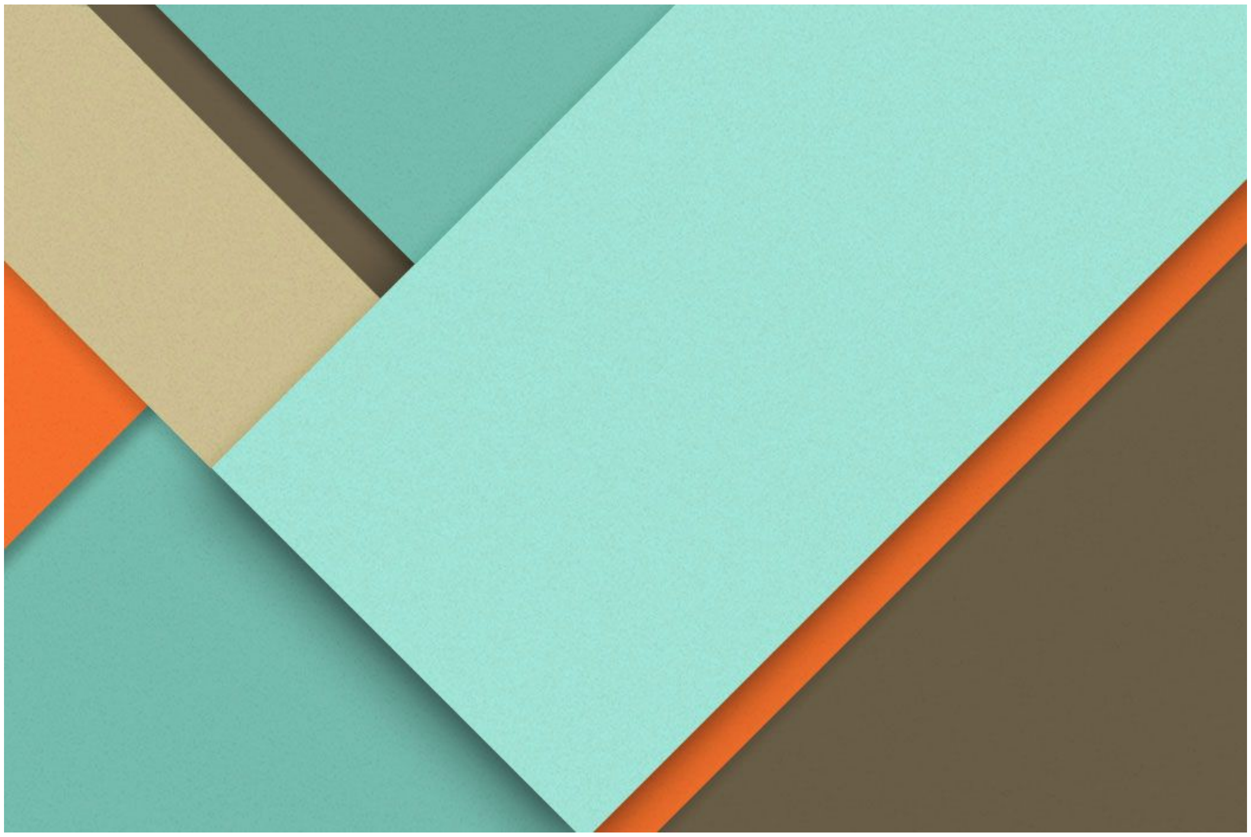


The Wumpus World:

a probability based agent

Yongrui Pan (n10296255)
Jianwei Tang (n10057862)

A statement of completeness

We have defined/completed two functions in probability_based_move.py file.

They are:

1. PitWumpus_probability_distribution(): it returns the joint probability distribution of the wumpus world.
2. next_room_prob(): it returns the cell location for the next move.

We did not modify any part of the code which was not allowed to modify.

The workload is fairly distributed. I would like to use a table to describe it.

The completeness of the function PitWumpus_probability_distribution()	Tang Jianwei
The completeness of the function next_room_prob()	Pan Yongrui & Tang Jianwei
Code comments	Pan Yongrui
Report (test cases not included)	Tang Jianwei
Test cases	Pan Yongrui

Probability-based approach

The approach we have implemented is the one shown in the lecture slides: The inference by enumeration. The main reason of using this approach is because the function to calculate $\mathbf{P}(BS\_known \mid P\_q, PW\_known, y)$ is provided as consistent(self,known_BS,event) in the_wumpus_world.py. It is easily used and understood.

The concept of this approach is to calculate the probability summation of every legit Permutation and Combination of the wumpus world. The risk of the next move is the probability summation of every situation having a pit/wumpus in the next move. The illegit situations are filtered by function consistent(self,known_BS,event). For example, if a wumpus world which has a pit without breeze around, it will be filtered as illegit.

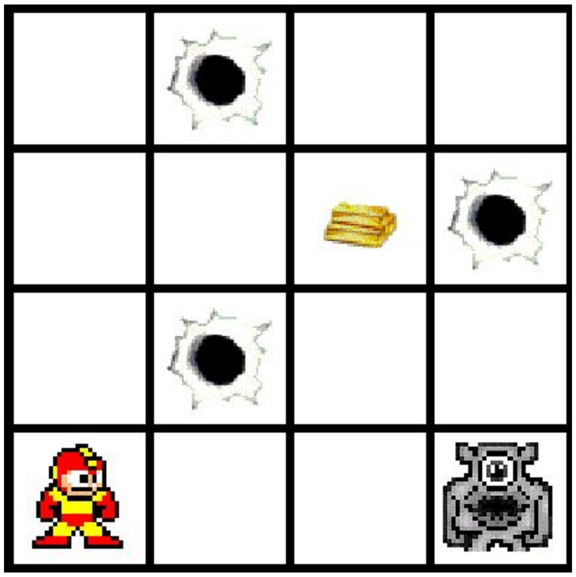
- We can calculate $\boldsymbol{P}\left(P_q|PW_{known}, BS_{known}\right)$ by doing an inference by enumeration:

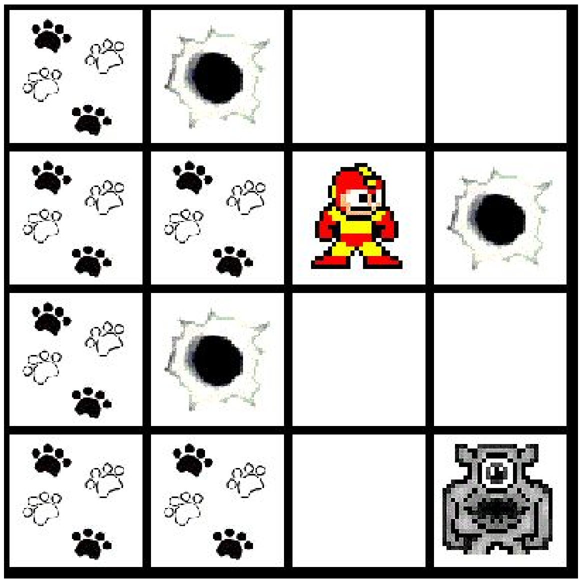
$$\boldsymbol{P}\left(P_q|PW_{known}, BS_{known}\right) = \alpha \sum_{y\in R_{unknown}} \boldsymbol{P}\left(P_q, PW_{known}, BS_{known}, y\right)$$

$$= \alpha \sum_{y\in R_{unknown}} \underset{\uparrow}{\underline{\boldsymbol{P}(BS_{known}|P_q, PW_{known}, y)}} \times \underset{\uparrow}{\underline{\boldsymbol{P}(P_q, PW_{known}, y)}}$$

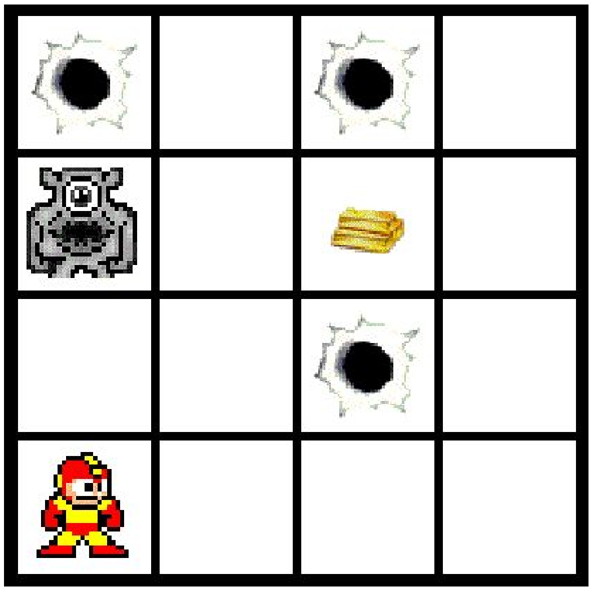
It is worth noting that we made an improvement toward the way searching next move. (we have discussed it with the tutor in the last tutorial). According to the requirement, we are supposed to consider the queries as the cells around the person. It means the probability calculation would be performed right after the person standing on a breeze cell. However, here we try to expand the queries by discovering every safe room before doing probability calculations, so the query rooms may not directly near the person's cell. As a consequence, we find the safest room by comparing every possible query. It brings better performance.

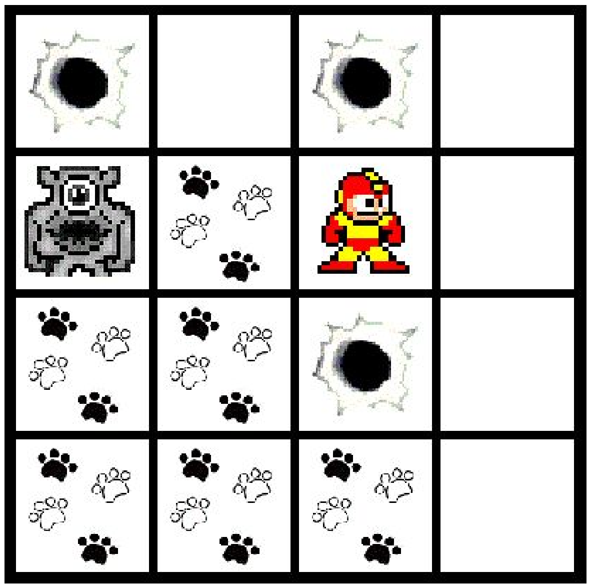
Test cases



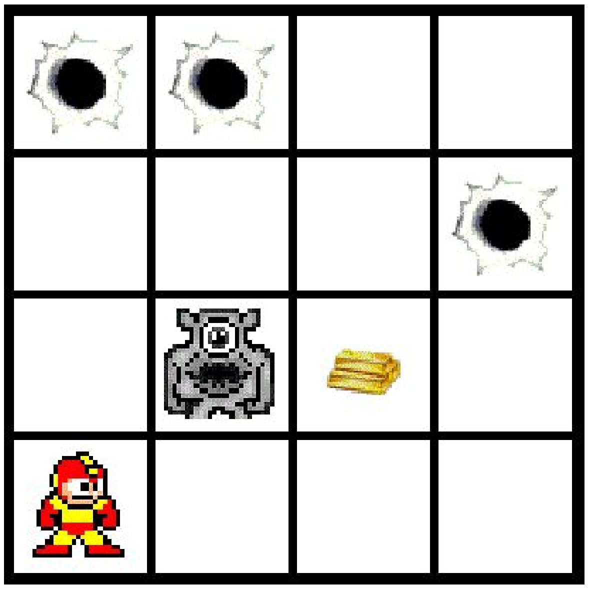


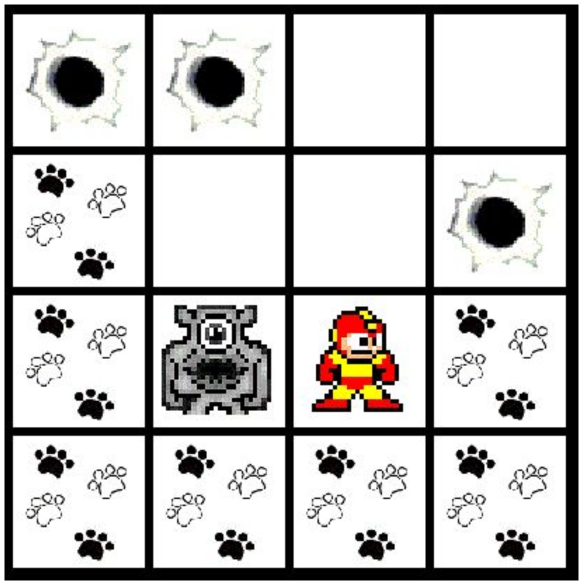
In the first case, the logic-based agent can only visit room (1,3) and (2,4) then went back to (1,4) and stop. Our probability-based agent calculated room (1,2), (2,3) and (3,4) and found that room (1,2) has the lowest probability of contain a pit/wumpus in the three rooms and move to room (1,2). Then, the logic-based agent can help to move to room (1,1) and (2,2) which are safe as they are not adjacent to breeze/stench. Then the probability-based agent started to calculate the probability of the four current query rooms (2,1), (3,2), (2,3), (3,4) and found that (3,2) has the lowest probability and found the gold.





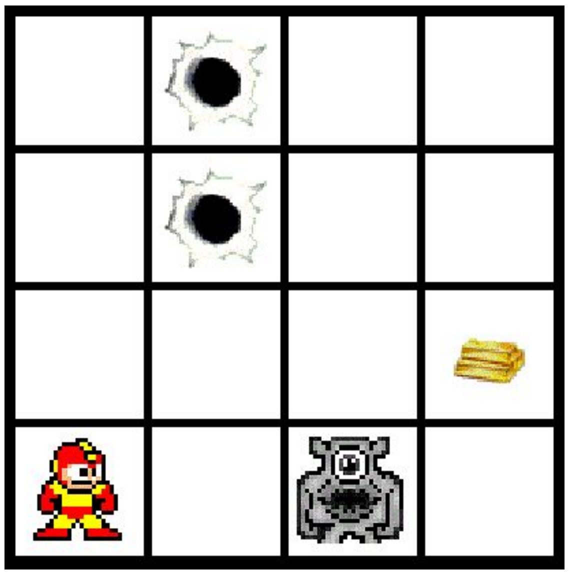
In the second case, the logic-based agent visited room (1,3), (2,4), (2,3), (3,4) then went back to (1,4). Then the probability-based agent calculated wumpus/pit probability of the four query rooms (1,2), (2,2), (3,3), (4,4) and found that (2,2) has the lowest probability and moved to the room. Then, because there is still no safe room, the probability-based agent calculated the probability of current five query rooms, (1,2), (2,1), (3,2), (3,3), (4,4), and found that (3,2) has the lowest probability then found the gold.

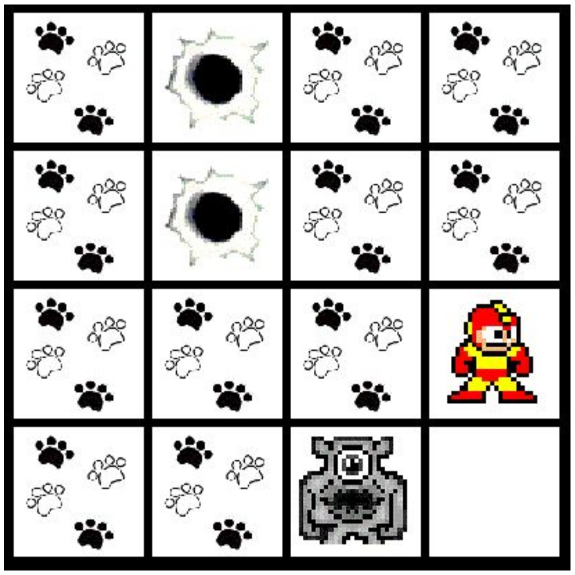




In the third case, the logic-based agent can only visit room (1,3) and (2,4) then went back to (1,4) and stop. Then the probability-based agent calculated room (1,2), (2,3) and (3,4) and found that room (1,2) has the lowest probability in the three rooms and move to room (1,2). There is still no safe room, so the probability-based agent continued to calculate room

(1,1), (2,2), (2,3), (3,4) and found that (3,4) has the lowest probability then moved to it. Finally, the logic-based agent found the gold in room (3,3)





In the fourth case, the logic-based agent can only visit room (1,2), (1,3), (2,3), (2,4) then went back to (1,4) and stop. Then the probability-based agent calculated the probability of room (1,1), (2,2), (3,3), (3,4), and found that (1,1) has the lowest probability and moved to it. Then the probability-based agent calculated room (2,1), (2,2), (3,3), (3,4), and found (3,3) has the lowest probability then moved to it. There is still no safe room, so it started to calculate the probability of query rooms (2,1), (2,2), (3,2), (4,3), (3,4) and found (3,2) has the lowest probability and moved to it. Then the probability used the same strategy and move to (3,1), then (4,1). Finally. The logic-based agent found safe room (4,2), (4,3) and found the gold.