Wumpus World Final AI Report

Team name - MindTheMinotaur

Member #1 (Smitha Balagurunatan/58328870) **Member #2 (Shikhar Sanstuti/47565791)**

I. Minimal AI (C++ Shell)

I.A. Briefly describe your Minimal AI algorithm:

Algorithm MinimalAI():
Input: A world of size N*N
Output: Agent walks a row of N*N and returns back by backtracking

```
Agent starts moving in a row
if glitters : grab , backTrackToStartPosition
        return climb
if bump or breeze or stench : backTrackToStartPosition
        return climb
```

I.B. Describe your Minimal AI algorithm's performance:

Cave Size	Sample size	Mean Score	Standard Deviation	99% Confidence Interval
4x4	10000	61.4352	248.799	61.4352 ± 6.4
5x5	10000	45.2724	219.124	45.272±5.645
6x6	10000	34.3776	195.994	34.378±5.049
7x7	10000	25.9325	175.308	25.933±4.516
Total Summary	40000	41.754	209.806	41.754±2.702

II. Draft AI (C++ Shell)

II.A. Briefly describe your Draft AI algorithm, focusing mainly on the changes since Minimal AI:

Algorithm DraftAI():
Input: A world of size N*N
Output: Agent walks the complete perimeter of N*N matrix in both clock and anti-clockwise direction.

```
Agent starts moving in row direction
if agent walking in a row or column direction do
    if glitters : grab , backTrackToStartPosition
        return climb
    if breeze or stench : backTrackToStartPosition
        if walked in row but not in column direction : startWalkingInColumnDirection
        else : return climb
    if bump : turn & move forward
if agent walked in row and column direction : return climb
```

II.B. Describe your Draft AI algorithm's performance:

Cave Size	Sample size	Mean Score	Standard Deviation	99% Confidence Interval
4x4	10000	160.286	370.964	160.286±9.556
5x5	10000	109.022	325.315	109.022±8.380
6x6	10000	75.0413	279.094	75.041±7.189
7x7	10000	55.9534	247.716	55.953±6.381
Total Summary	40000	100.076	305.772	100.076±3.938

III. Final AI (Python Shell)

III.A. Briefly describe your Final AI algorithm, focusing mainly on the changes since Draft AI:

Algorithm FinalAI():

Input: A world of size N*N

Output: Agent makes intelligent moves with help of DFS and knowledge base maintained. Wumpus shoot is also implemented.

```
if glitters : grab, backTrack()
        return CLIMB
if scream : unmarkSurroundingTilesDanger(current_position)
        disable STENCH
if stench and wumpus_not_shot : return SHOOT
if stench or breeze : markSurroundingTilesDanger()
        move back to previous safe spot                 // A knowledge base is maintained
if bump : current_position <- previous_position
current_position <- next_spot                           // Update current position and moves
current_direction <- next_spots direction
if next_spot not in traversed_path or safe_spots or visited array : add next_spot to it
        unmarkDangerTiles(next_spot)
        return next_spots[0]
return FORWARD
```

III.B. Describe your Final AI algorithm's performance:

Cave Size	Sample size	Mean Score	Standard Deviation	99% Confidence Interval
4x4	10000	280.5998	451.8116083501618	280.600±11.639
5x5	10000	223.891	423.4699294153009	223.891±10.909
6x6	10000	174.9467	392.4271744656707	174.947±10.109
7x7	10000	139.5479	364.8802032250996	139.548±9.399
Total Summary	40000	204.746	408.147	204.746±5.257

IV. In about 1/4 page of text or less, provide suggestions for improving this project.

We are executing DFS to mark tiles as safe or dangerous and using this as a knowledge base, we are deciding our next best move. Additionally, we are shooting the Wumpus when a stench is detected for the first time, which disables the stench and increases the probability to find and grab gold.

The attainable score can be improved by optimizing the backtracking logic back to the start location (0,0) once gold has been found or no more moves can be made. Implementing Breadth First Search (BFS) or Dijkstra's shortest path algorithm for this would help. The agent can then reach the start location (0,0) using a shorter path, than the originally traversed path, saving a few moves before reaching the start location (0,0) to climb out.

The present agent is timid and only approves an unexplored adjacent tile as safe if there is 0% probability of that having a wumpus/pit. Since it is mentioned that the probability of a pit containing tile is 20%, we can employ probabilistic reasoning to increase the instances of approved adjacent tiles. As a consequence, we increase the number of moves being made and increase the possibility of finding gold.