1751048 – Khuu Kien An

1751050 – Dinh Ho Gia Bao

1751067 – Nguyen Minh Hieu

CS420 – Artificial Intelligence

Project 02 Report

Wumpus World

HCM UNIVERSITY OF SCIENCE

17/6/2020

1. Introduction

Wumpus world is the one of the most-well known problem in Artificial Intelligence for the knowledge-base section. The Wumpus world's objective is to kill at most as possible "Wumpuses" in the world with the purpose of not being killed by them. In the second project, we try to apply Logical search for level one and experiment Q-learning approach for level two in this project.

The Wumpus world is a dark cave that you can only see the nearest four doors's rooms without knowing anything inside those rooms and the rooms connect to each other. The room has a room with a monster which is called Wumpus that eat anything into its room. There are some Stench rooms near that monster's room for recognizing it. An agent can shoot it by the arrows. This world have some rooms contain a large, deep hole in the ground called Pit rooms. Agent can get the cold feeling with Breeze room to recognize the nearest pit . If agents accidentally enters that room, he will fall off and get stuck forever. And also that,there are some rooms that contain gold. The objective in the Wumpus world is that we try to explore as most as possible K rooms, try not to be killed by Wumpus and not to be fallen off at Pit rooms, picking gold is the optional for the reward, better result. In this case, we will implement the Logical search for the 10x10 size Wumpus world.

Assignment Plan:

We form three-member group. We want to try to logical search algorithm that learned in class to understand how they constructed, performed in the Wumpus world algorithms, research the Q-learning to understand, learn new concepts, knowledge in reinforcement learning.

The agent in our project start at (1,1) (for simplicity). The representation we apply for the each cell as states in the maps with five tuples: <Breeze, Stench, Pit, Wumpus, Gold, Ok> for one hundred cells including the initial potion. Cell will set the flag Breeze when it contains Breeze. For the Stench, Pit, Wumpus, Gold cells, it will do the same. For the safe, empty cell, it will set the tuple "Ok" be true. The agent can do some actions as moving "up", "right", "left", "down" directions, shooting for killing wumpus, picking the gold for gain more scores. We use five inputs for implementation and the path of the agent in Wumpus world is shown in console with expanded nodes and a final return path. Due to the lack of time, our current graphics just show the state expanded (Yellow for agent, Blue for wumpus and Red for gold. A Sleep(s) in Graphics.cpp can be adjusted to delay the output (for easily observe). The graphics library use is glut in OpenGL (x64).

Roles

Dinh Ho Gia Bao: Design the agent for the problem.

Khuu Kien An: Fix bugs, writing report for level 1, discussing for reasoning progress.

Nguyen Minh Hieu: Research Q-learning, write report for level 2, generating maps, support ideas.

2. Level 1

Agent do not knows anything in the world, there are some obstacles like monster, pits. It has only times to visit specific K rooms. The agent will have two set of actions are “Exploring” and “Reasoning”.

For “Exploring”, the agent will do the actions: moving the directions(it can go backward direction) for discovering anything new in the world like ‘Stench’, ‘Pit’ to gain the information so that it can collect for knowledge base to do more actions in exploring like shooting the Wumpuses, avoiding pits…

We have “Reasoning” for knowledge gaining by “Exploring” the world. While agents do the “Exploring” for new room as states, it also collect the information in the current room it stays. The information will be constructed into the logical sentences, store it into the knowledge base. And based on the logical sentences in the knowledge base, the agent will select the best action in “Exploring” for expanding more rooms in Wumpus World.

The progress of Reasoning:

At the beginning, the agent will move any directions since it didn’t gain any information from the knowledge base or some cases like it just moves empty rooms sequentially.

When agent arrives to Breeze’s Room, it will see the tuple “Breeze” in that cell is true makes agent collect that Breeze symbol for knowledge base. It will make sense for agent can assume that some adjacent cells near that Breeze cell as insecure cells. While agent are doing the “Exploring”, it try to dodge some insecure cells that it might have pit. For a cell from a room with breeze, if the opposite in horizontal or vertical direction of it is “A”, then that room will be absolutely a pit. Moreover, a room with number of potential pits be two will also be an absolute pit.

When it arrives to Stench's Room, it will see the tuple "Stench" in that cell is true, it will do the same like Breeze's Rooms for collecting Stench symbol for knowledge base. It make agent assume that some adjacent cells near that cell might have Wumpus. It will do the action "Shooting" in 4 adjacent rooms (except the agent's previous room) in "Exploring" for killing Wumpus to expand more spaces, rooms, which cost a 1000 decrease in score. The value "S" from the original room value will be deleted from the string, others remain unchanged.

If agent assume some cells as insecure cell might have both Pit and Wumpus, it will set that tuple "Ok" in cell is true. Because one room cannot have both Pit and Wumpus at the same time.

For the cell contaning gold, the agent will pick the gold immediately to gain 100 score.

The progress of Exploring:

The maximum number of rooms that an agent can visit is K. If a value K is reached or visit a wumpus cell (set status to dead), the agent will return to its start at (1,1). A Bread-First Search algorithm is used for expanded nodes with a frontier and visited status for each cell. If a cell is not in the frontier or explored set and it is ok to go next, it is pushed back in the frontier to be expanded in the future. We also define a data structure for return path, which is by using a backtrack table. This table is quite similar to the mechanism of a hash table, each entry will contain a key to a queue to store previous cells from a cell.

Some significant output:

- **Normal input:**

```
-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.B.-.-.-
S.-.-.-.-.B.P.B.-.-
W.SG.-.-.-.-.B.G.B.-
S.-.B.-.-.-.-.B.P.B
A.B.P.B.-.-.-.-.B.-
```

Input

```
************************************ Return path ************************************

Agent go to (1,5)
(1,1) (2,1) (3,1) (2,1) (2,2) (2,1) (3,1) (4,1) (5,1) (6,1) (7,1) (8,1) (7,1) (7,2) (7,1) (6,1) (6,2) (
6,1) (5,1) (5,2) (5,1) (4,1) (4,2) (4,1) (3,1) (3,2) (3,1) (3,2) (2,2) (3,2) (4,2) (5,2) (6,2) (7,2) (6
,2) (6,3) (6,2) (5,2) (5,3) (5,2) (4,2) (4,3) (4,2) (3,2) (3,3) (3,2) (3,3) (4,3) (5,3) (6,3) (5,3) (5,
4) (5,3) (4,3) (4,4) (4,3) (3,3) (3,4) (3,3) (3,4) (4,4) (5,4) (4,4) (4,5) (4,4) (3,4) (2,4) (3,4) (3,5
) (4,5) (3,5) (3,4) (2,4) (2,5) (3,5) (2,5) (1,5)


Agent is coming back to start!!!
Back to cell (1,4)
Back to cell (1,3)
Back to cell (1,2)
Back to cell (1,1)

************************************ Total Score : -1900 ************************************
```

Output

- **Agent is eaten by Wumpus**

```
-.-.SG.W.S.W.S.W.S.G
-.BG.G.S.B.S.G.S.W.S
B.P.B.BG.P.B.-.S.S.B
B.BG.P.B.B.-.S.W.BS.P
P.B.B.P.BG.SG.W.S.-.B
B.BS.P.B.S.G.S.G.G.-
S.W.B.S.W.S.B.-.-.G
-.S.P.B.S.B.P.B.-.B
S.W.BS.BG.B.P.B.P.B.P
A.SG.B.P.B.B.-.B.-.B
```

Input

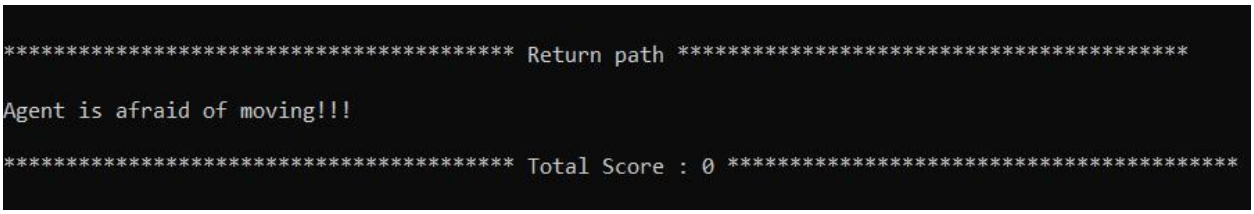
```
************************************* Return path ***************************************

Agent go to (4,2)
(1,1) (2,1) (3,1) (2,1) (2,2) (1,2) (2,2) (2,1) (3,1) (4,1) (3,1) (4,1) (4,2)
Agent is eaten by a wumpus at (4,2)

************************************* Total Score : -11900 ********************************
```

Output

- **Agent is afraid of moving due to several breeze rooms surrounded:**

```
BG.P.B.-.-.-.-.-.-.-
-.B.-.G.-.-.-.BG.-.-
-.-.-.-.-.B.B.P.B.-
G.G.B.G.B.P.B.B.-.-
-.B.P.B.-.B.-.-.-.G
BG.-.B.BG.-.-.-.B.-.-
P.B.B.P.B.-.B.P.B.G
B.P.BG.B.-.B.P.B.P.BG
-.B.-.B.-.-.BG.G.B.-
A.-.B.P.B.-.G.-.-.G
```

Input



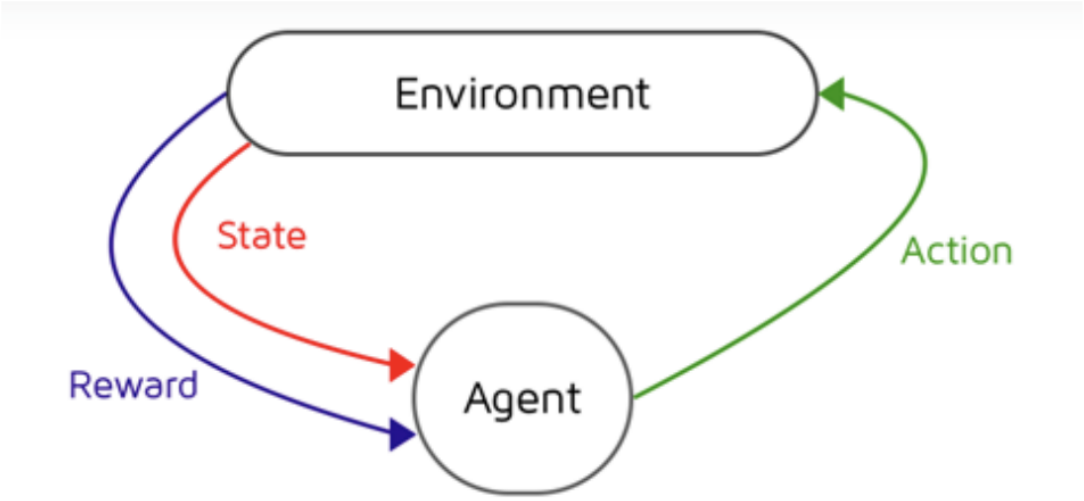
Output

3. Level 2 - Report

Q-Learning for Wumpus World

Reinforcement learning: the agent will try and interact (over and over again) with the environment for each interaction (creates new state) it receive a feedback(a score, reward,..) to judge if it good or bad, and it will learn from it.

It is similar like training a dog or a cat, for each of their action like sit, roll over, we reward them with food, and overtime they will learn our commands, for example: when we tell them to sit, if the dog don't sit, we make them sit and reward them with a food, and we do it over and over, eventually, at some point, when you tell the dog to sit, it will sit. Which means the dog learned that it behavior is a good behavior.



(reinforcement learning – from CS420 Sildes)

We consider a set of states S, set of actions A, set of rewards R.

At time step t, the agent receive some representation of the environment (state s_t), from that it perform an action A and we have (s_t, a_t) then the environment will be transitioned to new state s_{t+1} (time step t+1), at this time, the agent will receive a reward r_t.

Q-Learning is a model-free, value based Reinforcement Learning algorithm. Q-Learning seeks to **find best action to take in the current state** (maximize the total reward), find the optimal policy by learning the optimal Q-Value for a pair of state and action. And all the Q-Values for each pair of state and action is stored in a Q-Table. Q-Learning apply for stochastic problem whereas the states are not well-determine such as Wumpus World,….

How Q-Learning works?

Formula based on (Bellman Equation):

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)}^{\text{temporal difference}}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

As we have explain above:

- s_t is the old state when agent performs action a_t a transition the environment to new state s_{t+1}. α and γ are learning parameters.
- $Q(s_t, a_t)$ is the Q-Value when agent performs action a_t at state s_t.
- α is the learning rate. (0 □ α □ 1).
- γ is the discount factor (0 □ γ □ 1).

How do it work?

The Agent must avoid danger and collect the gold.

Step 1: Initialize a Q-Table

Take the example of Wumpus World Game. We have 10 x 10 map which there are 100 states. And 5 actions which is “Go Up”, “Go Down”, “Go Left”, “Go Right”, “Shoot Arrow”. As first, the agent doesn’t know anything about the environment so we set those Q-Value = 0.

States	Actions					
		Up	Down	Left	Right	Shoot
	S_0	0	0	0	0	0
	….	..	..	..	..	..

	S_{99}	0	0	0	0	0

Step 2: Choose an action:

- Based on highest Q-Value in Q-Table at a state.
- If the first time (all Q-Values is 0), choose random action.

Step 3: Perform action

Step 4: Receive feedback (Reward)

Example in Wumpus World:

- **Add 100 points** for picking up each **gold**.
- **Reduce 1000 points** for shooting an **arrow**.
- **Reduce 10000 points for dying** (by being eaten by the Wumpus, falling in a pit, or being trapped inside the cave).
- There is no cost for moving from one room to the next.
- There is no bonus point for killing Wumpus.
- Agent will have 0 point at the beginning.

Step 5: Update Q-Table, by calculating the Q-Value with above formula. Then we repeat the step 2 again, **when reach the maximum steps or died**, **it will consider end an episode, and another episode starts the agent will start over**, and the Q-Table also updated through episodes (the agent build up knowledge over trials), the agent will learn through many experiences after an amount of episodes.

Why can it used to solve the problem?

Wumpus World is a stochastic problem which means the agent just know the current state, it does not know the next state until it performs an action. Q-Learning is an “model-free” reinforcement learning, and the model-free advantages is it needs no accurate representation of the environment in order to be effective.

With the Q-Value in the Q-Table is updated every episode(try to go and fail/success again and again), after an amount of episodes the agent will learn how to avoid the pit and the Wumpus and take the gold, by taking the highest Q-Value at each state with action.

Comparing with logical search :

In Logical search, the agent already has knowledge about how to avoid obstacles, it don’t need the training process.

In Q-Learning, the agent has to learn through trials and errors (training process).

4. Reference

Q-Learning research:

[1] https://towardsdatascience.com/q-learning-54b841f3f9e4

Viblo: [2] https://viblo.asia/p/gioi-thieu-ve-hoc-tang-cuong-va-ung-dung-deep-q-learning-choi-game-cartpole-Az45bYy6lxY#_buoc-2-implement-thuat-toan-9

[3] https://www.quora.com/How-does-Q-learning-work-1

Deeplizard:

[4]https://www.youtube.com/watch?v=qhRNvCVVJaA&list=PLZbbT5o_s2xoWNVdDudn51XM8lOuZ_Njv&index=6

https://www.quora.com/What-is-the-difference-between-model-based-and-model-free-reinforcement-learning

Wiki: [5] https://en.wikipedia.org/wiki/Q-learning

Freecodecamp: [6] https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/

Q-Learning apply to Wumpus World:

[7] https://prezi.com/ybhwl7golsx0/q-learning-algorithm-to-solve-wumpus-world/

[8] https://github.com/albertfiati/Wumpus

[9] https://pdfs.semanticscholar.org/1117/cac935fe43d1744ccf762b2b486003ca07f5.pdf

[10] http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=90B3575614B598D1CE91A7BAF7769180?doi=10.1.1.143.8088&rep=rep1&type=pdf