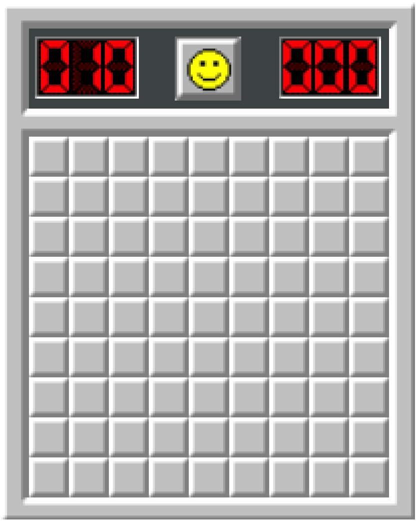
A Detailed Guide to Minesweeper

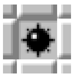
February 2023

TL;DR if you're new

This is minesweeper



This is a mine. They occupy some spaces of the board.



Play by clicking squares. Numbers say how many of the neighboring squares (including diagonals) have mines.

Win by uncovering all squares without mines.

DO NOT READ ANY FURTHER

I strongly encourage you to play and learn the game yourself. IMO this is the best way to have fun playing minesweeper.

Contents

1 Introduction

Who am I?

I am not among the elites of minesweeper. As of writing, my best time on expert difficulty is 80 seconds[1], leaving me much room to improve. I do, however, love playing this game.

I discovered minesweeper in 3rd grade. My parents got me an old IBM ThinkPad 20M to do essays and nothing else. For gaming, I was restricted to the 4 games that came with Windows 98: Solitaire, Freecell, Hearts, and Minesweeper. At the time, I barely understood the rules or strategy, and I was not allowed free time on the internet to learn either. I got more into minesweeper towards the end of high school, when I inherited the family laptop nobody bothered to use because the case was so broken beyond repair. Unable to effectively play any other games on it, I turned to playing the built in Windows 7 minesweeper to pass time. Being more mature, I actually sat down and learned the game. Using the laptop trackpad, I was able to clear the expert difficulty on a regular basis. During college, I discovered minesweeper.online, and started to actually try to get better at the game. Even as I got to build my own PC capable of playing higher fidelity games, minesweeper was a game I'd always play during the frequent gaps I procrastinated on assignments.

Requiring fast-paced logic balanced with quick risk assessment, playing minesweeper has become one of my favorite past-times. It is my hope that more people will be able to enjoy this game as much as I do.

What is this?

This document is an unecessarily long guide on minesweeper. As a graduate student in engineering, I find myself typesetting a lot of homework and papers in LaTeX, so I thought it'd be fun to put my thoughts on minesweeper in a similar format. There are already a fair number of papers exploring the math of minesweeper[2], however the math I present in this document in completely in the service of the player, aiming to root the decisions that can be made in minesweeper within a mathematical framework. My goal is to assemble and formalize everything I've learned about minesweeper, from the patterns beginners learn to the guessing strategies I've developed based on computer simulations, into a single document in the most long-winded way possible.

Who is this for?

Mostly myself. However, I'd be overjoyed if anyone else read over this document and either learned something or otherwise found it enjoyable. I aim to break down the game to the most basic level, so players of any skill level will be able to take away something from this document.

How is this document structured?

Section 2 starts out with the very basics that are typically picked up within the first days of playing. For players already familiar with the game, no new information will likely be learned here. Section 3 dives deeper into the clicks that you can make without guessing. In this section, I attempt to formalize much of the logic used to solve the more advanced patterns. Section 4 looks at guessing, and if possible, the optimal strategies for guessing. I will attempt to approach the problem of guessing from both a mathematical approach, and a computer-aided approach (TODO). Section 5 takes a side step to look at efficiency, which is given by solving a board in the least amount of click possible. Finally, section 6 looks at various goals of playing minesweeper and how strategies may differ between each goal.

[1] My minesweeper.online profile: https://minesweeper.online/player/1896875

[2] https://minesweepergame.com/math-papers.php

Acknowledgements

Shout out to the kind people at minesweeper online (although I've never really interacted with them), and their great collection of guides[3].

[3] https://minesweeper.online/help/guides

2 Basics

Minesweeper may seem intimidating for a beginner. This section is a gentle introduction to the mechanics of the game, followed by a simple set of tools to allow beginners to get their first wins. To remain beginner friendly, math will be kept at a minimum in this section. By the end of this section readers should be able to clear intermediate level boards with confidence in their decisions, and be able to tackle expert boards with some success.

2.1 How to Play

Controls

Controls are largely dependent on the platform you decide to play minesweeper on. The following are universal actions and the keybinds I tend to find associated with them when playing with mouse and keyboard.

- **Start a new game**: Click the smiley face, F2 (also Spacebar for minesweeper.online)
- **Clear**: Left-Click
- **Flag/Unflag**: Right-Click
- **Chording**: Left+Right-Click, Middle-Click (also Left-Click for minesweeper.online)

Note: For some versions, right clicking will cycle through a flag, a ? question mark, and an unmarked square. If able, I recommend turning off "Marks (?)" in the settings, as question marks don't serve too much utility while playing.

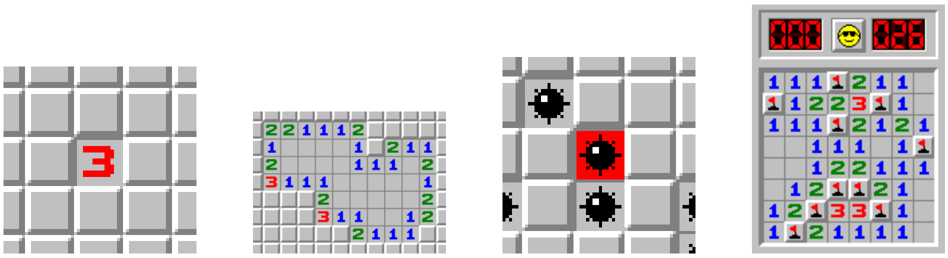
Clearing Squares

Left-clicking on a square is a commitment. You are declaring, "I swear on my life that there is no mine on this square." Should you left-click on a square containing a mine.
Four things can happen when you clear a square:

1. There is no mine in the square, and a number appears. This number indicates the number of mines in the eight neighboring squares (or less if the square is on an edge).
2. There is no mine in the square, and many numbers appear. This is called an **opening**. An opening occurs when there are no mines in the square you click, as well as the squares neighboring the square you click. This is an effective "0" number, but games typically leave the square blank, and automatically clears all the squares around the zeros recursively until no more squares can be cleared.
3. You click a mine. You are dead.
4. You've cleared the last square that's not a mine. You've won the game

The First Click

The game starts when you clear the first square. The first left-click you make is guaranteed to never be a mine. Leaving the details for future sections (Section 4.4), basic rule of thumb is to **choose a corner for your first click**. The idea is to hope for an opening on your first click that will minimize the amount of needed guessing. Hand waving a lot, corners are a good choice because it only has 3 neighbors, giving it the high odd of having no neighboring mines, leading to an opening. If the first corner does not lead to a sufficient opening, clicking on other corners is also an okay idea (more on this later).



(a) Single number (b) Opening (c) Mine (d) Win (Note the sunglasses on the smiley face

Figure 1: The four possible scenarios when clearing a square

Use Flags to Mark Where You Think Mines Are

Flags can be placed on the grid by right-clicking a square. This will decrement the mine counter. Flags can still be placed on squares without a mine, and this will still decrease the mine counter. Flags do NOT verify that a mine is at a given square. They are simply a useful tool the player can use to mark possible mine locations.

Flagging is more than just a visual aid. They can be used for chording (discussed below), minecounting (Section 3.4), and for preventing possible misclicks onto the mine. For beginners and anyone not playing for speed, I highly recommend to **place a flag at every location you know there is a mine**.

Do note that there exists a "No Flag" playstyle where players intentionally play without flags. This can improve speed in some cases since you are not using extraneous clicks to flag obvious mines. In section 5, we'll find that a hybrid approach where only some mines are flagged may actually be the fastest strategy. In my opinion, for higher difficulties expert and above, not flagging at all is psychotic, but if this is you, you do you.

Chording

Chording is a technique where you click a number with the number's number of flags adjacent to it (we'll call this a **chordable** number). This clears all the squares around the number you click (and if a surrounding square is an opening, a recursive clearing is performed).

Chording can be normally be performed with either a middle-click, or by pressing left and right-click at the same time. As chording typically follows placing a flag, you can keep holding the right-click from placing the flag, and left-click the fulfilled number next to the flag to chord. However, when the platform allows, sometimes just left-clicking the fulfilled number will also chord.

If used correctly, chording can greatly reduce the number of clicks you use. Like flagging, chording all the time is usually not optimal for speed. However, advanced players can use chording to speed up clears. For beginners, chording is hardly necessary to know, but I personally used chording a lot when learning. Chording reduces the number of raw left-clicks you perform on the unknown squares, reducing possible misclick encouters.

2.2 Basic Patterns

So how do you know which squares to clear and which squares to flag? While I encourage beginners to exercise basic logic to figure out which squares deserve which click, this section will provide a quick cheat-sheet of common simple patterns that occur in play. These patterns are ordered from most simple/common, to the slightly more obscure. This section will not contain all commonly known patterns, but these are the patterns I personally think warrant memorization for a beginner.

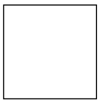
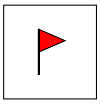
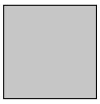
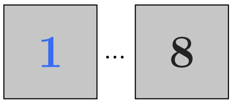
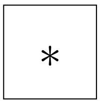
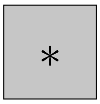
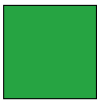
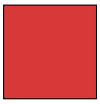
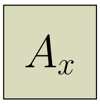
	Unknown square: May or may not contain a mine
	Known mine
	Zero square (opening): Does not contain a mine and adjacent squares do not contain a mine
	Numbered square: Does not contain a mine and # adjacent squares contain a mine
	Unknown Don't Care (UDC) square: May contain a mine, but we don't care if it contains a mine, or what numerical value it has if it doesn't contain a mine
	Don't Care (DC) square: Does not contain a mine, but we don't care about its numerical value
	Safe square: It can be inferred that there are no mines here; should be cleared
	Unsafe square: It can be inferred that there is a mine here; can be flagged
	Mutually Shared Mine (MSM) squares: All squares containing the same letter share a possibility of having x number of mines. The existence of x mines within these squares excludes the possibility of the others containing a mine

Table 1: Diagram Key

Section 3 will go over these patterns (and more) in more detail. For now, these patterns can be taken for granted, and will typically be enough to solve nearly all intermediate boards and some expert boards.

Before discussing these patterns, a key for the figures used will have to be introduced. A key for the diagrams used in this document is shown in Table 1 Most of these will be identical the the notation used in game. The first six rows in the table will depict the state of the board. Green cells and red cells indicate areas where conclusions can be drawn hence can be either cleared (green) or flagged (red). The last row describes cells where it is unknown if a mine exists in any individual square, but it is known that mines exists among the squares marked by the same letter.

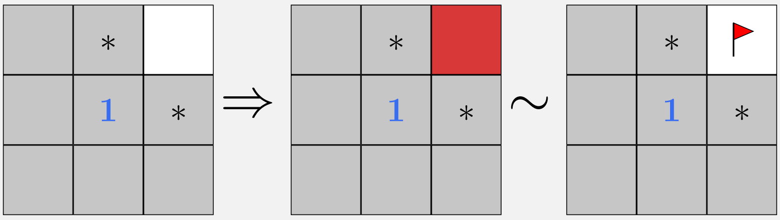
Board diagrams show only a subset of a board, and are assumed to extend infinitely in all directions. Squares beyond the edges of the board can be imagined as don't care (DC) squares.

2.2.1 Counting Patterns

All Mines

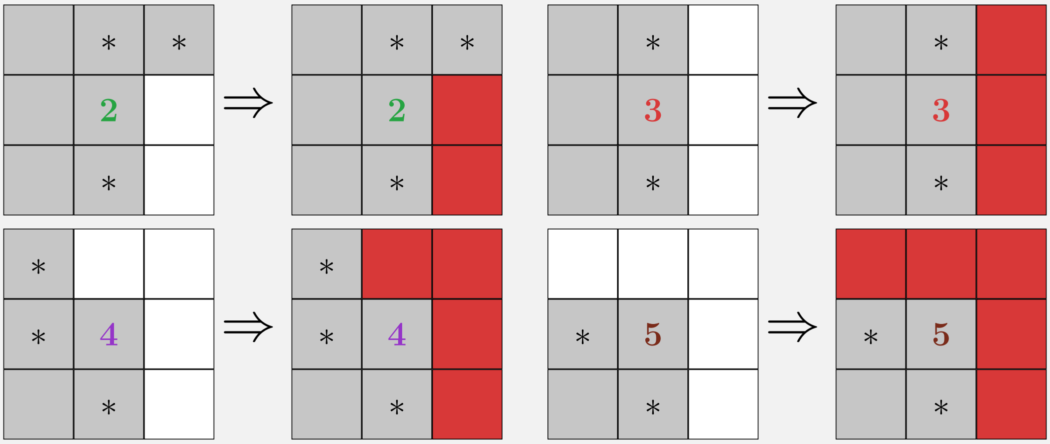
When the number of unknown spaces surround a number equals that number, one can infer that all those spaces must be mines.

Example 2.1: 1-Corner



This is a classical situation where the location of a mine can easily be determined.

Example 2.2: Common "All Mines" Patterns



These are some more common situations where the location of mines can easily be deduced.

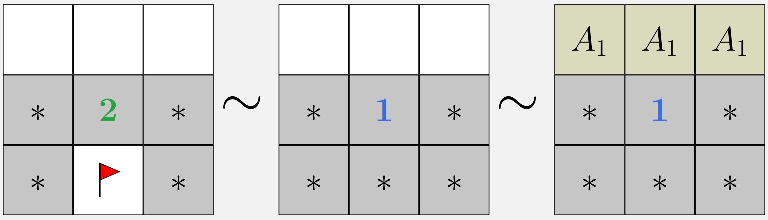
As the only action that can be performed with this pattern is flagging, no new information is actually obtained from this pattern, but placing flags following this pattern greatly helps with the visualization of the known information regarding the board.

Mine Reduction (Chordables)

When a numbered cell is adjacent to a number of known mines (flags or a complete set of MSM squares), the number of mines can be subtracted from the known number and logic can be applied as normal, treating the known mines as don't care (DC) squares.

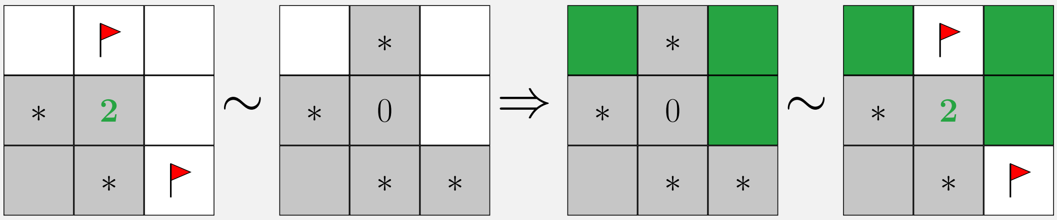
If a mine reduction reduces a number to 0 (i.e. the number of mines around a numbered square equals the number in the square), all neighboring cells to the number can be cleared. If the known mines are explicitly marked with flags, the numbered square can be clicked to perform a chord.

Example 2.3: Mine Reduction



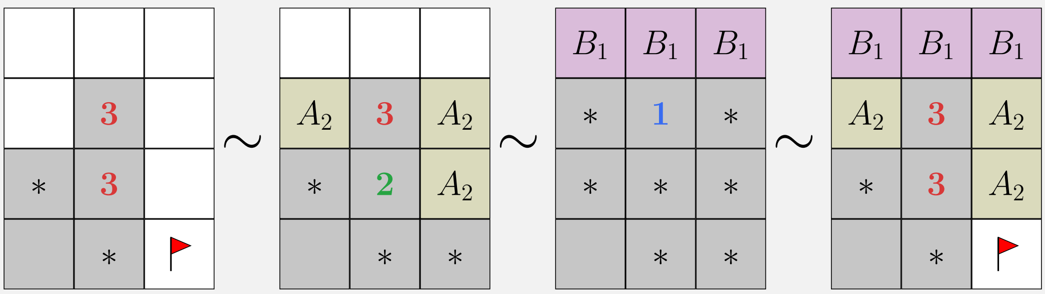
Since there is one known mine adjacent to the 2, the 2 can be thought of as a 1. From there, no more action can be taken, but it can be concluded that the remaining three unknown squares mutually contain one mine.

Example 2.4: Mine Reduction into Chordable



An example of a case where mine reduction reduces the number to zero. In this case, all neighbors of the reduced number can safely be cleared or chorded. More often than not, the middle two steps are not actually visualized since it's not difficult to count the number of mines around a square.

Example 2.5: Mine Reduction with Mutually Shared Mine Squares



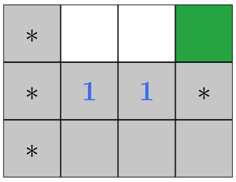
A more complicated example illustrating how mutually shared mine (MSM) squares can also be used in mine subtraction. Note that all the MSM squares in the same group must be adjacent to the number you want to reduce. This concept will be key in understanding the next two patterns.

2.2.2 Core Patterns

There are two core patterns, the 1-1 pattern and the 1-2 pattern. For someone discovering minesweeper strategy on their own, these will often be among the first patterns they discover. In their simplest form, these patterns are a bit restricting, but the logic used to discover them lead to some nice generalized patterns that can still be recognized at a glance. In a Section 3, it will be discussed how these two patterns are actually the same pattern.

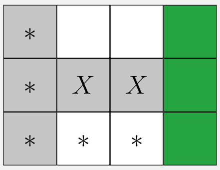
1-1 Pattern

When encountering a horizontally or vertically adjacent numbers with the same value, this is a common tool. The 1-1 pattern traditionally has the following form:

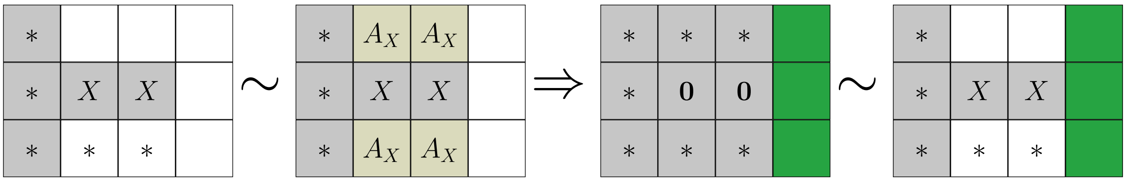


While most commonly occurring on the straight edge of an opening, it can be generalized to the following

Generalized 1-1 Pattern

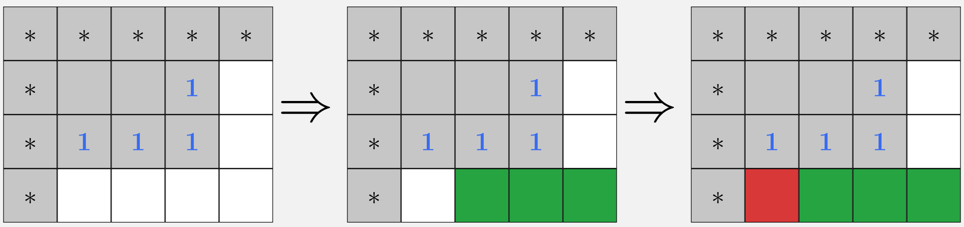


where X is any number between 1 and 3. The reasoning behind this pattern can be seen as a sequence of mine reductions as follows:



The left X makes all adjacent unknown squares into an MSM group. since the MSM group is also adjacent to the right X, the right X can be reduced to a zero, hence all the remaining unknown squares adjacent to the right X can safely be cleared.

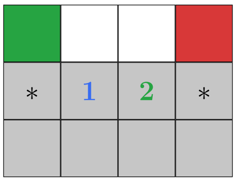
Example 2.6: 1-1 Pattern in 2×3 corner opening



A 1-1 pattern can be seen here going vertically. This infers 3 clearable squares, which in turn imply that a mine exists due the the existence of a 1-corner after clearing the 3 squares.

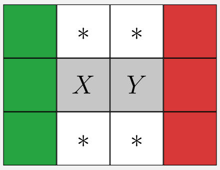
1-2 Pattern

Along with the 1-1 pattern, this pattern is another common tool used when encountering a straight line of numbers. The most common manifestation of the 1-2 pattern has the following form:



Like the 1-1 pattern, the 1-2 pattern can also be generalized to the following

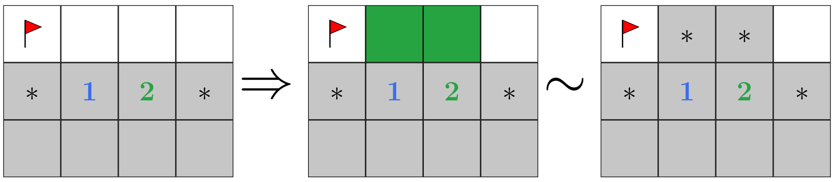
Generalized 1-2 Pattern



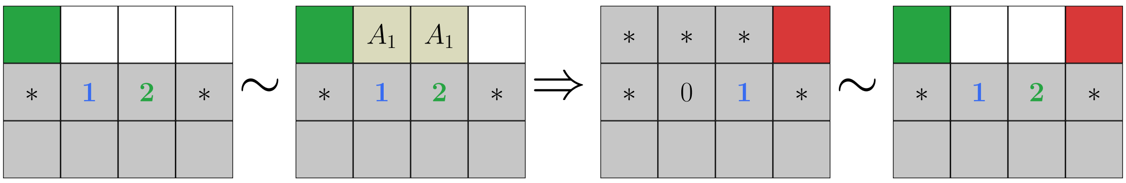
Given $Y - X$ is equal to the number of squares on the right side that are unknown.

We can see that in our common case, $X = 1$ and $Y = 2$, with there being $2 - 1 = 1$ unknown square in the upper right corner. It is of note that the 1-1 pattern is a special case of the generalized 1-2 pattern, where $X = Y$.

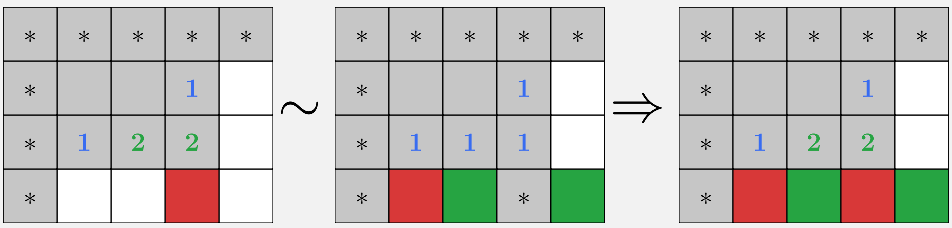
The explanation for this pattern is a little bit more involved, so I will instead explain the common case, and the general case can be proven with similar logic. Suppose the upper left corner contains a mine. Then with chording, we get the following



which leads to an invalid board since there are no ways to place two mines adjacent to the 2. Hence the initially assumed mine must be safe. We can then conclude the upper right corner is a mine through mine subtraction.



Example 2.7: 1-2 Pattern in 2×3 corner opening



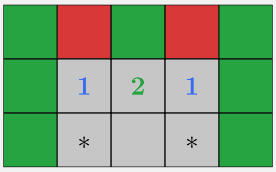
A mine can be inferred from the 1-2 pattern. After mine reduction, we can see athe 1-1 pattern from Example 2.6, where we can infer two more clear squares and one more mine.

2.2.3 Intermediate Patterns

1-2-1 Pattern

This pattern can be solved with the previously mentioned 1-2 pattern, but it is common enough to warrant memorization on its own.

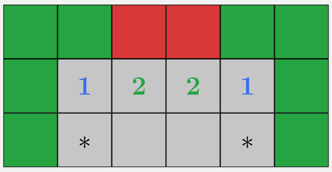
1-2-1 Pattern



1-2-2-1 Pattern

Like the 1-2-1 pattern, this is another manifestation of the previously mentioned 1-2 pattern that is also common enough to warrant straight memorization.

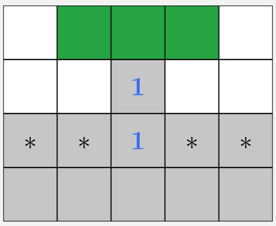
1-2-2-1 Pattern



1-T Pattern

This normally occurs following a 1-1 pattern or a 1-2 pattern if a 1 is appears on clear. If the 1 is cleared adjacent to another 1, all three squares opposite are clear. One may notice that this is simply a variation of the 1-1 pattern, but is worth memorization.

1-T Pattern



Mirror Pattern

This normally occurs following a 1-T pattern, but also frequently occurs between two openings in close proximity. Whenever a square is on a wall and the only adjacent unknown squares are on one side, followed by a numbered square 2 away, that number must "mirror" the number on the wall.

Mirror Pattern

	A_Z	A_Z	A_Z	
	A_Z	**Y**	A_Z	
	*	*	*	
*	*	**X**	*	*

Where $Z = Y - X$.

The most notable case is when $X = Y$, meaning $Z = 0$ so all of A are clearable

Mirror Pattern Special Case

		X		
	*	*	*	
*	*	**X**	*	*

Where $Z = Y - X$.

2.3 "Simple" Guessing

There may be situations where you are unable to apply patterns you know or use logic to find the next square to click. It may also be that time is ticking and you simply want to get another click in as quick as possible. In both of these cases, you are forced to make a guess. Guessing inherently has a chance of ending your game immediately, but in my opinion, it is what makes playing minesweeper fun.

Unfortunately the math for determining how to guess is rather cumbersome, and not too useful for beginner and intermediate players, so as a TL;DR, the guessing strategy for the generically available difficulties using only local information is as follows:

Guessing Priority via "Simple" Heuristic

1. Blind Guesses (guesses not adjacent to a numbered square) with less than 3 completely unknown neighbors
2. **Corners**; Blind Guesses with exactly 3 completely unknown neighbors
3. Squares near numbers with maximum effective mine count of 1 among ≥ 3 shared squares and ≤ 4 completely unknown neighbors; Blind Guesses with exactly 4 completely unknown neighbors
4. **Edges**; Blind Guesses with exactly 5 completely unknown neighbors
5. Low mine probability areas near numbered squares (just use intuition)
6. Anything else

A square is said to be completely unknown if it is not adjacent to any cleared cells. The explanation for this table is given in Section 4.4

3 Playing Without Guessing

On average, an expert board requires about 170 clicks. It would be infeasible to attempt to guess on too many clicks, so exercising logic to confidently clear the board when possible is required.

It should be noted that No Guessing (NG) variations of minesweeper exist, where every move can be made definitely with logic. If one wants to practice logic and learn advanced patterns, NG minesweeper is a good place to start.

3.1 Mathematical Notation

From now on, here be dragons. In order to describe the logic of minesweeper, I will go into an unnecessarily detailed construction of how a board state can be described. The purpose of this notation is two-fold. First, when discussing patterns, it would be ideal to identify them in an orientation independent manner, which this notation will aim to achieve. Second, defining board state in terms of real-valued functions will make discussing probability more concise. As with much of mathematics, formalizing a problem into a usable language is half the battle. For this document, I will adopt similar notation to that use by Philip Crow in "A Mathematical Introduction to the Game of Minesweeper"[4] with some modifications.

Minesweeper Board

A Minesweeper Board B with r rows and c columns is defined as a 4-tuple of functions

$$B = (n, M, C, N)$$

where $n \in \mathbb{Z}^+$ is the number of mines on the board, and $M : \mathbb{Z}^2 \to \{0, 1\}$, $C : \mathbb{Z}^2 \to \{0, 1\}$, and $N : \mathbb{Z}^2 \to \mathbb{Z} \cup \{*\}$ are "knowledge" functions defined as follows:

- "Mined": $M(a) = \begin{cases} 1 & \text{if } a \text{ is known to contain a mine} \\ 0 & \text{o/w} \end{cases}$
- "Clear": $C(a) = \begin{cases} 1 & \text{if } a \text{ is known to not contain a mine} \\ 0 & \text{o/w} \end{cases}$
- "Number": $N(a) \in \{0, \ldots, 8, *\}$ is the number of mines known to be adjacent to a, where a $*$ indicates we don't know/care

with the following properties:

- $\forall a \in \mathbb{Z}^2$, $M(a) + C(a) \leq 1$: a cell cannot both contain a mine and not contain a mine
- If $a \notin [0, c) \times [0, r)$, $C(a) = 1$: cells not on the board are known to not contain a mine
- $\sum_{a \in \mathbb{Z}^2} M(a) \leq n \leq \sum_{a \in [0,c) \times [0,r)} (1 - C(a))$: there must be a valid total number of mines
- $\forall a \in \mathbb{Z}^2$ if $N(a) \in \mathbb{Z}$, $N(a) \in [\sum_{b \in K(a)} M(b), 8 - \sum_{b \in K(a)} C(b)]$: $N(a)$ must be consistent

We'll describe the grid of a minesweeper board as a subset of the positive region of a two-dimensional plane of integers, where the lower left corner of the board is at the origin. We will assume the board dimension (r, c) to be a constant throughout our discussion. The board state is described as a set of three functions M, C, and N. M indicates if a cell is known to contain a mine or not (i.e. flagged, or just mentally noted). C indicates if a cell is cleared (i.e. opening or numbered square). N describes the number of mines adjacent to a cell. Note that the value of $N(a)$ is only known when $C(a) = 1$, so if $C(a) = 0$, we don't know or don't care about the value of $N(a)$. As M and C are indicator functions, we will often refer to their size as the size of the set they indicate. Together, the cells indicated by M and C are known as the knowledge set, since it includes cells we have knowledge of. M, C and other indicator functions $f : \mathbb{Z}^2 \to \{0, 1\}$ can and will be also notate the set they indicate (e.g., saying $M \subset \mathbb{Z}^2$ is valid).

[4] https://minesweepergame.com/math/a-mathematical-introduction-to-the-game-of-minesweeper-1997.pdf

In order to further our discussion of minesweeper logic, it can be useful to introduce some more functions that can be derived from the 3 defined functions. These are described below.

Function	Signature	Definition	Description
U	$U : \mathbb{Z}^2 \to \{0, 1\}$	$U(a) = 1 - M(a) - C(a)$	$U(a)$ indicates if the content of a cell is unknown
N_k	$N_k : \mathbb{Z}^2 \to \{0, 1\}$	$N_k(a) = \begin{cases} 1 & \text{if } N(a) = k \\ 0 & \text{o/w} \end{cases}$	N_k indicates that a cell has number k
K	$K : \mathbb{Z}^2 \to 2^{\mathbb{Z}^2}$	$K(a) = \{b \in \mathbb{Z}^2 : \|b - a\|_\infty = 1\}$	$K(a)$ is the set of cells neighboring a
K^+	$K^+ : \mathbb{Z}^2 \to 2^{\mathbb{Z}^2}$	$K(a) = \{b \in \mathbb{Z}^2 : \|b - a\|_\infty \leq 1\}$	$K^+(a)$ is the set of cells neighboring a including a itself.
K_F	$K_F : \mathbb{Z}^2 \to 2^{\mathbb{Z}^2}$	$K_F(a) = \{b \in K(a) : F(a) = 1\}$	$K_F(a)$ is the set of cells neighboring a indicated by function F
P_F	$P_F : \mathbb{Z}^2 \to \mathbb{R}$	$P_F(a) = P(F(a) = 1)$	$P_F(a)$ is the probability that a is indicated by F (Most often as P_M)

Clearing squares functionally increases the size of C, and sequences of logic functionally increase the size of M. While the knowledge functions may change as actions are performed, the underlying game remains the same. As such, it can be useful to describe relations between boards.

Board Continuation

Let $B_1 = (n_1, M_1, C_1, N_1)$ and $B_2 = (n_2, M_2, C_2, N_2)$ be two minesweeper boards. We say B_2 is a continuation of B_1, denoted $B_1 \Rightarrow B_2$, if all of the following conditions hold:

- $n_1 = n_2$: same number of underlying mines
- $\forall a \in \mathbb{Z}^2$, $M_1(a) \leq M_2(a)$ and $C_1(a) \leq C_2(a)$: B_2 has at least the mine state of B_1
- $\forall a \in [0, c) \times [0, r)$ such that $N_1(a) \in \mathbb{Z}$, $C_1(a)N_1(a) = C_1(a)N_2(a)$: The known numbers in B_1 are in B_2

Put in English, $B_1 \Rightarrow B_2$ means that the board state of B_2 can possibly be the result of playing B_1.

Board Completion

A minesweeper board is **complete** if $\forall a \in \mathbb{Z}^2$, $M(a) + C(a) = 1 - U(a) = 1$.

A board is complete if the contents of all the squares are known (full knowledge set). Under these definition, it should be clear that the playing minesweeper is equivalent to finding a sequence of boards $B_1, B_2, \ldots, B_n$ such that for all $i < n$, $B_i \Rightarrow B_{i+1}$, and B_n is complete.

An observant reader may have noticed that it's possible to have a board that is valid under our definition, but is completely unsolvable. This is because the definition can only define validity within each square's immediate neighborhood. However, if a board is complete, validity by our definition holds if and only if the board is legal (TODO prove?). As such, we'll say a board B is **solvable** if there exists a complete board B_n such that $B \Rightarrow B_n$.

Minesweeper boards can often get very complicated very fast as the number of mines increases, so it can be useful to describe an equivalence relation between boards with a different number of mines.

Board Similarity

Let B_1 and B_2 be two minesweeper boards. We say B_1 is similar to B_2 with respect to $A \subset \mathbb{Z}^2$, notated $B_1 \sim_A B_2$, if for all $a \in A$, the following holds true:

- $U_1(a) = U_2(a)$
- If $U_1(a) = 1$, $P_{M_1}(a) = P_{M_2}(a)$

We'll say that two boards are similar if they have shared a set of unknown squares, and those unknown square have the same probability of containing a mine. At the risk of shooting myself in the foot by using P_M in the definition, I believe that this is the most intuitive definition of board similarity. It should be clear from this definition, that two boards are similar if and only if their continuation of play is identical. For simplicity in some examples, the set A over which the sets are similar may not be explicitly stated. In which case $B_1 \sim B_2$ if and only if $B_1 \sim_{U_1} B_2$ and $B_1 \sim_{U_2} B_2$.

3.2 Basic Patterns Revisited

With our new notation, we can revisit our basic patterns in a more formal sense. Since these theorems outline what a player ought to do in a board state, it can be useful to understand mathematically what an action that ought[5] to be done would look like.

Action that ought to be done on a board

Let $B = (n, M, C, N)$ and $A \subset U$. There **ought** to be k mines in A, notated $M(A) \mapsto k$ if $\forall B' = (n', M', C', N')$ such that $B \Rightarrow B'$, $\sum_{a \in A} M'(a) = k$. Equivalently defined, being ought to be k clear cells in A is notated $C(A) \mapsto k$.

Note that if $M(A) \mapsto |A|$, all the cells contain mines and ought to be flagged, and if $C(A) \mapsto |A|$, all the cells are safe and ought to be cleared.

Proposition 3.1

$M(A) \mapsto k$ if and only if $C(A) \mapsto |A| - k$

TODO: prove (should be intuitive though)

From this, we can understand that we ought to increase the size of our knowledge set M or C by including values in U (recall that $U(a) = 1 - M(a) - C(a)$ are unknown cells, or cells from which we have no knowledge of yet), if and only if all valid complete board states that can result from a board has that augmented knowledge set.

Simply put, if $M(\{a\}) \mapsto 1$, we concluded a contains a mine and should be flagged. On the other hand, if we ought to have $M(\{a\}) \mapsto 0$, we concluded a does not contain a mine and should be cleared.

Theorem 3.1: Induced MSM

Suppose we have board B. Let $\hat{B} \sim B$ such that $\forall a$, $\hat{M}(a) = 0$. Then $\forall a$ such that $\hat{N}(a) \in \mathbb{Z}$, $M(K_U(a)) \mapsto \hat{N}(a)$

Theorem 3.2: MSM Subset

Let $A \subset B \subset U$. If $M(A) \mapsto k_A$ and $M(B) \mapsto k_B$, then $M(B \setminus A) \mapsto k_B - k_A$

[5] As per Hume's Guillotine, although you ought to do an action logically, you are not forced to comply, but noncompliance will result in your death. For real though, a bit of poor choice of wording, but I couldn't think of better notation

Theorem 3.3: True 1-2 Pattern

Let $A, B_1, \ldots, B_m \subset U$ such that $\forall i\ A \cap B_i \neq \emptyset$ and if $i \neq j$, $B_i \cap B_j = \emptyset$. If $M(A) \mapsto k_A$ and $M(B_i) \mapsto k_i$, then if $|A \setminus \bigcup_{i=1}^m B_i| = k_A - \sum_{i=1}^m k_i$, then we have the following

- $\forall a \in A \setminus \bigcup_{i=1}^m B_i$, $M(\{a\}) \mapsto 1$
- $\forall b \in (\bigcup_{i=1}^m B_i) \setminus A$, $M(\{b\}) \mapsto 0$

Let us now formally state the basic minecounting theorems.

Corollary 3.1.1: All Mines

If $N(a) = |K_U(a)|$, then $M(K_U(a)) \mapsto 1$.

Theorem 3.1.1: Chordable

If $N(a) = |K_M(a)|$, then $M(K_U(a)) \mapsto 0$.

TODO: prove these two. should be easy to do with contradiction.

Corollary 3.3.1: Generalized 1-2 Pattern

If $C(a) = C(b) = 1$, $|b - a| = 1$, and $N(b) - N(a) = \sum_{c \in K^+(b) \setminus K^+(a)} U(c)$, then we have the following:

- $\forall c \in K^+(b) \setminus K^+(a)$, $M(c) \mapsto 1$
- $\forall c \in K^+(a) \setminus K^+(b)$, $M(c) \mapsto 0$

TODO: prove

3.3 More Patterns

TODO

3.4 Minecounting

TODO

Complement MSM

The **Complement MSM** of a board is the set

$$U_C = U \setminus \bigcup_{\alpha} K(\alpha)$$

where $\alpha \in \mathbb{Z}^2$ such that $N(\alpha) \in \mathbb{Z}$ and $C(\alpha) = 1$

4 Guessing

4.1 Probability

Probability a Square Contains a Mine

The first part of informed guessing comes from knowing what the probability of any unknown square containing a mine is. Our goal is to define the probability that a square contains a mine given a particular board state.

Suppose we have probability space $(\Omega, 2^\Omega, P)$. Recall that if the sample space Ω is finite and P describes a discrete uniform distribution, the Probability of an event $A \in 2^\Omega$ is simply the number of samples where A occurs divided by the size of the sample space.

$$P(A) = \frac{|A|}{|\Omega|}$$

Let us determine our probability space for our mine probability problem. Suppose our current board state is B. The probability space of interest is over the set of complete board states that continue B, $\Omega_B = \{B' | B \Rightarrow B', B' \text{ complete}\}$. Note that the function $M(a)$ is a valid random variable over this space, where the selected board determines which mine function to use.

We can then see that our desired probability is $P_M(a)$. Under the definition of random variables and the assumption that all complete boards have equal probability, we get the following formulation.

Single Square Mine Probability

The probability of a square $a \in \mathbb{Z}^2$ containing a mine for board B is given by

$$\begin{aligned} P_M(a) &= P(M(a) = 1) \\ &= P(\{B' \in \Omega_B | M_{B'}(a) = 1\}) \\ P_M(a) &= \frac{|\{B' \in \Omega_B | M_{B'}(a) = 1\}|}{|\Omega_B|} \end{aligned}$$

This has the simple interpretation of the number of valid board complete boards with a mine at a, divided by the total number of valid complete boards. Since a complete board has $M(a) = 1 - C(a)$ for all a, it must also be that $P_C(a) = 1 - P_M(a)$.

It is often the case that it is very difficult to compute these counts, and hence very difficult to compute exact probabilities. However, computers can be used to compute this probability exactly. Some methods to compute this probability are explored in Section 7.

An interesting property of the probability $P_M(a)$ is that the sum of all $P_M(a)$ is equal to the number of mines on the board n. If $B' \in \Omega_B$, then $B \Rightarrow B'$, meaning all B' have the same number of mines n. Also note that for any complete $B' \in \Omega_B$, $\sum_{a \in \mathbb{Z}^2} M_{B'}(a) = n$. Taking the expected value $E[\sum_{a \in \mathbb{Z}^2} M(a)]$

$$\begin{aligned} \sum_{a \in \mathbb{Z}^2} P_M(a) &= \sum_{a \in \mathbb{Z}^2} (1 \cdot P(M(a) = 1) + 0 \cdot P(M(a) = 0)) \\ &= \sum_{a \in \mathbb{Z}^2} E[M(a)] \\ &= E\left[\sum_{a \in \mathbb{Z}^2} M(a)\right] \\ &= n \end{aligned}$$

Higher Dimensional Mine Probability

Suppose we have a relatively good idea that one square is likely a mine (or not a mine). How does this change our probabilities for other squares? For this, recall the definition of conditional probability and

Baye's theorem.

$$P(H|E) = \frac{P(H,E)}{P(E)} = \frac{P(E|H)P(H)}{P(E)}$$

Given two squares $a_1, a_2 \in \mathbb{Z}^2$, we can call $P(M(a_1), M(1_2)) = P_{M,M}(a_1, a_2)$ the 2-dimensional joint mine probability distribution. We can equivalently define any n-dimensional joint mine probability function $P_{M,\ldots,M}(a_1, \ldots, a_n)$. We can go further and even define any general joint probability $P_{F_1,\ldots,F_n}(a_1, \ldots, a_n)$ where F_k can be any indicator function over the cells $\mathbb{Z}^2$ like C, M, or N_k. These n-dimensional joint probabilities can be computed analogously to the single square case as follows

n-Dimensional Joint Probability

Let $a_1, \ldots, a_2 \in \mathbb{Z}^2$ and $F_1, \ldots, F_n$ be indicator functions over $\mathbb{Z}^2$. Then the n-dimensional joint probability can be computed by

$$P_{F_1,\ldots,F_n}(a_1, \ldots, a_n) = \frac{|\{B' \in \Omega_B | (F_1)_{B'}(a_1) = 1, \ldots, (F_n)_{B'}(a_n) = 1\}|}{|\Omega_B|}$$

where the joint probability over $a_1, \ldots, a_n$ is equal to the number of boards completing B that fulfill the indicator function.

Returning to our question of the probability of a square being a mine given knowledge of another square, this becomes a simple case of conditional probability.

$$P_{M|F}(a_1|a_2) = P(M(a_1) = 1 | F(a_2) = 1) = \frac{P_{M,F}(a_1, a_2)}{P_F(a_2)} = \frac{|\{B' \in \Omega_B | M_{B'}(a_1) = 1, F_{B'}(a_2) = 1\}|}{|\{B' \in \Omega_B | F_{B'}(a_2) = 1\}|}$$

where F can be M, C, or any other indicator function. So as we can see, if we have knowledge of n squares, we can compute the probability of m squares if we have the $(m+n)$-dimensional and n-dimensional joint probability distributions. In our case of a single square predicting another, we need a 2-dimensional joint distribution in addition to the normal single dimensional distribution.

Under this notion, we can define the notion of progress.

Progress

Knowledge of a square a to be indicated by F yields **progress** for board B if $\exists b \in U$ such that

$$P_{M|F}(b|a) = 0$$

In other words, knowledge of a yields progress if that knowledge allows for the clearing of another square b, since clearing a square will yield new knowledge in the form of $N(b)$.

Probability of Winning

Now let's actually look into the probability of a guess leading to a win. One can describe the ideal guessing strategy as guessing the unknown square that maximizes the chance of winning. Put informally this perfect strategy can be given by

$$\text{Best Guess} = \arg\max_{a \in U} P(\text{Win}|\text{Guessed } a)$$

Let's attempt to break down this statement into something computable.

4.2 Usefulness - Minimizing Risk

TODO

4.3 The 50-50

TODO

4.4 Heuristics - "Simple" Guessing Revisited

Despite the discussion of probability, humans are not very good at computing with large amounts of information very fast. As such, heuristics are typically used to reduce the scope of the computation while still yielding approximate results.

Blind Guessing

Under the assumption that all cells contain a mine independent of each other (which is usually NOT true, but is an okay approximation), it can be said that each cell has an equal chance of containing a mine. We can make another naive assumption that mine statistics are stationary, or in other words, don't change from the first board B_0. This chance is given by taking the number of mines possible on the board, and dividing by the total number of squares.

Blind Probability

A naive approximation of the probability of a mine on any given square is given by

$$P_M(a) \approx \frac{n}{rc}$$

This probability is also known as the **mine density** of the board. The lower the mine density of a board, the lower the chance blind guessing will lead to death, generally leading to an easier difficulty. We can see that this probability is only dependent on the difficulty or board settings.

Difficulty	n	rc	Blind Probability
Beginner[6]	10	$8 \times 8 = 64$	$\frac{10}{64} \approx \mathbf{0.156}$
Intermediate	40	$16 \times 16 = 256$	$\frac{40}{256} \approx \mathbf{0.156}$
Expert	99	$16 \times 30 = 480$	$\frac{99}{480} \approx \mathbf{0.206}$

Table 2: Blind Probabilities by Difficulty

The blind probability can be used as a benchmark to measure whether or not a certain method of guessing should be employed.

Simple Guessing with Numbered Squares

Blind guessing can be improved if some information is known. In most cases, the information we'll deal with for a given square are the immediately adjacent numbers. In the case that multiple numbers are adjacent to a cell, it will be safer to assume that it is the riskiest probability. It should be noted again that this probability is only a heuristic that is easy to compute on the fly, and is does not reflect the actual probability of a mine being in a cell.

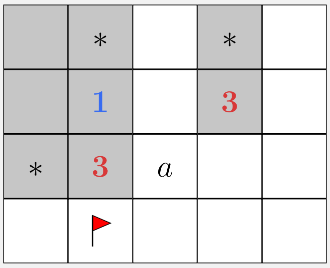
Simple Single Number Probability (SSNP)

A naive approximation of the probability of a mine on a square adjacent to number squares is given by

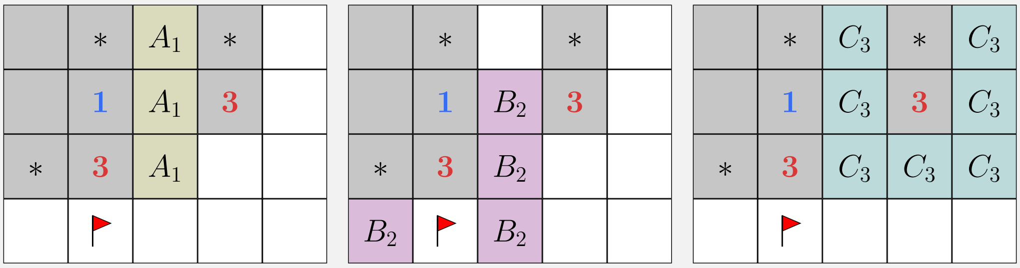
$$P_M(a) \approx \max_{b \in K(a)} \frac{N(b) - \sum_{c \in K(b)} M(c)}{\sum_{c \in K(b)} U(c)}$$

The SSNP of a mine is simply the max of the local mine probabilities of the neighboring cells. Although this approximation is still inacurate, it is still relatively easy to guess the value of in less than a second.

Example 4.1: Computing SSNP



Look at the MSMs of the adjacent numbers



$$\frac{1-0}{3} = \frac{1}{3} \qquad \frac{3-1}{4} = \frac{1}{2} \qquad \frac{3-0}{7} = \frac{3}{7}$$

So $P_M(a) \approx \frac{1}{2}$

		# of unknown (MSM) squares adjacent to cell						
		2	3	4	5	6	7	8
mine count	1	0.5	0.333	**0.25**	**0.2**	**0.167**	**0.143**	**0.125**
	2		0.667	0.5	0.4	0.333	0.28	**0.25**
	3			0.75	0.6	0.5	0.429	0.375

Table 3: Simple Single Number Probability

Table 3 shows the simple single number probabilities for up to an effective mine count of 3. It should be notated that as the effective mine count increases, the probability of a mine via this probability significantly increases. Since the mine density of a board is often between 0.15 and 0.25, basing guesses on the simple single number probability is often only a good idea if the effective mine count is 1, or very rarely 2.

Evaluating Guesses - Usefulness

At first glance, one may assume that the best guess is the guess with the lowest probability of a mine. However, this is only half correct. In reality, a guess is pointless if no new useful information is gained, forcing you to guess again. As such, in addition to the probability of a square being a mine, you also need to consider the probability of a square being useful.

Guessing Score

The **Guessing Score (GS)** of an unknown square is given by

$$GS(a) = (1 - P_M(a)) \cdot (S_{useful}(a))$$

If a guess is required, the square with the highest GS should be clicked

A guess is useful if it minimizes the overall risk you'll have to take when solving the rest of the board. The usefulness of a guess is in fact much harder to evaluate than the probability of a guess being a mine. In my opinion, evaluating the usefulness of a guess is the hardest skill to develop as a minesweeper player. To keep things simple, I will only give one usefulness heuristics.

Usefulness: No New Mines

Define indicator $X(a) = 1$ if and only if $\sum_{b \in K(a)} C(b) = 0$, indicating that the cell has no cleared neighbors. We'll say $X(a)$ indicates that a is **completely unknown**

$$S_{useful}(a) \approx P(\sum_{b \in K_X(a)} M(b) = 0) \approx \left(1 - \frac{n}{rc}\right)^{\sum_{b \in K(a)} X(b)}$$

This heuristic essentially computes the likeliness that there are no mines adjacent to the cell you want to guess, except for the mines you already know about.

The downside of this heuristic however, is that all of the numbered square's unknown (MSM) squares must also be adjacent to the square being guessed in order to be valid. Despite this, there will normally exist guessable squares where this is valid.

Under the naive premise mine density is fixed, this heuristic only depends on the number of neighboring unknown cells. Table 4 gives the guessing score if the cell being guessed is a blind guess (not adjacent to a numbered square).

Difficulty	$\sum_{b \in K(a)} X(b)$	$P_M(a)$	$S_{useful}(a)$	$GS(a)$
beginner/intermediate	1	0.156	0.844	**0.712**
	2		0.712	**0.601**
	3		0.601	**0.507**
	4		0.507	**0.428**
	5		0.428	**0.361**
	6		0.361	0.304
expert	1	0.206	0.794	**0.63**
	2		0.63	**0.5**
	3		0.5	**0.397**
	4		0.397	**0.315**
	5		0.315	**0.25**
	6		0.25	0.199

Table 4: Guessing Score from a Blind Guess Near a Numbered Square

In the special case that the guess is especially blind (at least 2 squares away from any numbered squares causing all neighbors to be unknown squares), we can see that there are three main scenarios based on where the square being guessed is on the board: the middle, the edge, and the corner. If a cell is in the middle, it has 8 unknown neighbors. If a cell is on an edge, it has 5 unknown neighbors. Finally if a cell is on an edge, it only has 3 unknown neighbors. From this information, we can compute the GS from blindly guessing for each square based on difficulty (mine density). This is summarized in Table 5.

The final case is if the square being guessed is not blind, but rather adjacent to a numbered square.

Difficulty	Location	$P_M(a)$	$S_{useful}(a)$	$GS(a)$
	corner		0.601	**0.507**
beginner/intermediate	edge	0.156	0.428	**0.361**
	middle		0.257	0.217
	corner		0.5	**0.397**
expert	edge	0.206	0.315	**0.25**
	middle		0.158	0.125

Table 5: Guessing Score from a Blind Guess

Table 6 summarizes the Guessing Score in these cases. Note that Mine Count (MC) is obtained through mine subtraction of the numbered square, MSM count is the number of squares in common between the numbered square and the cell being guessed, and unknown count is the number of unknown squares exclusively adjacent to the cell being guessed.

			# of completely unknown squares adjacent to cell							
			Beginner/Intermediate				Expert			
MC	MSM	$P(M(a)=1)$	2	3	4	5	2	3	4	5
	2	0.5	0.356	0.3	0.253	0.214	**0.315**	**0.25**	0.198	0.158
1	3	0.333	**0.475**	**0.4**	0.338	0.285	**0.42**	**0.333**	**0.265**	0.21
	4	0.25	**0.534**	**0.451**	**0.38**	0.321	**0.473**	**0.375**	**0.298**	0.236
	5	0.2	**0.57**	**0.481**	**0.405**	0.342	**0.5**	**0.4**	**0.318**	**0.252**
	4	0.5	0.356	0.3	0.253	0.214	**0.315**	**0.25**	0.198	0.158
2	5	0.4	**0.427**	0.36	0.304	0.257	**0.378**	**0.3**	0.238	0.189

Table 6: Guessing Score from a Guess Near a Numbered Square

The contour of the guessing score is rather complicated with all of the independent variables involved. However, since generally an edge square is usually available to be guessed, we mostly just need to hone in on the cases where guessing near a number is better than guessing in the corner, and guessing on an edge.

It can be seen that the only guess guessing near a number is better than guessing in the corner is when both the effective mine count is low across many squares with few unknown adjacent squares. The cases where this is true are relatively uncommon, and would probably be best detected via intuition.

More interesting are the squares that are better to guess at than guessing at the edges. Notably, in all difficulties, it is better to guess a 33% mine probability square along a wall of numbered squares (1MC, 3MSM, 3 unknown) than it is to guess at an edge.

To summarize the tables, we can generate a simple guessing priority mentioned earlier in Section 2. Note that the intersection of the guessing score functions are not linear, so very rough approximations are made to succinctly state some priority list.

Guessing Priority via "Simple" Heuristic

1. Blind Guesses (guesses not adjacent to a numbered square) with less than 3 completely unknown neighbors
2. **Corners**; Blind Guesses with exactly 3 completely unknown neighbors
3. Squares near numbers with maximum effective mine count of 1 among ≥ 3 shared squares and ≤ 4 completely unknown neighbors; Blind Guesses with exactly 4 completely unknown neighbors
4. **Edges**; Blind Guesses with exactly 5 completely unknown neighbors
5. Low mine probability areas near numbered squares (just use intuition)
6. Anything else

It is of course important to note that this guessing priority entirely depends on the heuristic we defined. The goal of the "simple" heuristic is to only use local information in the vicinity of the guess, and to make broad probability assumptions to reduce probability calculations to simply counting squares. A different heuristic may lead to a different guessing priority.

5 Efficiency

TODO
tbh, it'd be difficult for me at the moment to do much better than Danoouch for this topic. Until I do more experimentation and math, please just refer to their guide[7]

[7] https://docs.google.com/document/d/1BBa4YgPucipQFE8jCYRE83y2iDzCE-71FZP7J_GXU6M/edit

6 Strategy

TODO: tbh, I'm still figuring this out too.

6.1 Winrate

- Flag everything (for minecounting)
- Only guess when necessary

TODO

6.2 Speed

- don't flag every known mine
- memorize more patterns
- guess slightly more aggressively
- play with efficiency in mind

TODO

6.3 Difficulty

- Restart until you have a good start
- Play near the edges

TODO

6.4 Efficiency

- Restart until you get a good amount of openings in the corners
- Chord if and only if doing so clears ≥ 2 squares
- Guess with the anticipation of openings

TODO

7 Algorithms

There are limits to what humans are able to compute, but for computers, those limits are much higher. Although this document aims to serve as a guide for human play, computer simulations able to compute probabilities described in Section 4 may aid us by adding guessing strategies to our repertoire. This section will cover the data structures and algorithms that aid in probability calculation.

7.1 Logical Board Reduction

MSM Graph

The **MSM Graph** (V, E) for a board B is defined by

$$V \subset \{A \subset \mathbb{Z}^2 \mid \exists k \text{ s.t. } M(A) \mapsto k\}$$
$$E = \{(A_1, A_2) \in V \times V \mid A_1 \neq A_2, A_1 \cap A_2 \neq \emptyset\}$$

where V is a subset of inferable MSMs, and a pair of MSMs have an edge in E if they intersect.

7.2 Single Mine Probability

Equiprobability Set

Equiprobability Sets $G_1, \ldots, G_n$ are disjoint subsets of $\mathbb{Z}^2$ such that $\forall k$, $\forall a, b \in G_k$ we have $P_M(a) = P_M(b)$

Graph

Let $G_1, \ldots, G_n$ be a set of equiprobability sets. The **Equiprobability Graph** $(V_1 \cup V_2, E)$ for a board B is a bipartite graph defined by

$$V_1 = \{G_1, \ldots, G_n\}$$
$$V_2 = \{a \in \mathbb{Z}^2 | C(a) = 1, N(a) \in \mathbb{Z}\}$$
$$E = \{(G, a) \in V_1 \times V_2 \mid G \subseteq K(a)\}$$

where the vertices are partitioned into V_1, a set of equiprobability sets, and V_2, the set of numbered cleared squares in B, such that a set and a number have an edge between them if the set lies in the neighborhood of the number.

TODO

1. more/better pictures
2. rewrite section 2 to be more beginner friendly
3. computer simulations for guessing
4. peer review? (idk anyone to review this)