**CS-171 Wumpus World Final AI Report**

**Team name NAXavierDog**

**Member #1 (name/id) Jiliang Ni (81224742)**  **Member #2 (name/id or N/A)**

[Delete this line.] Space suggestions below are approximate estimates. Use more or less text in each, as desired.

[Delete this line.] You are obliged to use Times New Roman font, size 12. **Do not exceed one page total.**

**I. In about 1/2 page of text, describe what you did to make your Final AI agent “smart.”**

In fact, when we consider to face the unobservable map, we let agent move along a fixed and pre-set route, and when agent meets breeze, stench, bump, it will make the corresponding action.

事实上,当我们思考这个不完整地图的信息时, 我们让agent以一种固定的路线走, 并且对于碰到风,臭气,墙做出分别对的操作,当拿到金子时,直接以最近的路线回家,没拿到金子则一直按照固定路线直至走完.

而我认为我们的AI最聪明的地方,不是在于如何处理风,臭气,墙的情况. 而是, 这个路线的设计, 我们首先让agent一开始的方向即向右, 一直往前走,直至碰到风或者墙, 到这里, 我们走过一条直线, 这条直线就叫main, 这条main的上方平行的格子必定是安全的,因为在第一次碰到风或者墙之前,main里的格子全都不是安全的,没有风的,意味着,相邻的上方格子也都是可走的,于是以main上的每个格子为起点, 让agent往上的方向一直走,直至碰到风或者墙,这样走出的一条直线就叫part, 然后往下走返回这条part的起点,然后换下一个main上没走的过的点作为新的part的起点,重复搜索. 这样的模式直到main上的所有点都作为过一次part的起点结束(这个时候agent会回到起点,因为我们选择每条part起点的方式是从main离起点最远端往起点开始选择).

这整个一次搜索的路线我用一个形象的比喻,叫它梳子搜索.当然,这把梳子上的每一根小牙齿是长短不一的. 到这里,我们并没有得到200分.

于是我们让agent以开始方向为向上,再次展开了梳子搜索. 虽然这样的第二次搜索会走一些之前走过的重复格子,但是第二次梳子搜索也大大增加了之前搜索不到的地方,比如一下小牙齿的尽头的风的背后区域.

想像一下,当两把梳子的牙齿垂直交错时,实际上大部分格子已经被搜索到了.这样带来的收益远大于因为重叠导致的step cost.

事实上我们聪明的地方还在于,这样的算法, 再碰到风的时候,我们并不需要记录任何的memory去用作以后的判断,无论何时碰到风,我们都是180度转向,原路离开,而不是去尝试绕过去,因为对于绕过去之后的现象,太多未知情况需要去处理,那样需要记录所有格子的信息.而对于风之后的未涉及的区域我们并不担心我们会遗漏他,因为我们另一次梳子搜索的牙齿会替我们达到那里.

所以这种梳子算法,对于电脑本身而言,他是不需要很多memory的,他只需要记录一些flag.

**II. In about 1/4 page of text, describe problems you encountered and how you solved them.**

**当我们碰到一些需要掉头转向的格子,比如风,我们得进行两次左转才能进行下一次forward, 那么此时就需要一个flag来记录我们转的次数, 否则,再不知道转了几次的情况下, agent无法知道现在的朝向.**

**当agent第一次碰到臭气时,我们会让他射箭,如果没有scream,那就证明没有射死怪兽,这个时候他仍然站在臭气上无法移动, 为了解决这个问题,我们设置了一个叫justshooted的falg,让agent知道自己上一步是射箭,则这一步,他就可以往前移动了.**

**III. In about 1/4 page of text, provide suggestions for improving this project.**

**对自己作业的提高:**

**我觉得在每一条main的主干道伸出去的part次干道上, 可以让part的每一个走过的node为起点,再延伸出一条次次干道.**

**这是对project的提议:**

**测试1万次地图时需要先去world那个folder里make tournamentSet, 然后还得回到python shell 去run Main.py. 这只是一次操作. 当我们不确定我们分数范围的时候, 想测试30次 10000次的地图时,很麻烦.**

**能不能提高箭的数量**