Sunny Cheng, Maxwell Wang
22 Mar 2021

Minesweeper

Professor Cowan
CS 440

Intro

Group members: Sunny Cheng (sc2040) and Maxwell Wang (mlw195)
Group contribution: Each group member contributed equally. Maxwell did the minesweeper board implementation as well as the basic agent. Sunny did the improved agent and the bonus. We worked together to develop the algorithm.
Group statement: All work here is our own, and we did not copy or take code online or from other students.

Representation

We represented the minesweeper board as a 2D list of characters:

- ? = unsearched
- X = uncovered mine
- 0-8 = uncovered clue
- M = flagged as mine
- C = flagged as clear

We realized that solving a minesweeper board is just solving a system of equations, so we decided to use a matrix containing these equations as our knowledge base. Each time a new clue is uncovered, the equation representing that clue is represented as a new row in our matrix. Then, converting the matrix to RREF will represent the inferred relationships between the clues.

Inference

Our algorithm is called DISARM: Defensive Intelligent Super Agent Removing Mines.

To perform inference, we decided to represent our knowledge base as a system of equations. For every clue (a cell with a number), we could rewrite that as an equation. To do this, we represent each cell (i, j) on the board as some a_k where $k = i \cdot dim + j$ (this will make it easier to convert between coordinates and equations). If a cell has a mine, then its value is 1. If it is safe, it has value 0. So for some clue value n with surrounding hidden cells $a_{k_1}, ..., a_{k_i}$, we write that as the equation $\sum_{j=1}^{i} a_{k_j} = n$. For every board, we will iterate over all the cells and generate equations for every cell with a clue. Then, once we have collected all the clues, we will attempt to solve for new information.

To do this, we combine all the rows of equations into a single large matrix. We then perform operations to transform it into reduced row echelon form. This firstly ensures that for each leading 1 in each row, there are no 1s under it. This is very helpful for us, as it represents the combination of different clues. If two rows both have a leading 1 in the same column, RREF will combine them for us by subtracting the first row from the second. Secondly, it ensures there are no 1s above each leading 1. This is also a combination of clues, as it will ensure if any two equations share information, they are combined.

Once we have the matrix in RREF form, we can take the rows of the matrix and convert each row back into an equation, and then convert the equation into new information. To do this, we use the same method as the basic agent: if the new clue value is the same as the number of hidden cells, each hidden cell must be a mine. If the new clue value is zero, all hidden cells are safe. We collect all of the new information and update the board. We can keep doing this until there is no new information, and then we will have to randomly guess a cell to pick.

For an example, see the play-by-play section.

Decisions

When we are performing inference, we search every safe cell on the board. However, if we cannot perform any more inference to reveal safe cells, then we will randomly select a cell on the board to guess. We have developed an optimized selection algorithm for the bonus. When we perform inference, we try to solve some system of equations. However, if that fails, then we know there must be some unknowns in our system of equations. We choose to select the unknown cell in the maximum number of equations to reveal. This ensures that we are doing the most possible to add new information to our knowledge base. Even if the cell is a mine, it will enable us to solve many more equations within our knowledge base and generate the most amount of new information.

Play by Play

After watching an entire game, there were no places my program made a move we did not agree with. However, there were places where the program made a move that surprised me. We will examine that particular move here. For the entire play by play, please see the appendix.

On the fourth move, using inference, the agent was able to make this particular move on the board's top right corner and determine that two cells were safe:

?	?	?	?
1	1	3	?
0	0	1	?
0	0	1	?

$\rightarrow$

?	?	2	?
1	1	3	2
0	0	1	?
0	0	1	?

Let's consider how it was able to determine that. First, we will turn all the clues into equations. We see the board as:

a_1	a_2	a_3	a_4
1	1	3	a_5
0	0	1	a_6
0	0	1	a_7

And so we have clues:

$$\begin{aligned} a_1 + a_2 &= 1 \\ a_1 + a_2 + a_3 &= 1 \\ a_2 + a_3 + a_4 + a_5 + a_6 &= 3 \\ a_5 + a_6 + a_7 &= 1 \\ a_6 + a_7 &= 1 \end{aligned}$$

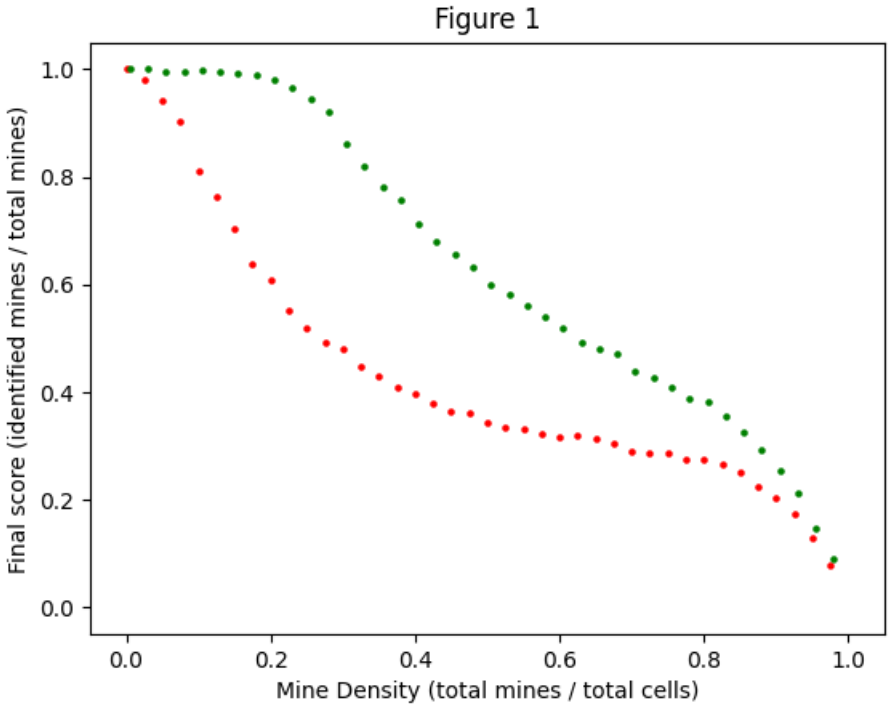
This is the matrix and after RREF we get

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 3 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 1 & 0 & 1 & 0 & 0 & -1 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Reading the 3rd and 4th rows, they clearly say $a_3 = 0$ and $a_5 = 0$, representing those cells as safe. This would not have been possible using the basic agent, and is an example of intelligence by the improved agent using relationships between clues to reveal extra information.

Performance

All of the following plots are done on a 30 by 30 grid. There are 40 densities, and at each density, 20 games were run and the average final score was plotted. The following is a plot of mine density vs final score. Red is the basic agent and green is the improved agent:



This does agree with our intuition, as the improved agent should be beating the basic agent but also would not be perfect. Additionally, both agents have higher final scores with low density, and decrease as the density increases. There does not seem to be an exact cutoff for when minesweeper becomes hard like the maze. However, for the basic agent, the final score decreases rapidly and is under 50% by a mine density of 0.3, where it then slowly decreases. The improved agent does not drastically fall until mine density 0.3 as well, but as it passes 0.3 it starts to slowly descend as well. The improved agent beats the simple agent almost all the time. It beats it by the widest margin near 0.3, and the margin decreases as the mine density increases. This makes sense, since as the density increases, it becomes more and more of a guessing game rather than how well one can figure out things from clues. The improved agent is very often able to deduce new information as can be seen by the wide gap between the final scores of the agents.

Efficiency

We primarily ran into time constraints rather than space constraints. It was definitely an implementation specific constraint. Problem-wise, minesweeper is not much more different than the fire maze as they are both grids. However, our minesweeper boards were an order of magnitude smaller than the maze sizes. This is due to our implementation, specifically, the RREF part of our matrix. To RREF, each of the m rows may perform at most m row operations, and each row operation takes m time, leading to an m^3 run time. However, in this case $m = n^2$ as there are n^2 cells, so RREF could take as long as n^6, which is very bad, while size only grows as n^2 where n is the dimension of the board.

We have implemented several optimizations to speed up the algorithm.

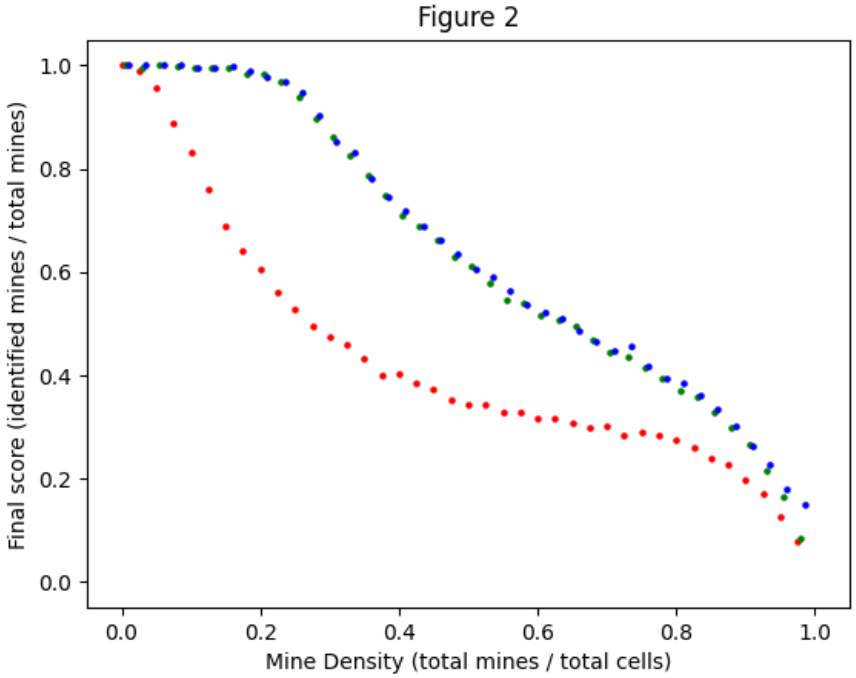
1. If a clue's neighbors are completely revealed, do not add it to our knowledge base.
2. When solving the matrix, remove all columns with only 0s.
3. Only use the inference algorithm when the basic algorithm fails.

1 obviously makes sense as a clue will not help us if we already know everything about the cells around it. 2 speeds up the RREF portion of the algorithm, as it is very possible out of n^2 cells there are only a handful that we get information about clues from, so we do not need the extra columns if they give us no information. 3 enables us to use RREF less times by ensuring that only when the basic agent fails that we turn to the improved agent to find more information. As the basic agent is fairly fast, this improves run time by a lot.

We could further improve it by isolating the clues. If two sets of clues do not share any common cell, then there is no need to combine then in the same matrix. They could be put in separate matrices and solved separately, This would improve the speed of RREFing the matrix and improve performance.

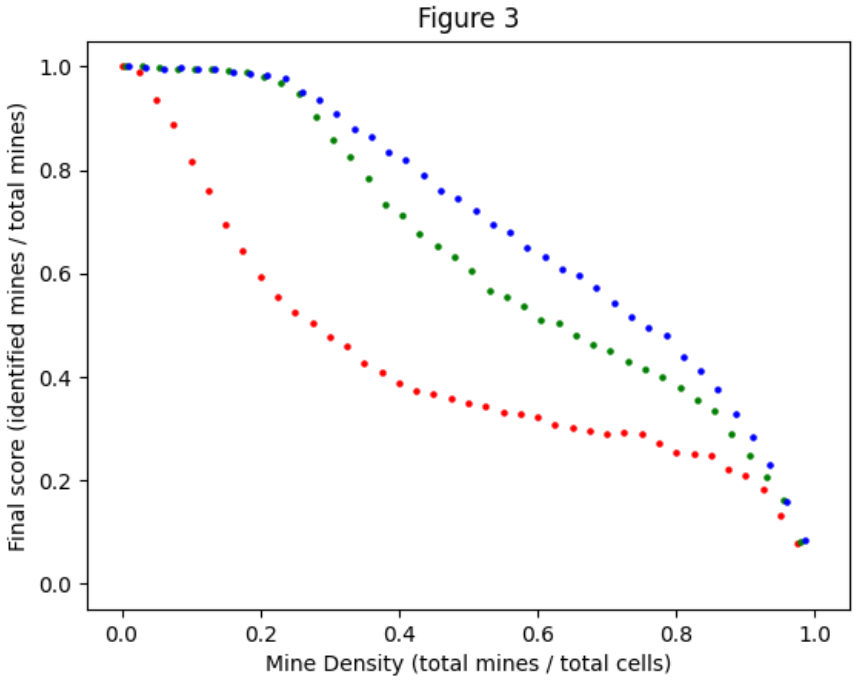
Bonus

Global information. If we are given the number of mines on the board, we can easily add this to our knowledge base. Essentially, this is the clue where the value is the number of mines, and every cell on the board is a neighbor of this. So, the equation would be $\sum a = n$ for all hidden cells a and for the number of mines n. We simply add this row to our matrix and solve similarly. With this information, our plot becomes the following (Red = basic, Green = Improved, Blue = Improved with global information):



We see that the global information does not seem to have a clear benefit to the agent. This is because the added information rarely comes into play for the improved agent. In one case, we find all mines and using the global info we would know the remaining cells are safe, but in this case even without the information we would have not hit any more mines. In another case, when the number of hidden cells left is equal to the number of mines, we would be able to instantly mark all cells as mines. However, for this case we can generally already can solve it with inference. We only see a small improvement in the last data point, likely from the agent guessing all of the non-mined cells and then automatically determining the rest of the cells as mines.

Better decisions. We have developed an optimized selection algorithm for the bonus. When we perform inference, we try to solve some system of equations. However, if that fails, then we know there must be some unknowns in our system of equations. We choose to select the unknown cell in the maximum number of equations to reveal. This ensures that we are doing the most possible to add new information to our knowledge base. Even if the cell is a mine, it will enable us to solve many more equations within our knowledge base and generate the most amount of new information, so that there is a smaller chance we will have to resort to guessing again. With this information, our plot becomes the following (Red = basic, Green = Improved, Blue = Improved with better decisions):



We see that the selection decisions primarily helps the improved agent starting at around 0.3 density. This makes sense as if the density is low, it wouldn't be so bad to pick a random cell as there are not many mines. Additionally there may not be too many constraints to help solve, so the cell we reveal may not benefit us much. Thus in the 0.5 to 0.7 density area, the better selection decision seems to be most helpful as if one guesses more often than not one will hit a mine, and while that doesn't change with our selection, we will also have a better chance of revealing more clues. Overall, it is clear that the better selection algorithm clearly improves the agent.

Appendix

The following is a play-by-play of a 10 by 10 board with 10 mines by our agent.

```
Using inference
Clues
Mines:
Safe:
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
Forced to guess: 9 0
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
0 ? ? ? ? ? ? ? ? ?
Clues
Mines:
Safe: (9, 1) (8, 0) (8, 1)
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
0 0 ? ? ? ? ? ? ? ?
0 0 ? ? ? ? ? ? ? ?
Clues
Mines:
Safe: (7, 1) (7, 2) (8, 2) (7, 0) (9, 2)
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
1 1 3 ? ? ? ? ? ? ?
0 0 1 ? ? ? ? ? ? ?
0 0 1 ? ? ? ? ? ? ?
```

```
Using inference
Clues
Mines:
Safe: (6, 2) (7, 3)
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? 2 ? ? ? ? ? ? ?
1 1 3 2 ? ? ? ? ? ?
0 0 1 ? ? ? ? ? ? ?
0 0 1 ? ? ? ? ? ? ?
Clues
Mines: (6, 1) (8, 3) (6, 3)
Safe:
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? M 2 M ? ? ? ? ? ?
1 1 3 2 ? ? ? ? ? ?
0 0 1 M ? ? ? ? ? ?
0 0 1 ? ? ? ? ? ? ?
Clues
Mines:
Safe: (7, 4) (8, 4) (9, 3) (6, 4) (5, 1) (6, 0) (5, 3) (5, 2)
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? 1 2 1 ? ? ? ? ? ?
1 M 2 M 2 ? ? ? ? ?
1 1 3 2 3 ? ? ? ? ?
0 0 1 M 1 ? ? ? ? ?
0 0 1 1 ? ? ? ? ? ?
Clues
Mines:
Safe: (4, 4) (4, 0) (4, 3) (5, 4) (4, 2) (9, 5) (5, 0) (7, 5) (4, 1) (8, 5) (9, 4)
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
1 1 1 0 0 ? ? ? ? ?
1 1 2 1 2 ? ? ? ? ?
1 M 2 M 2 ? ? ? ? ?
1 1 3 2 3 2 ? ? ? ?
0 0 1 M 1 2 ? ? ? ?
0 0 1 1 1 1 ? ? ? ?
```

```
Clues
Mines: (6, 5)
Safe: (4, 5) (5, 5) (3, 3) (3, 4) (3, 2) (3, 5)
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? 2 0 0 1 ? ? ? ?
1 1 1 0 0 0 ? ? ? ?
1 1 2 1 2 1 ? ? ? ?
1 M 2 M 2 M ? ? ? ?
1 1 3 2 3 2 ? ? ? ?
0 0 1 M 1 2 ? ? ? ?
0 0 1 1 1 1 ? ? ? ?
Clues
Mines: (3, 1)
Safe: (2, 4) (2, 2) (4, 6) (2, 3) (5, 6) (3, 6) (6, 6) (2, 5)
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? 2 0 0 1 ? ? ? ?
? M 2 0 0 1 1 ? ? ?
1 1 1 0 0 0 0 ? ? ?
1 1 2 1 2 1 1 ? ? ?
1 M 2 M 2 M 2 ? ? ?
1 1 3 2 3 2 ? ? ? ?
0 0 1 M 1 2 ? ? ? ?
0 0 1 1 1 1 ? ? ? ?
Clues
Mines: (2, 6) (2, 1)
Safe: (1, 2) (1, 5) (3, 7) (5, 7) (1, 4) (3, 0) (6, 7) (1, 3) (4, 7)
? ? ? ? ? ? ? ? ? ?
? ? 1 0 0 1 ? ? ? ?
? M 2 0 0 1 M ? ? ?
2 M 2 0 0 1 1 1 ? ?
1 1 1 0 0 0 0 0 ? ?
1 1 2 1 2 1 1 0 ? ?
1 M 2 M 2 M 2 1 ? ?
1 1 3 2 3 2 ? ? ? ?
0 0 1 M 1 2 ? ? ? ?
0 0 1 1 1 1 ? ? ? ?
Clues
Mines:
Safe: (0, 1) (3, 8) (0, 4) (2, 7) (5, 8) (1, 1) (0, 3) (2, 0) (6, 8) (0, 6) (0, 2) (0, 5) (4, 8) (1, 6) (2, 8)
? 0 0 0 0 0 1 ? ? ?
? 1 1 0 0 1 2 ? ? ?
2 M 2 0 0 1 M 2 1 ?
2 M 2 0 0 1 1 1 0 ?
1 1 1 0 0 0 0 0 0 ?
1 1 2 1 2 1 1 0 0 ?
1 M 2 M 2 M 2 1 0 ?
1 1 3 2 3 2 ? ? ? ?
0 0 1 M 1 2 ? ? ? ?
0 0 1 1 1 1 ? ? ? ?
```

```
Clues
Mines:
Safe: (7, 7) (0, 0) (4, 9) (7, 9) (2, 9) (3, 9) (1, 0) (5, 9) (6, 9) (7, 8)
0 0 0 0 0 0 1 ? ? ?
1 1 1 0 0 1 2 ? ? ?
2 M 2 0 0 1 M 2 1 0
2 M 2 0 0 1 1 1 0 0
1 1 1 0 0 0 0 0 0 0
1 1 2 1 2 1 1 0 0 0
1 M 2 M 2 M 2 1 0 0
1 1 3 2 3 2 ? 1 0 0
0 0 1 M 1 2 ? ? ? ?
0 0 1 1 1 1 ? ? ? ?
Clues
Mines: (7, 6)
Safe: (8, 8) (8, 9) (8, 7) (1, 8) (1, 9)
0 0 0 0 0 0 1 ? ? ?
1 1 1 0 0 1 2 ? 1 0
2 M 2 0 0 1 M 2 1 0
2 M 2 0 0 1 1 1 0 0
1 1 1 0 0 0 0 0 0 0
1 1 2 1 2 1 1 0 0 0
1 M 2 M 2 M 2 1 0 0
1 1 3 2 3 2 M 1 0 0
0 0 1 M 1 2 ? 2 0 0
0 0 1 1 1 1 ? ? ? ?
Clues
Mines: (1, 7)
Safe: (9, 7) (9, 8) (8, 6) (9, 9) (0, 8) (0, 9)
0 0 0 0 0 0 1 ? 1 0
1 1 1 0 0 1 2 M 1 0
2 M 2 0 0 1 M 2 1 0
2 M 2 0 0 1 1 1 0 0
1 1 1 0 0 0 0 0 0 0
1 1 2 1 2 1 1 0 0 0
1 M 2 M 2 M 2 1 0 0
1 1 3 2 3 2 M 1 0 0
0 0 1 M 1 2 2 2 0 0
0 0 1 1 1 1 ? 1 0 0
Clues
Mines: (9, 6)
Safe: (0, 7)
0 0 0 0 0 0 1 1 1 0
1 1 1 0 0 1 2 M 1 0
2 M 2 0 0 1 M 2 1 0
2 M 2 0 0 1 1 1 0 0
1 1 1 0 0 0 0 0 0 0
1 1 2 1 2 1 1 0 0 0
1 M 2 M 2 M 2 1 0 0
1 1 3 2 3 2 M 1 0 0
0 0 1 M 1 2 2 2 0 0
0 0 1 1 1 1 M 1 0 0
```