HW 8: Portfolio Project

Jeremy Tsang

December 7, 2020

The Problem

In Minesweeper a player is given board of m rows and n columns with $k < m \cdot n$ mines. The goal is for the player to open all cells that do not contain a mine.

Decision Question

From Kaye (Springer-Verlag New York, Volume 22, Number 2, 2000) the decision problem is: "Given an arbitrary set of mines and numbers on a rectangular grid, can mines be placed consistently, following the usual minesweeper rules?"

This problem is NP-complete. A proof by Kaye is given in the aforementioned paper where he goes on to show that SAT can reduce to the Minesweeper problem. He goes to show that one can construct the basic logic gates AND, OR, and NOT (and therefore an entire computer) using regions of a Minesweeper board to model a wire carrying voltage. This can be done by assigning a "direction" to the region where the unopened cells are the digital logical inputs. Hence, by strategically placing the unopened cells it is possible recover the inputs of SAT.

The Verification Algorithm of User's Solution

Store the mines in a 2D array with m rows and n columns. Consider the player's moves as single bit string (certificates) of length $m \cdot n$ where bit i represent the cell at row $\lfloor i/n \rfloor$ and column $i\%m$ we can easily check whether or not a player has "exploded" by simply checking the cell at.

```
# bit_string: string of bool. Length m*n
# mines: 2D array of bool with height m and width n
# m: int height of the minesweeper board
# n: int width of the minesweeper board
verification(bit_string, mines, m, n):
    for i in len(bit_string):
        if bit_string[i] == mines[i / n][i % m]:
            return False
    return True
```

This simple brute force algorithm has a single `for` loop which has at most $m \cdot n$ iterations. Hence it's runtime is $\mathcal{O}(m \cdot n)$ which is polynomial in the number of cells in the board.