 3.18**



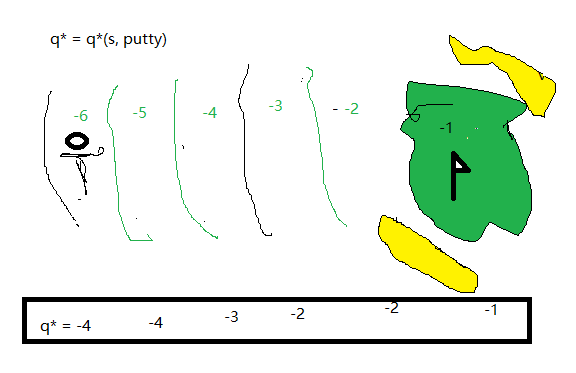
* **Answer 3.19**



* **Answer 3.20**



* **Answer 3.21**



* **Answer 3.22**

For , the optimal policy is 

For ,   
For  

* **Answer 3.23**



* **Answer 3.25**



* **Answer 3.26**



* **Answer 3.27**



* **Answer 3.28**



* **Answer 3.29**







