IIT Jodhpur

ARTIFICIAL INTELLIGENCE 2

ASSIGNMENT 2

REINFORCEMENT LEARNING

Instructor: Debarati Bhunia Chakraborty

Submitted by: Shristy Gupta

Roll Number: M20CS015

PROBLEM STATEMENT:
PROBLEM 1

d. **To Do:**

1. **Knowledge Base Creation:** How(in what format) the information of the visited nodes are stored in the agents memory.
   Print the knowledge base with every iteration
2. **Policy Design:** What set of rules are considered to design the agents behavior/ interaction in the environment.
3. **Reward Function Design:** Design a reward function that will help the agent to reach the goal as early as possible. Note: You can take inference from the Romania Map diagram in the book Peter Norvig-Russel. (A* search Section)
4. **Path Cost:** Print the total Path cost for the path followed by the agent. (Optimal path cost must be there in the end, as with more iterations, the agent will interact more with the environment and thus learn more which leads to more knowledge in the Knowledge base.
5. **Path Followed:** Print the path followed for every iteration and the most optimal path.

e. **Expectations:**

1. The complete assignment answer is divided into 2 sections.
   a. **Theory [50 Marks]**
      Which includes,
      i. Policy Design **[20 Marks]**
      ii. Reward Function Design **[15 Marks]**
      iii. Knowledge based format (How the knowledge about the world is stored and called when repeating) **[15 Marks]**

   b. **Code [50 Marks]**
      Which includes,
      i. Visualisation of the Agent world and agent interaction. (anyway you like it, for example, you may build it like a video stream)
      [10 Marks for Agent World, 10 Marks for Agent live interaction with the environment]
      Note: If the position of the goal, walls and other entities are not satisfied as stated, 20 will be deducted from this section straight away.
      ii. Path Cost: Final Path cost (You are required to print it on output window) **[10 Marks]**
      iii. Path Followed(You are required to print it on output window) **[5 Marks]**
      iv. Knowledge Base(You are required to print it on output window) **[5 Marks]**
      v. Visualisation in form of recorded video **[10 Marks]**

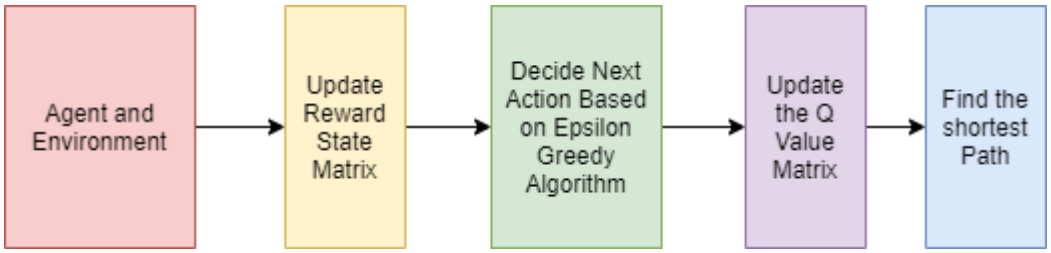
f. **Assignment Report:**
   a. All the details mentioned in theory section

b. Path travelled, path cost, knowledge base created and goal reached status for at least 3 iterations.
c. Path travelled, path cost, knowledge base created and goal reached status for final output.

SOLUTION 1

System Architecture:

Pygame has been used to create the game



Three main equation followed for deriving shortest path:

The Q value:

$$Q(s,a) <= (1-\alpha)Q(s,a) + \alpha(R(s,a) + \gamma maxQ(s',a'))$$

$$Q(s,a) = Q\ value\ of\ action\ from\ state\ s\ to\ s'$$

$$\alpha\ is\ the\ learning\ Rate\ which\ is\ chosen\ as\ 0.9\ here$$

$$R(s,a)\ Reward\ Function\ for\ action\ a\ from\ state\ s\ to\ s', stored\ in\ matrix\ here$$

$$\gamma\ is\ the\ discount\ factor\ for\ future\ rewards\ chosen\ as\ 0.9\ here$$

$$maxQ(s',a')\ is\ the\ max\ Q\ value\ for\ next\ state\ s'\ for\ all\ the\ future\ actions\ a'$$

Then compute the Temporal difference value with the help of the given Q value:

$$TD(S_t,a_t) = r_t + \gamma\ max_a(S_{t+1},a) - Q(S_t,a_t)$$

$$TD(S_t,a_t) is\ the\ temporal\ difference\ for\ the\ action\ taken\ for\ the\ previous\ state$$

$$r_t\ is\ the\ reward\ received\ for\ the\ action\ taken\ in\ the\ previous\ state$$

$$\gamma\ Discount\ factor, chosen\ as\ 0.9\ here$$

$$max_a(S_{t+1},a)\ Largest\ Q\ value\ available\ for\ any\ action\ in\ the\ current\ state$$

$$Q(S_t,a_t)\ The\ Q\ value\ for\ the\ action\ taken\ in\ the\ previous\ state$$

Epsilon Greedy algorithm:

$$\textbf{Action at time } (t) = \begin{cases} \textbf{max } Q(a) \textbf{ at time } t, & \textbf{probability } 1-\in \\ \textbf{Random Action}, & \textbf{probability } \in \end{cases}$$

Knowledge base Creation: Here each movable state/ white state has been assigned with -1 reward. This is because we always want to motivate the agent to take the optimal path and not get stuck in an infinite loop. This is also the reason that the path cost is slightly negative. The reward state of the obstacles is chosen as -100 as a penalty state and the goal state is chosen as 100

Policy Design: Here in each epoch optimal path is found by:

- Choosing an action like moving forward, backward, straight up or down based on *epsilon greedy algorithm to use exploitation vs exploration*
- *Update the Reward state and then update the temporal difference chart*
- Determine Q state for the state-action pair.
- Based on Q state find the optimal path from the bellman equation, and then move the agent in the desired path

Reward Function Design: The cumulative reward needs to be maximized this and the formation of loop has to be avoided hence following Reward state is initialized

- For movable state -1 reward
- For Obstacles/ Brown walls -100 reward
- For Goal +100 reward

Path Cost: Path cost is the overall cost for reaching the goal in 1000 epochs

Path Followed: Path followed will be the co-ordinates that are returned from Q-Value AS SHOWN AS FOLLOWS

Functions in the code:

```
find_island(current_row_index, current_column_index):
define actions
numeric action codes: 0 = up, 1 = right, 2 = down, 3 = left
Create a 2D numpy array to hold the rewards for each state.
The array contains 8 rows and 8 columns (to match the shape of the envi
ronment), and each value is initialized to -100.
```

```
shristy_state_action(state_action,action_index,row_index,column_index,world
_rows,world_column):
define a function that will get the next location based on the chosen
action
```

```
Q_Learning_Path(start_row_index, start_column_index, num_round):
```

Define a function that will get the shortest path between any location within the environment.
Use the concept of exploration and exploitation to deduce the best path

```
epochs(start_row, start_col)
run through 1000 training episodes
define training parameters
epsilon = 0.9,the percentage of time when we should take the best actio
n (instead of a random action)
discount_factor = 0.9 discount factor for future rewards
learning_rate = 0.9 the rate at which the AI agent should learn
receive the reward for moving to the new state, and calculate the tempo
ral difference
update the Q-value for the previous state and action pair
```

OutPut:

```
FINAL PATH COST
0.8
PATH FOLLOWED
ROUND 1: [[7, 0], [7, 1], [7, 2]]
ROUND 2: [[7, 0], [7, 1], [7, 2], [7, 3]]
ROUND 3: [[7, 0], [7, 1], [7, 2], [7, 3], [7, 4]]
: [[7, 0], [7, 1], [7, 2], [7, 3], [7, 4], [7, 5], [7, 6], [7, 7], [3, 7], [2, 7], [2, 6], [2, 5]]
```

```
KNOWLEDGE BASE
[[[-3.60000000e-01 -3.60000000e-01  0.00000000e+00  0.00000000e+00]
  [-3.60000000e-01 -9.00000000e+01  0.00000000e+00  0.00000000e+00]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [-3.60000000e-01 -6.51600000e-01 -9.00000000e+01  0.00000000e+00]
  [-6.91200000e-01 -6.87600000e-01 -6.87600000e-01 -3.96000000e-01]
  [-9.85716000e-01 -9.85716000e-01  7.10909640e+01 -6.91200000e-01]]
```

```
[[-3.60000000e-01  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-6.87600000e-01 -9.52956000e-01  8.96000000e+01 -9.99000000e+01]
 [ 5.18389776e+01  6.45358284e+01  8.02400000e+01  7.21472400e+01]]
```

```
[[-3.60000000e-01  9.99900000e-01  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00 -3.60000000e-01  0.00000000e+00 -9.00000000e+01]
 [-9.00000000e+01  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 8.02400000e+01  8.02400000e+01 -1.00000000e+02  1.00000000e+02]
 [ 7.18160000e+01  8.02400000e+01  7.18160000e+01  8.96000000e+01]]
```

```
[[ 4.96674000e-01  5.00000000e-01 -9.54045000e-01  4.99950000e-02]
 [ 1.00000000e+00 -9.99999000e+01  4.99999923e-02  4.99500000e-02]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-3.96000000e-01 -9.00000000e+01 -6.80760000e-01 -9.00000000e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 8.02400000e+01  7.18160000e+01  6.42344000e+01 -1.00000000e+02]]
```

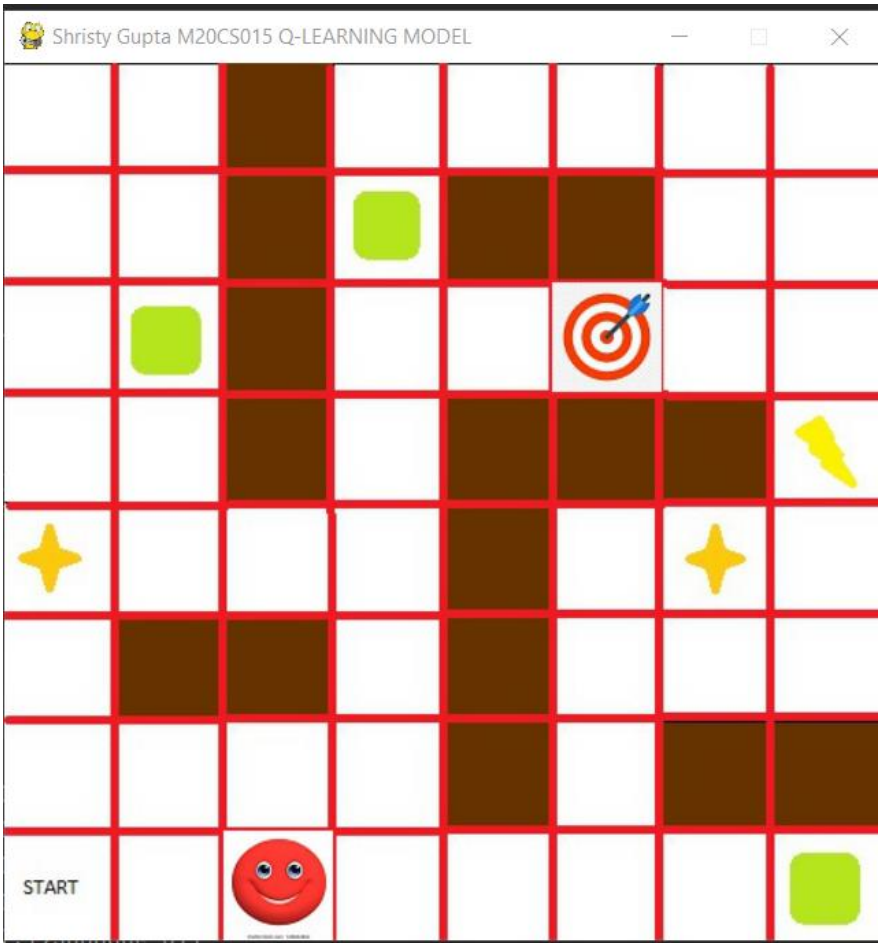
```
[[ 5.00000000e-02  4.99987833e-02 -1.25949097e+00 -9.55000000e-01]
 [ 5.00000000e-01 -3.61703341e-01 -9.00000000e+01 -9.57657600e-01]
 [-9.00000000e+01 -6.51600000e-01 -9.00000000e+01  4.99958930e-02]
 [-6.91596000e-01 -9.00000000e+01 -1.23008110e+00 -3.58432134e-01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-9.90000000e+01 -9.00000000e-01 -6.80760000e-01 -9.00000000e+01]
 [-9.00000000e+01 -6.51600000e-01 -1.01119727e+00 -3.60000000e-01]
 [ 7.18160000e+01 -1.25700156e+00 -1.18034676e+00 -1.29060000e+00]]
```

```
[[-9.55000000e-01 -1.00000000e+02 -1.53298283e+00 -1.25953123e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-7.76994881e-01 -9.00000000e+01 -1.26507000e+00 -9.00000000e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-9.51375600e-01 -1.07457660e+00 -3.87867600e-01 -9.00000000e+01]
 [-9.00000000e-01 -9.52956000e-01 -9.00000000e+01 -9.89727840e-01]
 [-1.25080956e+00 -9.85716000e-01 -9.00000000e+01 -9.56556000e-01]]
```

```
[[-1.25950000e+00 -1.78020339e+00 -1.78019502e+00 -1.53549973e+00]
 [-9.99000000e+01 -1.73306888e+00 -1.81278128e+00 -1.53355000e+00]
 [-9.00000000e+01 -1.51800483e+00 -1.04755000e+00 -1.53922587e+00]
 [-1.29389130e+00 -9.99000000e+01 -7.19500000e-01 -1.33844212e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [-6.91596000e-01 -9.00000000e+01  5.00000000e-02 -9.00000000e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

```
[[-1.53355000e+00 -1.34279500e+00 -1.60851550e+00 -1.60851550e+00]
 [-1.78019500e+00 -1.04755000e+00 -1.34279500e+00 -1.60851550e+00]
 [-1.34279500e+00 -7.19500000e-01 -1.04755000e+00 -1.34279500e+00]
 [-1.04755000e+00 -3.55000000e-01 -7.19500000e-01 -1.04755000e+00]
 [-1.00000000e+02  5.00000000e-02 -3.55000000e-01 -7.19500000e-01]
 [-3.55000000e-01  5.00000000e-01  5.00000000e-02 -3.55000000e-01]
 [-1.00000000e+02  1.00000000e+00  5.00000000e-01  5.00000000e-02]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]]
```

The final game board looks like below



PROBLEM 2

You need to write an extended abstract (2-4 pages) for the paper of your choice which is based on reinforcement learning, satisfying conditions mentioned below. There are 2 choices, you need to select any one of them. **[50 Marks]**

Format:

a. **Title Page.** On the title page include the title, your name, and the date.
b. **Abstract**. An abstract is a brief summary of **your review.**
c. **Introduction**
d. **Discussion**
e. **Conclusions**
f. **References**

Choice 1:

a. Paper published after Jan-2019
b. Published in conference: NeurIPS, CVPR,ICCV, CVPR, ECCV, AAAI, ICML (Strictly)

Choice 2:

a. Published in conference: NeurIPS, CVPR,ICCV, CVPR, ECCV, AAAI, ICML (Strictly)
b. Paper must have citations greater than 1000. No restriction on date published.

SOLUTION 2

The paper chosen for review is SME-Net: Sparse Motion Estimation for Parametric Video Prediction through Reinforcement Learning. It is choice 1 paper:

Choice 1:

a. Paper published after Jan-2019
b. Published in conference: NeurIPS, CVPR,ICCV, CVPR, ECCV, AAAI, ICML (Strictly)