# **CAP 5636 – Fall 2020 - Homework 2**

Step 1: (10 pts)

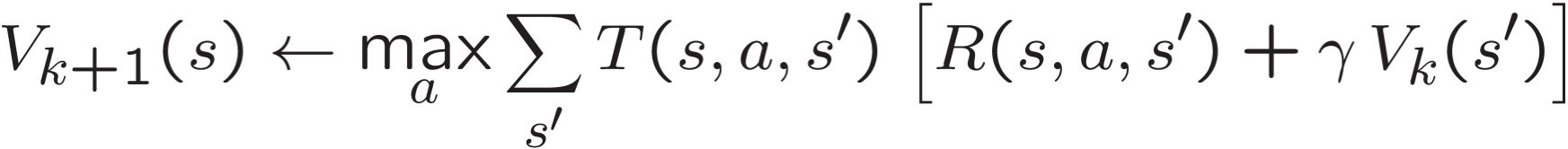
Solve problems Q1 to Q4 from the Berkeley AI reinforcement learning project:

**Q1. Value Iteration**

Solution: For this value iteration, we need to implement QValue formula which is as below:

Start with V0(s) = 0: no time steps left means an expected reward sum of zero.

Given vector of Vk(s) values, do one ply of expectimax from each state:



Now as to implement this through coding we need to edit below code in **valueIterationAgents.py:**

Text

Description automatically generated

Original code.jpg

After creating the learning code:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Below is command which we use to run the program:

python gridworld.py -a value -i 5

After writing code below is the final output:

A close up of a green screen

Description automatically generated

Output on grid.jpg

Below is Output on terminal:

Text

Description automatically generated

If we run the autograder command get the below output:

Text

Description automatically generated

For this Question we got below values against each iteration and the graph is plotted in Step 2.

|  |  |
| --- | --- |
| **Iteration** | **Values** |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0.37 |
| 5 | 0.5 |
| 6 | 0.58 |
| 7 | 0.61 |
| 8 | 0.63 |
| 9 | 0.64 |
| 10 | 0.64 |

**Q2. Bridge Crossing Analysis**

For this question we have to change analysis.py. Change the values for answerDiscount = 0.9 and answerNoise = 0.2.

Original code:

Text

Description automatically generated

We tried lot of combinations for answerDiscount and answerNoise values. Like answerDiscount = 0.1 and answerNoise = 0.5. For mentioned values below is output and we used this command:

python gridworld.py -a value -i 100 -g BridgeGrid --discount 0.1 --noise 0.5

Graphical user interface, application

Description automatically generated

A picture containing graphical user interface

Description automatically generated

After creating the learning code:

Text

Description automatically generated

We are using below command to get the output:

python gridworld.py -a value -i 100 -g BridgeGrid --discount 0.9 --noise 0.0

This is the final output for the program:

A picture containing table

Description automatically generated

**Q3: Policies**

In this question we need to use the optimal policy types for below. We are editing **analysis.py**, in the file we going to edit **answerDiscount, answerNoise and answerLivingReward** paramters in the file.

1. Prefer the close exit (+1), risking the cliff (-10)

For this we are editing def question3a:

Original code:

Text

Description automatically generated

After creating the learning code:

Text

Description automatically generated

1. Prefer the close exit (+1), but avoiding the cliff (-10)

For this we are editing def question3b:

Original code:

Graphical user interface, text

Description automatically generated

After creating the learning code:

Text

Description automatically generated

1. Prefer the distant exit (+10), risking the cliff (-10)

For this we are editing def question3c:

Original code:

Graphical user interface, text

Description automatically generated

After creating the learning code:

Graphical user interface, text

Description automatically generated

1. Prefer the distant exit (+10), avoiding the cliff (-10)

For this we are editing def question3d:

Original code:

Graphical user interface, text

Description automatically generated

After creating the learning code:

Graphical user interface, text

Description automatically generated

1. Avoid both exits and the cliff (so an episode should never terminate)

For this we are editing def question3e:

Original code:

A picture containing graphical user interface, text

Description automatically generated

After creating the learning code:

Text

Description automatically generated

Once done editing the analysis.py we are going to run code using below command:

python autograder.py -q q3

Below is the output for the program:

Text

Description automatically generated

**Q4:** **Q-Learning**

For this question we have edit 4 functions as mentioned in the instructions from file qlearningAgents.py and functions are update, computeValueFromQValues, getQValue, and computeActionFromQValues:

Original code:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

After creating the learning code:

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Final output on gridworld:

After running program for 100 iterations, we got below output:

We got different values through the 5 iterations for each episode and those values are stored in one csv file.

A picture containing shape

Description automatically generated

Table

Description automatically generated

Step 2: Visualize the evolution of the V-value (4 pts)

Chart, line chart

Description automatically generated

Step 3: Visualize the evolution of the Q-value (4 pts)

Chart, line chart

Description automatically generated