Wumpus Project Report

Kaavya Rekanar
kare15@student.bth.se
940521-7184

Siva Venkata Prasad Patta
sipa15@student.bth.se
931221-7137

Aim:

Create an intelligent agent that understands the rules of wumpus game and play it accordingly in Wumpus world to obtain an acceptable score. A template of the code has been given to us, to which changes and developments are to be made such that the agent's performance is enhanced. We intend to achieve our aim using Q-learning algorithm for programming.

Modifications:

Myagent.java:

The modifications to the code have all been done in MyAgent.java itself, and some of the significations changes have been listed below:

Line 13-18: initialization of required variables

Line 42: Marking of the map borders

Line 70: doAction() starts. Simulation is run for the first time

Line 101: Qlearning starts

Line 166: Actual steps of Qlearning

Line 278: Decides a move based on exploration vs exploitation

In MyAgent.java a simulation program has been implemented which clones the wumpus world and stores it in the (world w2). A simulation is run for 100 times on the (w2.world) when each map is executed for the first time. The simulation runs when we click on the **Run Solving Agent** for the *first time* and stores all the rewards in the w2.world and choses the best bath when we run the program.

Approach used:

Q-learning approach has been chosen by us as we opted to go for the model free active Learning based system. The formula that has been used to calculate the q-values for the state action pairs in the algorithm is:

$$Q(s,a)=Q(s,a)+\alpha(R+\gamma \max a' Q(s',a')-Q(s,a)) \quad \text{//line 221 in code}$$

Where,

s= *previous state*	R=*reward*
a=*previous action*	α=*learning rate*=$\frac{1}{1+N(s,a)}$
s'=*current state*	γ=*discount factor*=0.5
a'=*current action*	

Two arrays of size 4x4x5 and 4x4x4 have been used to store the q-values and n-values respectively. Here q is used to store the location of the rewards and the location of the Wumpus i.e., the first two dimensions represent the location of cell in the map and the third dimension represents the four possible directions in each cell. The direction of Wumpus (if found in neighboring cell) is stored in the 5th place of third dimension in the q-array.

Thus, each data cell in this 4x4x5 and 4x4x4 matrices represents a state-action pair. The N-values, as discussed earlier represent the frequencies of the state action pair. A threshold value of 7 has been defined to count the number of times an action can be possibly executed in one state to avoid unnecessary exploration and converge on the solution faster.

The rate of learning of the agent is high in the beginning and decreases subsequently as the state action frequency increases, the rate of learning decreases. The *α value establishes this. The discount factor is designed to make the agent prefer long term high rewards over short term rewards, for an optimal result. The agent can be made impartial to both by setting the γ value to 0.5.*

The various reward values are:

Killed by wumpus= -1000
Getting gold= +10000
Falling into pit= -5000
Shooting wumpus= null

The Explanation of Code Functioning:

1. The variables are initialized in the constructor which is called once per new map. The variables are:
    a. n, q arrays hold N values and Q values respectively are initialized to -1.
    b. The map boundaries are given negative reward in a q-table.
    c. The previous state is recorded in the three variables prevX, prevY, prevDir
2. If a map is run for the first time, simulate to update the Q value by reinforcement learning.
3. Simulation is repeated for 100 times on a copy of the map by calling q_learn(). It also records the location of the Wumpus when the simulation game ends.
4. Perform basic actions in the present cell as well as update the Q value of the previous state based on formula. Call decideMove() and go to next cell (after calling turn() to orient in the decided direction) while updating the N value.
5. The decideMove() method returns the least frequent state action pair if it is less than threshold else returns highest Q value based pare.