Minesweeper Final AI Report

Team name: mineYourStep
Member #1 (name/id): Yu-Chi Lin / yuchil9@uci.edu / 56426817
Member #2 (name/id): Mason Read / mcread@uci.edu / 56764926

I. Minimal AI

I.A. Briefly describe your Minimal AI algorithm. What did you do that was fun, clever, or creative?

Our minimal AI works mostly on the rules of thumb. One of the first things it checks for are tiles labeled as 0. When a tile has no mines in the vicinity, all of the surrounding tiles can be safely discovered, and thus are added to our Frontier Queue. Another basic rule of thumb is that if there is a tile with 1 mine nearby, we can be guaranteed that that is the only mine in the world, as thus every tile not within it's vicinity is safe to be discovered. Finally, when there is a tile with 1 mine it it's vicinity, and has only one undiscovered tile, all the others of which are discovered and not flagged, we can know that the mine is in the undiscovered tile. Similarly, if there is a tile with one mine in its vicinity that has one flagged tile, all other tiles around it are safe to discover. Using these simple rules of thumb, our AI is able to successfully complete all of the 5x5 worlds (or a world of any size) with only one mine. However, given we only used very simple rules of thumb, our AI at this stage would mostly unable to complete any other world with more than one mine, due to a lack of more complex reasoning about mines and probabilities of covered tiles containing mines.

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

Board Size	Sample Size	Score	Worlds Complete
5x5	1000	1000	1000
8x8	10	0	0
16x16	10	0	0
16x30	10	0	0
Total Summary	1030	1000	1000

II. Final AI

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

The main difference between the Final AI and the Minimal AI is the implementation of backtracking search. Although the minimal AI used only basic rules of thumbs to complete all 1 mine worlds, the Final AI needed to use much more complicated logic to complete more complex worlds. The Final AI solves these complex worlds by enumerating every possible combination of the covered nodes on the frontier once the rules of thumb are no longer useful. To generate each combination, we represented the frontier as a bitstring, (0 for no mine, 1 for a mine) and the generation of it was done recursively. To save time, the branch is pruned as soon as an invalid combination in the bitstring is detected. In addition to this, to stay under the time limit, nothing will be added to the frontier set once it's size exceeds 19 tiles. Between the basic rules of thumbs perfected with the Minimal AI and the backtracking search, the Final AI is able to perform quite well on all worlds.

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

Board Size	Sample Size	Score	Worlds Complete
5x5	1000	1000	1000
8x8	1000	836	843
16x16	1000	1648	789
16x30	1000	1164	303
Total Summary	4000	4648	3330

III. In about 1/4 page of text or less, provide suggestions for improving this project (*this section does __NOT__ count as past of your two-page total limit.*)

There are a few improvements that can be made to improve this project, in my opinion. One of the first major roadblocks we faced was the discrepancy between standard indexing on lists in programming and the standard used to communicate with the agent. We found we needed to reverse agentX and agentY when indexing the saved board. A little more clarity as to how to manage this would have been helpful. I think it also would have been helpful to have a couple more classes to tell us how to get started and to give us a couple more ideas.