CS-171 Wumpus World Final AI Report

Team name SpaceCat

Member #1 (name/id) Bianca Tang 38478644 Member #2 (name/id or N/A) N/A

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

In order to make my AI smart, I had my AI keep track of all the nodes in the frontier, as well as any node that has been generated. To prevent the generation of nodes that don't actually exist on the game board, whenever my AI feels a bump, it replaces the value of the variables maxX and maxY, which represent the last column/row that is included in the board. When maxX or maxY are changed, the AI trims both node lists. When generating new nodes, it takes these two variables into account.

While my AI agent is in the cave, my AI keeps track of which tiles are possible pits and which tiles have the possibility of containing the Wumpus based on its past and current observations. If an agent walks into a tile, it knows that its current tile is safe, and changes the "Wumpus alert" and "pit alert" of those tiles to -1. If its current tile has a breeze or stench, it increases the "Wumpus alert" or "pit alert" of its adjacent tiles to 1 to show that there is the possibility of a pit or Wumpus in those tiles. The agent is not allowed to make this change if the tile was previously marked safe (-1) or if the tile was marked as a pit or Wumpus (2). If its current tile has no breeze or stench, it knows that its adjacent tiles are safe. If any tile ever ends up having all 4 of its adjacent tiles reaching an alert level of 1 for the Wumpus or pit, the agent assumes that that tile contains a Wumpus or pit and sets the alert level to 2 even though there are cases that exist where this is not the case. My agent is cautious.

When my agent smells a stench, it shoots an arrow. If there is a scream, then my agent ignores all Wumpus info in its next moves. If there is no scream, then my agent has least found one Wumpus-free path and updates its Wumpus levels accordingly.

To go back to the entrance once the gold has been picked up, I had my AI agent save the parent of each node upon creating new nodes. The parent of a node in this case, is the frontier node that was explored just before the creation of the new node. Backtracking is then easy, because all the AI must do is follow the parent pointer in each node back to the entrance node by turning in the correct direction, and moving forward when the node it points to matches the parent node of the node it is currently on.

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

At the beginning of the project, I wasn't sure how to have the agent go back to the entrance after they picked up the gold or ran out of frontier nodes to explore. At first, I tried to keep track of the movement actions I took using a stack (forward, turn right, turn left), and then popping off the actions to backtrack once I found the gold or ran out of nodes to explore. However, this seemed cumbersome and bulky, especially with larger map sizes. Instead, I decided to put a parent pointer on each node (each square in the cave) that represents the parent node that each node was discovered through. Then, when the agent finds the gold, it can follow the parent nodes back to the entrance. While this doesn't give the most optimal path back to the starting area, it gives a path that is always safe, and it doesn't need the amount of memory that a list of movement actions would require. An even better way to solve this, however, would be to use A* with my own heuristic to estimate distance, while taking the possibility of pits and wumpuses, and the estimated number of turns needed to traverse a certain path into account.

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

While I ended up meeting the point requirement, there are many ways that I could improve my AI agent. One thing I could do is implement my own heuristic to search for the gold with A*. I could use the same heuristic to backtrack after the gold has been found. I could also introduce probability into my agent by giving it a chance to accept a risky move, like walking into an unexplored tile that is next to a breeze.

To improve the project itself, it would be nice if, at the end of the year after the final submission deadline has passed, the professor or TA could go over how they would go about implementing the AI and what questions they would ask themselves about implementation while they're tackling the problem. Alternatively, they could take submissions from previous years that have different implementations and go over how those AIs were implemented, what they could have done to improve their agent, and what they ultimately scored. It would be nice to see what we did right, what we could have improved on, and how other students tackled the problem.