TBMI26 – Computer Assignment Reports  
Reinforcement Learning

Deadline – Mars 12 2018

Author/-s:

In order to pass the assignment you will need to answer the following questions and upload the document to LISAM. If you meet the deadline we correct the report within one week after the deadline. Otherwise we give no guarantees when we have time.

1. **Define the V- and Q-function given an arbitrary policy as well as a given optimal policy (See lectures/classes).**

Q-function:

V-function:

1. **Define a learning rule for the Q-function (Theory, see lectures/classes).**

In the beginning we explore more (we use a large ɛ) and then we lower the ɛ and exploit instead.

We can also change the learning rate.

1. **Describe your implementation, especially how you hinder the robot from exiting through the borders of a world.**

We initialize a look-up table for the size of the world (10 \*15) and then depth 4 ( dimension that describes the action). The we let the robot run and learn it’s way. For every 1000th epoch we lover the epsilon so that the robot in the beginning explore a lot and later on it exploits instead.

To avoid the robot bumping into the wall we use a while loop. If the action is invalid a new action is choosen until the action is valid. See following code:

while act.isvalid ~= 1

[a, oa] = chooseaction(look\_up, pos\_prev(1), pos\_prev(2), [1 2 3 4], [1 1 1 1], eps);

act = gwaction(a)

end

1. **Describe the differences between the worlds explored by the robot. Any surprises?**

The first world hasa blob in the middle. The robot has to learn to walk around the blob.

In the second world, the blob is only there sometimes. The robot then must learn how to handle suddenly appearing things.

In the third and forth world the robot has to options, to walk around the big blob or take the shorter path between the blobs.

1. **For each world: Plot the V-function, i.e. how do you get to the goal from each position.**

|  |  |
| --- | --- |
|  |  |
| **World 1** | **World 2** |
|  |  |
| **World 3** | **World 4** |

When all arrows are plotted in world 1 and 2 we can that world 2 is a bit more chaotic than world 1. In world 1 the robot wants to get out of the water, but it world 2 it isn’t trained for that in the same way.

1. **For each world: describe the key observations you have made with respect to parameter choices. Provide documentation of the parameters you have used for each figure! A good rule is to provide each figure with a caption. Plot policies and the V-function for appropriate worlds to the extent you find appropriate in order to explain what you have done and learned during the assignment.**

We tried different values for the parameters in each world in order to find the optimal ones. The eps has a high value from the start (0.8 or 0.9) and when it is only 10 percent of the epochs left, the eps is lowered by 0.6. That means that the robot explore more in the beginning and exploit more in the end of the training session.

.

The alpha controls the learning rate, lower values put more emphasis on already learned stuff. In world 1,2 and 4 a alpha around 0.5 is good while in world 3 we use an alpha of 0.1.

The gamma parameter controls in we would go for long term or short term rewards. The only world were we don’t want a high gamma is the forth world, here a low gamma makes sure that the robot doesn’t go into the blob.

|  |  |
| --- | --- |
| 1. eps = 0.8; 2. alpha = 0.5; 3. gamma = 0.9; | 1. eps = 0.8; 2. alpha = 0.4; 3. gamma = 0.9; |
| **World 1** | **World 2** |
| 1. eps = 0.9; 2. alpha = 0.1; 3. gamma = 0.9; | 1. eps = 0.9; 2. alpha = 0.5; 3. gamma = 0.1; |
| **World 3** | **World 4** |

1. **What would happen if we where to only use Dijkstra's shortest path finding algorithm in the ''Suddenly Irritating blob'' world? What about in the static ''Irritating blob'' world?**

In each world, there is costs for taking a move. In the Dijkstra algorithm, the costs are what determins how effective the algorithm will be. In World 1, there is an area where always higher, which means that the Dijkstra will not prefer walking that path. Which makes the area to explore smaller. In world 2, the cost aren’t always the same, sometimes the costs are the same for the whole world, and sometimes not. This will make it hard for the Dijkestra algotithm to find its way. In conclusion, it would have a easier time finding the goal in world 1 compared to world 2.

1. **Include an in-depth description of the to/from HG worlds (world 3 and 4). What happens on the way from HG? How and why can this problem be solved with Q-learning? Which path does the robot prefer, and why?**

World **4 is much h**arder than world 3 since in world 3 the robot can just walk straight without falling into the blob. In the picture from world 3 you can see that the robot walks straight though the shortest path between the blobs.

In world 4 it is hard for the robot to find the shortest path without going into the blob. Therefor the robot goes around the blob and is therefor punished by the algorithm for taking a longer path.

1. **Can you think of any application where reinforcement learning could be of practical use? A hint is to use the Internet.**

In manufacturing we can train a robot to optimize the speed of, for example, packing a box for shipping. This robot can learn by failing and then optimize on many different situations of packing.

1. **How does the different parameters () influence learning and appearance of the Q- and V-functions?**

The different parameters influence in terms of how the algorithm learns the optimal path to the goal. A lower learning rate puts more focus on already learned experience and a higher rate will overwrite previous learned experience.   
The discount factor regulates the long-term and short-term rewards. A values close to zero will maximize a short term reward. A high value will maximize long-term reward.

The exploration factor regulates how much the algorithm will explore. A high value will explore a lot and a low value will exploit a lot.