

CZ3005 Artificial Intelligence

Lab Assignment 2

**Lab Group: TS6**

**Submitted By:**

Lee Su Bin

Koh Jun Jie

Goh Kee How

# 1. Tasks

**Agent.pl:** To help the agent navigate successfully and cover all possible safe locations without meeting the wumpus and the confundus approaches, our algorithm makes use of a few key ideas in explore(L).

The first key idea is that we always look for non-visited locations when deciding which is the next location to move towards from the current location. This allows us to cut back on unnecessary backtracking on locations that have already been visited. For instance if location (0,1) contains a stench, we would retreat back to location (0,0) and we can reason that locations (-1,0), (0,-1) and (1,0) remain unvisited. We will thus move to (-1,0) as the next action Agent will take. Should we come back to (0,0) again, we will then choose (0,-1) to visit as it the next unvisited location.

The second key idea is to reason safe locations using the absence of tingle/stench sensory inputs. For example, if the agent moves from (1,0) and to (1,1) and detects that there is a tingle at (1,1), it will reason that a Confundus Portal possibly exists at locations (0,1), (1,2) and (2,1). After some time, if the agent ends up at location (3,1) which is a safe location i.e. no stench/tingle, it can reason that location (2,1) does not contain a Confundus Portal. This is because (3,1) borders (2,1) and if (2,1) was to contain a Confundus Portal there will naturally be a tingle sensory input at (3,1) which is not in this case. Therefore, the agent can safely navigate to (2,1) and eliminate any possibility of a Confundus Portal in (2,1). The same strategy can also be applied for reasoning of safe locations using the absence of stench stationary inputs.

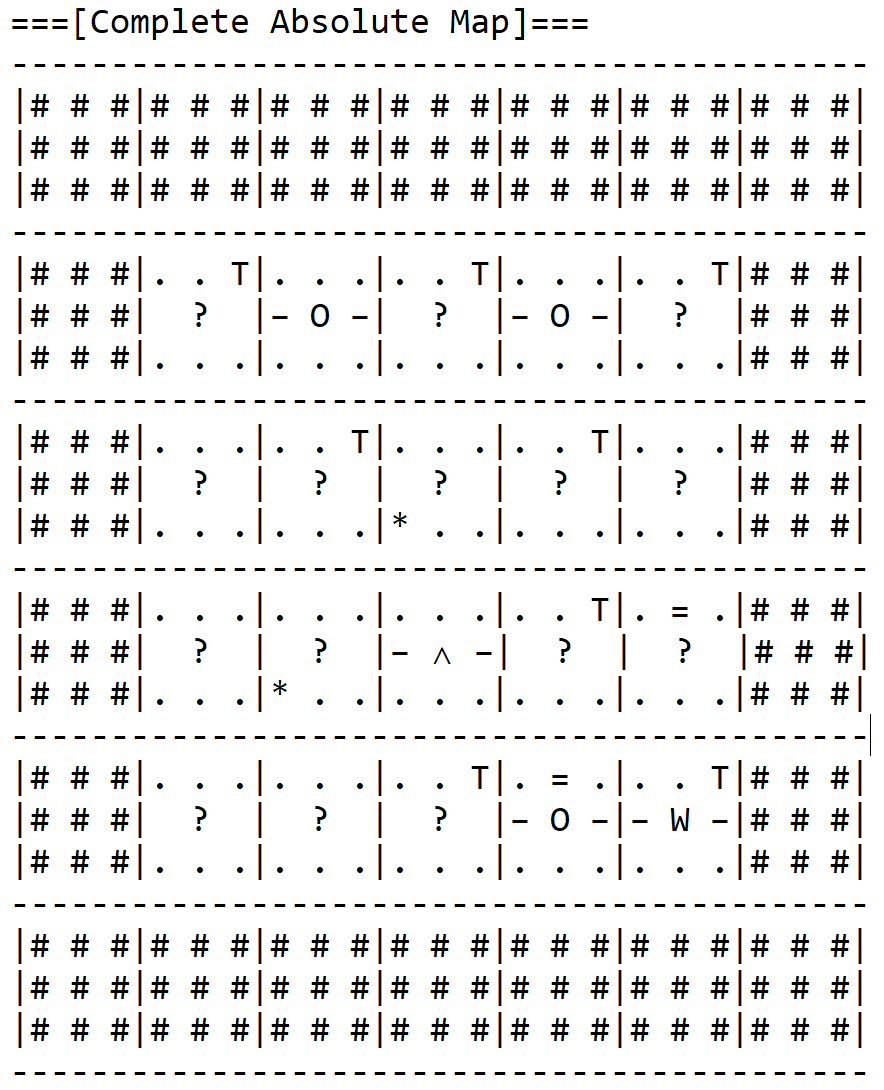
The third idea is that when the agent does not have any unvisited locations around its border. In this case the agent will select from among the visited locations the location that has the fewest visited counts to visit. For example, if the agent is at location (3,4) and locations surrounding it, (2,4), (4,4), (3,5) and (3,2) have all been visited with visited counts 2,3,3,3 respectively. It will visit (2,4) as the next action. In this way, the agent avoids being stuck since there are no other unvisited locations around it. And once the agent has moved to the next location, it is possible that there are unvisited locations surrounding the new location that agent has moved to and at that point the first key idea mentioned above will allow the agent to navigate to this univisited location. If all the locations surrounding the new locations are also visited, the same strategy will be applied to find the least visited location. This will in theory allow the agent to move through all the visited cells without being stuck.

The fourth key idea is to be able to reason the exact location of the Wumpus if the agent knows 3 out of the 4 locations with stench sensory inputs. There are three options on how the Wumpus can be located on the map. Firstly, it can be located at the corner locations which means two of the locations bordering the Wumpus are walls and the other two locations are locations that the agent can move to. Secondly, it can be located on the border of the walls which means three of the locations bordering the Wumpus are locations that the agent can move to. Lastly, it can also be located anywhere else which means that the agent can access all four locations surrounding the Wumpus. For example, if the agent knows that there are stenches at locations (3,4), (3,2) and (4,2), it can reason that the Wumpus is at (3,3). Using (3,4), (3,2) and cartesian coordinates principles, we know that the x-coordinate of the Wumpus locations is at 3. To derive the y-coordinate, we can sum up the y-coordinates of (3,4) and (3,2) and divide it by 2. In this way we can safely reason the location of the Wumpus and take actions to kill it. The same principle can also be applied if we have two horizontal locations of stench instead. Unless the Wumpus is located at the corners, this strategy allows us to remove the wumpus from the map and free up unvisited locations for the agent to move towards.

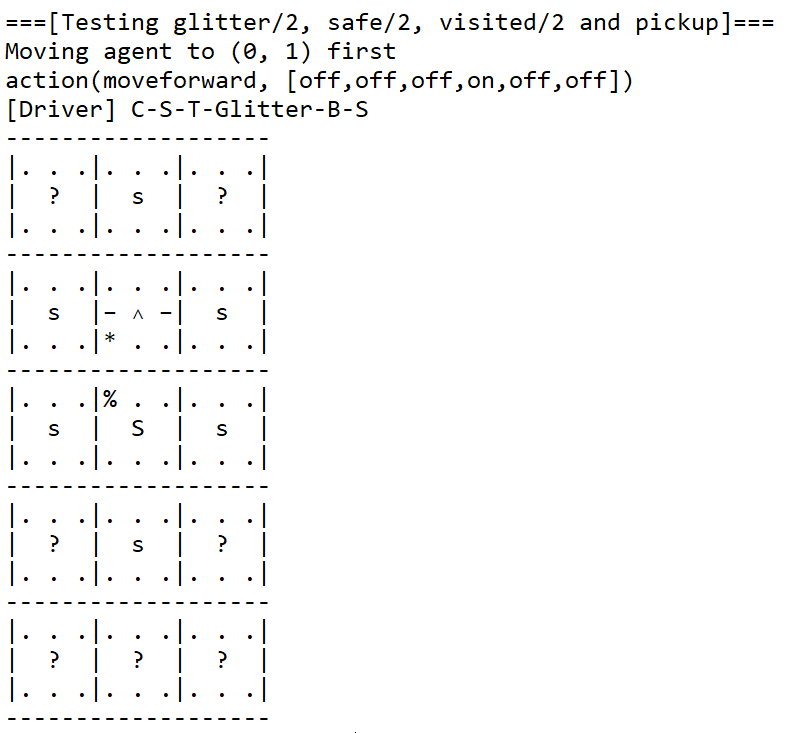
**Driver.py:** The Wumpus World Driver has to implement a map of **size 7 × 6** with the outer cells inhabited by Walls, and the inner cells populated by at least one Coin, one Agent, one Wumpus and 3 (three) Confundus Portals. The initial position and direction of the Agent can be random but should be safe. All sensory input generated for the Agent should refer to this map of the Wumpus World. The Driver is to ask the Agent to reset (reborn/0 call), and then create a feedback loop with the Agent that tests:

* Correctness of Agent’s localisation and mapping abilities,
* Correctness of Agent’s sensory inference
* Correctness of Agent’s memory management in response to stepping through a Confundus Portal
* Correctness of Agent’s exploration capabilities
* Correctness of the Agent’s end-game reset in a manner similar to that of Confundus Portal reset.

**Driver\_Printout.txt:** Initial absolute map of the world depicted by an array of map cells. Rows correspond to the second coordinate of the absolute position and the columns to the first coordinate of the absolute position. The map begins at a new line of the printout and finishes with a new line. The image below is the Absolute map’s layout. The remaining layouts & iterations can be found in the Printout.txt file.



Iteration of the following as many times as necessary – Action sequence printout (e.g., ”moveforward, turnleft, moveforward, pickup”) finished by a newline.



# 

# 2. Contributions

Agent.pl - Koh Jun Jie, Lee Subin

Driver.py, Driver\_Printout.txt - Goh Kee How, Koh Jun Jie

Friend-Driver.py - TeamPatience.py

# 3. Conclusion

Logic programming in Prolog is a completely different programming paradigm and thus we have to think and reason in a systematic fashion in order to get the outcome that we want. It was challenging at first, but after spending some time learning, it became more manageable.

# 4. References

<https://github.com/hilios/wumpus-prolog>

<https://github.com/shadabsk/Artificial-Intelligence-Lab>

<https://www.javatpoint.com/the-wumpus-world-in-artificial-intelligence>

<https://github.com/erikphillips/wumpus_world>

<https://github.com/MehwishQazi/WumpusWorld>

<https://github.com/gregscott94/wumpus-world>

<https://github.com/jason-padilla/WumpusWorldGame>

<http://www.let.rug.nl/bos/lpn//index.php>