Minesweeper Final AI Report

Team name_WuhanMk1____________________________________

Member #1 (name/id)_Yuhao Song 30153928_________________________

Member #2 (name/id)_Jiefu Ling 25008346__________________________

I. Minimal AI

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

The primary goal of Minimal AI is to complete Minesweeper game on a 5 * 5 board with 1 mine. Which means the hint of tiles will only be either 0 or 1. Therefore, we need to implant the function that can reach the frontier (frontier refers to all tiles with hint not equal to 0). The basic idea of this function is to traverse every tile (normally 8 tiles) around current tile, when the current tile returns a hint of 0. The traversal stops until it reaches a tile that does not returns a hint of 0, and finally all tiles with hint 1 will be found. After all hint tiles were found, there will be two situations: 1. The mine will be found and flagged 2. Two tiles remain covered, and we don't know which one is the mine.

When encountered situation 2, we traverse all tiles with hint 1, and found the tile that has exactly one covered tile in all its neighbour tiles. That covered tile must be the mine, and therefore, the game complete.

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

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

P.s: Game sizes other than 5 * 5 were not tested.

II. Final AI

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

After the implementation of the “reach the frontier” function, Draft AI and Final AI require the AI to complete three types of games: an 8 * 8 board with 10 mines, a 16 * 16 board with 40 mines, and a 16 * 30 board with 99 mines. Therefore, the hint of tiles can be numbers other than 0 or 1. The algorithm to solve intermediate game and expert game also require the “reach the frontier” function, because all difficulties need to eliminate tiles with hint 0 and clear the field for the next step. To find the mines around tiles with hint other than 0, we need to run constraint satisfactions to test every tile around current tile, normally 8 tiles. By importing multiple constraint satisfaction lists (cs, cs1, cs2, etc.), we make coordinates of mines more precise among a range of coordinates. Then, we calculate if the number of suspected tiles matches all hints provided by tiles around it. If the number of suspected tiles, which are covered, matches all hint from neighbour tiles, that tile must be a mine. The algorithm will traverse all tiles possible until all mines are being flagged or uncover a mine (fail the game).

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

Board Size	Sample Size	Score	Worlds Complete
5x5	N/A	N/A	N/A
8x8	1000	590	590
16x16	1000	1086	543
16x30	1000	0	0
Total Summary	3000	1676	1133

P.s: 16 * 30 sized game takes excessive amount of time when testing so it did not return any result.

III. In about 1/4 page of text or less, provide suggestions for improving this project (*this section does <u>NOT</u> count as past of your two-page total limit.*)

Error Message “Invalid Action” is ambiguous, not specific on where or what the error is. Also, when the program encountered an “Invalid Action”, it will run infinitely instead of break from the program. Therefore, the meaningless message “Invalid Action” will fill full screen, but useful information (printed by our code) will be pushed away and very hard to find.

Also, we think the method of making moves (UNCOVER, FLAG, UNFLAG, LEAVE) can have a little adjustment. Instead of returning an action, the action can be called by a function. That will make us better running and debugging this project.