CS-171 Wumpus World Final AI Report

Team name BallisticDoritos

Member #1 (name/id) Sasha Mirabi (ID: 41073249) **Member #2 (name/id)** Zarin Tasnim Ohiba (ID: 43201605)

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

In our Minimal AI, we were initially making moves in the outer region of the board only. The agent could detect danger zone but only in the outer circle; and once it hit a bump, it used to go back the way it came. From there to the second submission which was the Draft AI, we changed the search algorithm to explore more safe zones in the interior of the board. The code used a mixture of depth-first search algorithm and breadth-first search algorithm to explore as much area as possible without making unnecessary moves. In our Final AI, instead of selecting any possible safe zones, the agent uses a heuristic function which always figures out the nearest safe zone at any given point and tries to move in that direction. The work of the heuristic function here is to simplify the constraints and it just calculates the distance between every point in the board (the points in the safe zones that can be reached but were not explored), and the current coordinate and picks the one which is closest to the current node. If there are more than one closest node, the heuristic selects the last point that was added to the list of safe zones because it means that the last point would be easier to go to as the path to it can also be easily figured from the last few moves. We also implemented a method that is similar to simulated annealing in a way that would help the agent get of an infinite loop or local maxima. If the agent ends up making the same move in a sequence, it realized that it needs to change its way and hence proceed around a different way in the board. Whenever the agent detects stench or breeze, it adds the neighbors to potential danger zone and they remain unsafe until the agent explores more area to conclude that some blocks in potential danger zone are in fact safe and it adds to the safe zone and remove from potential danger making it a smart agent.

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

The first problem we encountered was to make the agent move in a way so that it explored all the safe zones rather than just moving in the outer circle. We used a mixture of depth-first and breadth-first search to make the agent smart enough to explore all the possible moves. Next we noticed that the agent was making a lot of repetitive moves to get to a block due to depth-first search implementation. So, we used a heuristic function to help the agent decide on which is the closest block that would take the fewest moves to reach. This solved the issue of repetitive moves. The other problem we faced was that due to the implementation of search algorithm, our agent was getting stuck in an infinite loop (by going back and forth between few tiles) in some of the worlds. So, we then kept track of all the moves. Whenever we get a sequence where the coordinate appears more than 50 percent of the time, we set that coordinate to be a trap. So, the next time we have a move taking us to that coordinate, we ignore it and chooses a different path. This is how we solved the problems that we encountered.

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

Our project can be improved by implementing a full version of simulated annealing and implementing depth first branch and bound algorithm to make the agent super smart. We could have also used a little bit of randomness. (We tried using randomness but it gave sort of the same score so we omitted it).

In terms of improving the project course, I would say for each deadline ask the students to use a particular algorithm that we learn in class. For example for the first deadline we can do depth-first search which is not the most efficient way. For second deadline do breadth-first search plus depth first search. And then depth first branch and bound for the final deadline which is one of the most efficient ways to solve this Wumpus world problem.

This way students will get to "SEE" the difference in scores by implementing various algorithm, making it a hands on experience.