An Investigation in Emacs 5x5 (FlipGrid)

Jackson Weidmann

2020, April 2

Contents

1 Introduction

In this paper I will be investigating the solutions to the 5x5 game from emacs on an OS X terminal. To avoid confusion I will be referring to the game as "flip grid". To access this game yourself, open the mac terminal, type emacs and hit enter. Then press ESC, then x and type '5x5' then hit enter. There will be an online version available soon. The program on emacs has a few techniques to solve the puzzle but it essentially relies on brute force. There is little information regarding the solution or the mathematics behind it. I found a paper that will be linked here that goes into some detail but skips over a lot. My goal is to address some of the questions regarding the solutions for different sizes. If we have time I may discuss this game played on different topologies.

2 Terminology

Some language surrounding this game can be ambiguos so I am going to explain my vocabulary here.

1. Game or Board refers to the grid of squares that we play on.

2. a square or a cell will refer to one of the squares in the grid.
3. a flip is when a cell flips between 0 and 1.
4. Playing on a square means that a flip is centered at that point. We could also call this a move or a move at a cell location.
5. Series or sequences of moves are just several moves performed at once. order does not matter so we can consider them to be performed simultaniously.
6. When a series of moves is performed on a row it is called a cycle.

3 Playing Flip Grid

I will now give an in depth explanation of game play and basic ideas of strategy. You start of with a 5x5 grid of squares, that are full of dots (or zeros). Your goal is to turn all the dots into hashtags (or ones). You can perform a move by choosing a square and "flipping" it, this causes that square and those adjacent to it to swap between 0s and 1s (dots become hashtags and hashtags become dots). It may be clear that the order in which you swap pieces does not matter.

The simple approach to solving this is to go row by row. Play a random configuration in the top row then in the second row play on the squares that have 0 above them. This will solve the top row and the second row state will be completely determined by which squares we played initially. Continuing this process we solve every row, but not necessarily the last one. With the correct board configuration the last row will end up solved, otherwise it will have gaps that can't be filled by playing on the rows below (as there are no lower rows).

4 Flip Grid Solution

4.1 Defining the Game

Before we can actually solve this game we must define it formally. First off the game takes place on a square board (there is no issue with using rectangles and this may be explored later) of size n. We will refer to a board as B_n. Let us consider a single cell, it's value is in 0,1. We can think of flipping this cell as adding 1 then taking mod 2.

Definition 4.1.1 *A cell in B_n at row n and column m is defined as $c_{i,j} = \mathbb{Z}_2$*

However $B_n \neq \mathbb{Z}_2^n$ because there are some combinations that are not valid game states. Let's call the initial board state B_n^0, this is a grid of all zeros. A move simply adds 1s to specific locations. We could think of a move as a grid of all 0s with 1s on the locations it would flip, that we add to B_n^0.

Definition 4.1.2 *Some move $P_{i,j}$ playing on $c_{i,j}$ is equivalent to $B_n^0 + P_{i,j}$*

Now we can figure out what a game board actually is.

Definition 4.1.3 *B_n is the group that is generated by the set of all moves $M = \{P_{i,j} : 1 \leq i, j \leq n\}$*

4.2 Row States

The solution mentioned above relies on the idea of working row by row. Row by row solving works here because the actual order of moves does not matter. We also will see that the state of the final row completely depends on where we play in the first row. Let us denote rows as $X_i = \{c_{i,j} \in B_n : 1 \leq j \leq n\}$. A row is considered solved if it consists of all 1s. The board is solved if all rows are solved.

Lemma 4.2.1 *If cycles have been performed on rows $X_i, i \leq k$ then we can solve row X_k by playing on all $c_{k+1,j} \in X_{k+1}$ where $c_{k,j} = 0$.*

Suppose we want to ensure some row $X_i = \{c_{i,j} : 1 \leq j \leq n\}$ ends up solved. The previous cycles leave X_k in some random state. Say $c_{k,j} = 0$, then let us play on some cell $c_{k+1,j} \in X_{k+1}$. This will affect all adjacent cells, but the only cell adjacent to $c_{k+1,j}$ in X_k is $c_{k,j}$. The cell $c_{k,j}$ becomes 1 and we can repeat this for all the 0s in X_k. □

Another way to say this is that you can completely solve a row without affecting the rows above it. If we look closely at this proof we might also notice that the state of a row can be determined from the two previous rows. This will help lead us to the Fundamental Theorem of Flip Grid that any solution can be determined from how we play on the top row. Before we get to this we should talk about cycles in more exact terms.

Definition 4.2.1 *A cycle is a function $Cy_n(k, X_i)$ that returns the state of X_i after row X_k is solved.*

We will define this function more precisely in a later section. A few notes on this definition; if $k + 2 < i$ then row X_i is far enough below that no moves affect it, hence $Cy_n(k, X_i) = X_i$. When $i < k$ we get the same result. When X_i is one or two rows below X_k then we get X_i after it's first and second cycle respectively. All rows X_s (except the bottom) will go through three cycles. The flips of the first two cycles are the results from solving rows X_{s-2} then X_{s-1}, and the last cycle solves X_s. The case of $k = 0$ will be discussed in section 4.3.

Now we are ready to approach the fundamental theorem. As we have seen we can solve a row without affecting the ones above it. We also are able to determine what the state of a row is based on the cycles (which are based off previous row states).

4.3 The Fundamental Theorem of Flip Grid

This is an important theorem because it is the basis of the algorithm that is used in this paper. I am not the first to come up with this concept (link source)

but I believe I am the first to formally prove it. They also do not provide much detail in their methods of determining how to play row one.

Let us now revisit the cycle function for B_n when $k = 0$. This is what determines the state of X_1 and X_2 after the initial play on X_1. But how are the moves determined? We will use an n-dimensional vector X_0, with 0s where we want to play on X_1 and 1s everywhere else. This will allow the cycle function to use X_0 to determine X_1 and X_2 the same way it determines rows when $k > 0$ (solving row X_k by playing on squares below 0s).

Theorem 4.3.1 *Any solution can be represented as how we play the top row*

We first note that it does not matter what order we perform moves so any solution can be performed row by row. It suffices to show that if any two solutions perform the same moves on the top row then they are the same solution. Suppose we play some set of moves on the top row. The first and second row states are determined by these moves. This determines the next set of moves because there is only one way to solve the first row while playing in the second row (4.2.1), all further play is on lower rows. This continues as each row is determined by the states of those above it. The play on the final row solves the board. □

4.4 Finding Solutions

The simple approach to finding solutions is to just test all possible first row states. This quickly becomes tedious for larger boards, although the symmetry of the board allows you to cut the search in half. Instead we will be using systems of equations to determine the final row state from the initial row state and work backwards.

Let us now define the cycle function more precisely. When we write ${}_0X_i$ we are referring to the initial row state of X_i which is just 0s. ${}_1X_i = Cy_n(i-2,{}_0\,X_i)$ and ${}_2X_i = Cy_n(i-1,{}_1\,X_i)$ and ${}_3X_i = Cy_n(i,{}_2\,X_i)$ which is solved. Consider row ${}_2X_{i-2}$, the effect of solving this row on ${}_0X_i$ is that we add 1 to all places where ${}_2X_{i-2}$ has 0s, from lemma 2.4.1. Now we present the formal defnition of the cycle function.

Theorem 4.4.1 *The first cycle of a row is as follows (where 1 is a 'row' of all ones)*

$$ {}_1X_i = Cy_n(i-2,{}_0\,X_i) = 1 -_2 X_{i-2} $$

Let's find the value of an element of the row. Consider ${}_0x_i, j \in_0 X_i$, which is made of zeros. If ${}_2x_{i-2,j}$ is zero then ${}_1x_{i,j} =_0 x_{i,j} + 1$ but otherwise we only add 0. So ${}_1x_{i,j} = 1 -_2 x_{i-2,j}$, which works for all rows hence

$$ {}_1X_i = 1 -_2 X_{i-2} = Cy_n(i-2,{}_0\,X_i) $$

□

Now after the second cycle is slightly more complex. We first will consider 2 matrices which will come into play. S_n is a vector with 2s on the first and last entry and 3s everywhere. T_n is a tridiagonal matrix with 1s on the thick

Theorem 4.4.2