LIGHTS OUT: DETERMINING SOLVABILITY ON RECTANGULAR BOARDS

TAMAR ELISE WILSON

A THESIS PRESENTED TO THE FACULTY OF MOUNT HOLYOKE COLLEGE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF ARTS WITH HONORS.

DEPARTMENT OF MATHEMATICS

SOUTH HADLEY, MASSACHUSETTS

MAY 8, 2009

Acknowledgements

I would like to thank my advisor, Jillian McLeod, for her support, guidance, and encouragement throughout the last two years. I would also like to acknowledge my readers, Bob Weaver and Harriet Pollatsek for their time and advice and finally, Laura, Ingrid, Celine, and Nicole, for helping with long nights of matrix multiplication and many games of pool.

Abstract

Lights Out is a game produced by Tiger Electronics consisting of a 5 by 5 grid of buttons, each of which can be either lit up or turned off. Each game begins with some ofthe buttons turned on, or lit. Pressing a button changes the status of that button- i.e., off to on or on to off- as well as the status of each its neighbors. The goal of the game is to systematically push some of the buttons until every light is turned off, winning the game. In the 5 by 5 grid, however, not every initial light pattern is solvable. In order to determine the solvability of some initial set of lights, we can convert the game into a system of linear equations given by a matrix R, which describes the effect of pushing any of the buttons, a strategy vector, $\vec{s}$, which will give us the solution if one exists, and an initial state vector, $\vec{x}$. Considering the game as a linear system allows us to easily generalize any solution methods to other sizes of game boards. Through the application of various solution methods and Fibonacci Polynomials, we classify rectangular game boards by the fraction of initial states that are solvable.

Contents

1. Introduction

1.1. **Introduction to the Game.**

Lights Out is a game produced by Tiger Electronics consisting of a 5×5 board of buttons, each of which can be lit or unlit. Pressing any button changes that button's state - lit or unlit - and also the states of the button's neighbors, or the buttons immediately above, below, and to the left and right. Buttons in the corners or on the side of the board change only the neighbors they touch; the effects do not wrap around the game board to the other side. Game play begins with some initial set of the buttons lit; the goal is to selectively press buttons until all of the lights are off. A few plays of the game lead to a general observation and strategy: the only way to turn off a light in a row without potentially turning on its neighbors is to push the button directly below it. Pressing that button will, of course, also change the buttons on its left and right as well as the one below, but if we consider just one row at a time, pressing the buttons below the lit buttons will allow us to get an all-off configuration in that row.

Example 1. *For example, consider the following initial set up, with lit buttons marked with an O.*

O	O			O
		O		
			O	
	O	O	O	
	O	O		

Pressing the first, second, and fifth buttons in the second row leads to

			O	O
O	O		O	O
	O	O	O	
	O	O		

effectively clearing the first row. Continuing to clear the subsequent rows in the same manner yields

O	O	O	O	O
	O	O		O
	O	O		

$\rightarrow$

			O	O
O			O	O

$\rightarrow$

O		O	O	O

Thus, any board can be condensed down into some configuration of lit buttons in the bottom row. Since each button has only two states, there are 2^5 possible configurations of these buttons. In order to solve the board from here, one can memorize sequences of button pushes that may clear the bottom row. Not every configuration of bottom row buttons, however, can be cleared, so not every initial configuration is winnable. Determining which configurations are winnable and what sequences of button pushes are needed to win can be done, albeit somewhat impractically, through trial and error. To determine solutions more easily, we can represent the game as a linear system and use linear algebraic methods to test if initial configurations are winnable.

1.2. **Converting to a Linear System.**

Before we translate the game to a linear system, we need a couple of observations. First, since a button can only be on or off, pushing a button twice is the same as not pushing it at all.

Thus, the entries of our matrix and vectors will all be taken from the binary field and all operations will be performed modulo 2. Also, if some sequence of button pushes changes the all-off board to some other configuration, repeating the same sequence will clear the board. Thus, some initial configuration is solvable if and only if we can find a sequence of pushes that lead from an all-off game board to the configuration.

To express the game as a linear system, first, number each button as follows:

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

An initial configuration can then be expressed as a 25×1 column vector with the i^{th} entry 1 if the i^{th} button is lit and a 0 otherwise.

Example 2. *The initial configuration in the previous example,*

O	O			O
		O		
			O	
	O	O	O	
	O	O		

could be expressed as $[1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ ]'$

We will write the game as a linear system of equations $R\vec{s} = \vec{x}$, where $\vec{x}$ is the initial configuration written as a vector, $\vec{s}$ is the solution vector, and R is the button push matrix, which expresses the effects of each button push. R is a 25×25 matrix, where each row corresponds to a button on the board. In the i^{th} row, there is a one in every entry corresponding to a button that changes state when the i^{th} button is pushed. The row vector for the first button is

$$[1\;1\;0\;0\;0\;1\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\,],$$

while the row vector for the seventh button is

$$[0\;1\;0\;0\;0\;1\;1\;1\;0\;0\;0\;1\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\,].$$

The resulting R can be written more simply as a block matrix

$$\begin{bmatrix} A & I & O & O & O \\ I & A & I & O & O \\ O & I & A & I & 0 \\ O & O & I & A & I \\ O & O & O & I & A \end{bmatrix}$$

where O is the 5×5 zero matrix, I is the 5×5 identity matrix and

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

Example 3. *To see the utility of the button push matrix, consider the vector*

$$\vec{s} = [0\;0\;0\;0\;0\;0\;1\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\,]'.$$

If we compute $R * \vec{s}$, we see

$$R\vec{s} = [0\;1\;0\;0\;0\;1\;1\;1\;0\;0\;0\;1\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\,]',$$

the vector representing the initial state produced by pressing just the 7^{th} button.

We can write any $\vec{s}$ as the sum of at most 25 s_i vectors, where each s_i has only one non-zero entry. Each $R{*}s_i$ will be the vector representing the buttons that change state when the button corresponding to the non-zero entry of s_i is pressed. Then $R * \vec{s}$ will be the sum of all those products- or, the cumulative effects of each button push indicated by a non-zero i^{th} entry. Based on this, our solution vector will dictate the game strategy played; if there is a 1 in the i^{th} entry, the i^{th} button should be pushed. Thus, given some initial state vector $\vec{x}$, we can determine if the system is solvable by testing for the existence of some vector $\vec{s}$ such that $R\vec{s} = \vec{x}$. If such a vector exists, we can translate that into a solution on the game board by pressing each button with a 1 in the corresponding entry in $\vec{s}$.

An advantage of writing the game written as a linear system is that it allows us to easily consider other sized boards or games with slightly different rules. Research has been done on larger square boards and rectangular boards of various dimensions. In the next section, we will present previous work on both the 5×5 game board as well as rectangular boards. In section 3, we will extend the solution methods given in section 2 and present results on the solvability of some sizes of rectangular game boards. We will also discuss the proportion of solvable initial states. We will give suggestions for further work in section 4.

2. Survey of Previous Approaches

Once the game is translated to a system of equations, there are a number of ways to work towards finding a solution to the game. For our purposes, a solution for some board size will be a complete classification of the board's solvability. We say a game is entirely solvable if every initial state of lit buttons can be transformed to an all-off state. Otherwise, we can consider the dimension of the null-space of the matrix as a measure of the solvability, since we will show that the dimension of the null space gives the proportion of initial states that are solvable. In the following sections, we discuss a few possible solution methods and their applicability to further questions.

2.1. **Direct Row Reduction.**

The most straightforward solution method is, of course, row reduction on R, the button push matrix. For the 5×5 game board, R is 25×25. Generally, for an $k \times n$ board, R will be $nk \times nk$. Thus, these matrices can be somewhat unwieldy, although the row reduction is easily accomplished with Matlab. Direct row reduction is perhaps the easiest way to determine solvability for any given game board. It also provides a basis for the null space, which has been used in the 5×5 case to create an algorithm for determining the solution vector $\vec{s}$ needed to win a game.

Anderson and Feil [1] use direct row reduction to describe a winning strategy in terms of an orthogonal basis for the null space. They first note that some initial state $\vec{x}$ will be winnable if and only if $\vec{x}$ is orthogonal to all the vectors in the basis. The 5 board, which has a null space of dimension 2, has basis vectors

$$n_1 = [0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ ],$$

$$n_2 = [1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ ].$$

Then for any given $\vec{x}$ with solution $\vec{s}$, we can find 3 other solutions: $\vec{s} + \vec{n_1}$, $\vec{s} + \vec{n_2}$, and $\vec{s} + \vec{n_1} + \vec{n_2}$.

In order to actually find the solutions, let E be a row-reduced form of R, so that the first 23 rows and columns form the 23×23 identity matrix. Also, let F be the 25×25 matrix such that $FR = E$ and take $\vec{x}$ to be some winnable initial state. Then we can re-write the system $R\vec{s} = \vec{x}$ as $E\vec{s} = F\vec{x}$. We know that there are four possible vectors $\vec{s}$ that will satisfy this equation. Note that we can always choose an $\vec{s}$ such that the last two elements of $\vec{s}$ are zero- since we are working modulo 2, our choice of n_1 and n_2 guarantees that one of $\vec{s}$, $\vec{s} + \vec{n_1}$, $\vec{s} + \vec{n_2}$, and $\vec{s} + \vec{n_1} + \vec{n_2}$ will have zeros as the final two entries. In this case, however, $E\vec{s}$ will just be $\vec{s}$ itself, since E is fully reduced. Therefore, we have $\vec{s} = F\vec{x}$ as our first solution and we can generate the other 3 by adding the basis vectors.

Creating an algorithm that finds solutions is a clear goal for the 5×5 case, since the problem is a direct translation of an actual game board. As we extend the problem to other dimensions, however, we will not continue to consider algorithms for finding $\vec{s}$; rather, we will attempt to determine when such an $\vec{s}$ will exist. The method presented by [1] would certainly apply to other sizes of game boards, but without the motivation of actual game boards, the solution algorithms provide little interest. Direct row reduction is, however, the most efficient way to find the solvability for a given board size using Matlab.

One of the drawbacks of the direct method is that it makes generalization difficult. The row reduction must be performed again for each new size of board, so patterns may be obscured and the effort may become prohibitive. Also, although Matlab simplifies computations, the very large R matrices produced by larger game boards can again obscure the causes of patterns in solvability . Martín-Sánchez and Pareja-Flores [5] give a solution method that corrects the second concern by reducing the large $nk \times nk$ system into an equivalent $n \times n$ system and eventually allows us to determine solvability using a polynomial function of A.

2.2. **Martín-Sánchez and Pareja-Flores.**

Martín-Sánchez and Pareja-Flores [5] consider the Lights Out game as both a logical problem and a linear algebra problem. In treating it as a logical problem, they explain condensing the board down to the bottom row. From there, they find the redundant or neutral solutions, finally developing an algorithm to first reduce any starting state to a problem involving only the bottom row and then to predict what buttons need to be pressed to solve the last row.

In treating the game as a linear algebra problem, [5] attempts to mimic this condensing method mathematically. The approach begins with our system, $R\vec{s} = \vec{x}$. By observing that for any 25×25 matrix J, the equation $J\vec{s} = (R + J)\vec{s} + \vec{x}$ is equivalent to $R\vec{s} = \vec{x}$ and by judicious choice of J to be

$$\begin{bmatrix} O & I & O & O & O \\ O & O & I & O & O \\ O & O & O & I & O \\ O & O & O & O & I \\ O & O & O & O & O \end{bmatrix}$$

where O represents the 5×5 zero matrix and I is the 5×5 identity matrix, we can develop a new system that is easily simplified. If we consider $\vec{s}$ to be

$$\begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{bmatrix}$$

where each s_i is a 5×1 subvector, then $J\vec{s} = (R + J)\vec{s} + \vec{x}$ becomes

$$\begin{bmatrix} s_2 \\ s_3 \\ s_4 \\ s_5 \\ 0 \end{bmatrix} = \begin{bmatrix} A & 0 & 0 & 0 & 0 \\ I & A & 0 & 0 & 0 \\ 0 & I & A & 0 & 0 \\ 0 & 0 & I & A & 0 \\ 0 & 0 & 0 & I & A \end{bmatrix} * \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{bmatrix} + \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

where $\vec{x}$ is written similarly to $\vec{s}$. Performing the operations on the right side of the equation gives

$$\begin{bmatrix} s_2 \\ s_3 \\ s_4 \\ s_5 \\ 0 \end{bmatrix} = \begin{bmatrix} A * s_1 + x_1 \\ s_1 + A * s_2 + x_2 \\ s_2 + A * s_3 + x_3 \\ s_3 + A * s_4 + x_4 \\ s_4 + A * s_5 + x_4 \end{bmatrix}.$$

Thus, s_2 can be written in terms of s_1 and x_1. Continuing this substitution gives us solutions to s_3, s_4 and s_5 entirely in terms of s_1 and $\vec{x}$. We are given $\vec{x}$, so we can solve the larger system by solving for s_1. Note that finding a vector s_1 that solves this system and finding such an $\vec{s}$ for the full system are truly equivalent, since any s_1 can be used to re-build the corresponding $\vec{s}$ by reversing the substitution. Conversely, clearly any $\vec{s}$ gives an s_1, since s_1 is simply the first 5 components of $\vec{s}$. We can see that this correspondence is unique by noting that $\vec{s}$ only exists if it satisfies the relationships given in the above matrix. Thus, there cannot be 2 distinct $\vec{s}$ with the same first five components.

If we perform the substitution and simplify, we can build a new system of equations that will allow us to solve for s_1,

$$\begin{bmatrix} s_2 \\ s_3 \\ s_4 \\ s_5 \\ 0 \end{bmatrix} = \begin{bmatrix} B_1 & B_0 & 0 & 0 & 0 & 0 \\ B_2 & B_1 & B_0 & 0 & 0 & 0 \\ B_3 & B_2 & B_1 & B_0 & 0 & 0 \\ B_4 & B_3 & B_2 & B_1 & B_0 & 0 \\ B_5 & B_4 & B_3 & B_2 & B_1 & B_0 \end{bmatrix} * \begin{bmatrix} s_1 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}.$$

Where the B_i's are derived from the substitutions and can be given by $B_0 = I$, $B_1 = A$, and $B_i = B_{i-2} + A * B_{i-1}$ for $i \geq 2$. Note that these B_i are just polynomials of A. The final row of the system, $0 = B_5 * s_1 + B_4 * x_1 + B_3 * x_2 + B_2 * x_3 + B_1 * x_4 + B_0 * x_5$ will give us the solution to s_1. The entries of all matrices are in $\mathbf{Z}_2$, so this equation is equivalent to

$$B_5 * s_1 = B_4 * x_1 + B_3 * x_2 + B_2 * x_3 + B_1 * x_4 + B_0 * x_5.$$

[5] refer to the right side of the equation above as the gathers of $\vec{x}$. Since we are given $\vec{x}$ for any particular system, $gathers(\vec{x})$ is just a 5×1 vector. By condensing the original system down to $B_5 * s_1 = gathers(\vec{x})$, we have a 5×5 system that is far easier to reduce. Row reduction on B_5 gives us that the null space of B_5 has dimension 2.

Since B_5 is a symmetric matrix, $gathers(\vec{x})$ must be orthogonal to the null space of B_5 in order to be solvable. Actually finding the solution entails noting first that any solution can generate three other solutions, by adding one or both of the vectors in a basis for the null space. A basis for the null space is

$$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix},$$

so at least one of the possible solutions will have zeros as the last two entries. Therefore, if E is the row reduced form of B_5, with the 3×3 identity matrix as the first three rows and columns of E, and X is the 5×5 matrix such that $E = X * B_5$, we can consider the system $E * s_1 = X * gathers(\vec{x})$. If we assume that s_1 represents the solution with zeros as the 4th and 5th entries, then $E * s_1 = s_1$, so the first solution is simply $X * gathers(\vec{x})$ and the others can be generated by adding the vectors in the null space.

The solution to the 5×5 lights out game given by [5] is far simpler than working with the complete system. It also provides a more intuitive method, since the linear algebra aligns with the actual game strategy of gathering the lights down to the 5th row and finding a final solution based on those 5 lights. Moreover, it gives a simple algorithm for computing the dimension of the null space, and thus the number of solvable states, for larger games. Although finding the dimension of the null space of a given board is simple using Matlab, we will at times need more detailed information about the button push matrices. The approach given in [5] reduces the $nk \times nk$ button push matrix generated by an $k \times n$ board into an $n \times n$ system, which is easier to work with by hand. This allows for exploration of patterns in the dimensions of the null space and subsequent generalizations to solvability. In particular, the $n \times n$ matrix used in the gathers formula depends only on A, the $n \times n$ matrix representing the effects of pushing each button in a row on the other buttons in the row. By considering multiple iterations of the formula for the B_i matrices, we can begin to examine the general solvability of games of size $k \times n$, with fixed n and varying k.

Although the solution given by [5] may make it easier to find the solvability of an $k \times n$ system, there is a major disadvantage. For any $k \times n$ game board, we need to compute B_k. Without an explicit formula, this must be done iteratively, which is time consuming for larger values of k. It is possible to find an expression for B_k using Matlab, but this provides information only for specific cases. An explicit formula, however, would allow for some generalization in conclusions on which values of k form solvable game boards. Work done by Goldwasser et al. [2], [3] on an analogous problem in computer science has produced some results which, when applied to the B_i matrices, give an explicit formula and some opportunities for generalization.

2.3. **Goldwasser et al.**

Goldwasser et al. take another approach to the $k \times n$ board. Their first paper [2] considers the Lights Out problem from a computer science perspective; the original motivation for the work is the nine tails problem, which is analogous to the 3×3 case. They show that in the 3×3 case every initial configuration is solvable by noting that for each individual button i, there is a set of button pushes that change button i's state without changing any of the other buttons. By combining these sets of button pushes, any initial configuration can be solved.

A number of results and theorems are then presented in [2] concerning the null-space of a general $nk \times nk$ button push matrix— that is, a button push matrix for any $k \times n$ board. Many of these results are fairly specific to their method, which we discuss only briefly. The following, however, concerning the number of solvable initial states, will be used later.

Theorem 1. *If the dimension of the null space of R is j, then $\frac{1}{2^j}$ of the possible configurations are solvable.*

Proof. We are working on an $k \times n$ board, so each initial configuration is a $nk-$ *dimensional* vector. Each entry can be either 1 or 0, so there are 2^{nk} possible initial configurations. The dimension of the row space is $nk - j$, so there are 2^{nk-j} solvable configurations. Thus, $\frac{2^{nk-j}}{2^{nk}}$ or $\frac{1}{2^j}$ of possible configurations are solvable.

□

In [2] the authors also present a method for finding the dimension of the null space of an $k \times n$ board. We begin by defining a null space matrix as the $k \times n$ matrix formed from a vector $\vec{x}$ in the null space of R, the button push matrix.In a null space matrix, the first row corresponds to the first n components of $\vec{x}$, while the second row corresponds to the next n components. In terms of the game, starting from a blank game board, if one pressed every button with a one as its corresponding matrix entry, the game board would again have all the lights off. If no such matrix exists, the null space is empty and the board is completely solvable. Finding these matrices, then, is one way to test the solvability of any given size board.

To find the null space matrices, we first fix the number of rows, n, and look for the values of k, the number of columns, for which there is at least one non-trivial nullspace matrix. This is done by selecting a $1 \times n$ vector $\vec{w}$ with entries in the binary field from any of the 2^n possible vectors, using $\vec{w}$ as the first row of a potential nullspace matrix N and then using substitution— similarly to the method given in [5] — to find subsequent rows. If, for some k, the k^{th} row is all zeros, the matrix consisting of $\vec{w}$ and the following rows is a null space matrix for the $n \times (k-1)$ case. Although we will not use this method for rectangular boards, finding the null space matrices led the authors to write the equation for each row of a potential null space matrix as a sequence of Fibonacci polynomials. The results given in [3] on Fibonacci polynomials can be applied to the B_i matrices from [5].

Fibonacci polynomials are series of polynomials defined recursively as $f_0(x) = 0$, $f_1(x) = 1$, and $f_i(x) = xf_{i-1}(x) + f_{i-2}(x)$. Since all operations in [3] are all performed modulo 2, the following results hold in the binary field.

First, [3] gives an explicit formula for f_{n+1}:

$$f_{n+1}(x) = \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n-i}{i} x^{n-2i}$$

We will also use the following lemma:

Lemma 1. *Let $f_0, f_1, f_2, ...$ be the sequence of Fibonacci polynomials over $GF(2)$. Then*

(1) *f_i is an odd function for i even and an even function not divisible by x for i odd, $i > 0$.*

(2) $f_{n-t} + f_{n+t} = xf_nf_t$ *for* $0 \leq t \leq n$

(3) $f_{2n} = xf_n^2$, $n \geq 0$

(4) $f_{2n+1} = f_n^2 + f_{n+1}^2$

(5) $f_{mn}(x) = f_m(x)f_n(xf_m(x))$

(6) $f_{2mn-p} = xf_{mn}f_{mn-p} + f_p$

(7) $f_{2mn+p} = xf_{mn}f_{mn+p} + f_p$

We will prove only the first three parts.

Proof. The first statement follows inductively from the recursive of the f_i polynomials. We will show an analogous result in Lemma 2 below. To prove (2), fix some n. If $t = 0$, we have $f_n + f_n = 2 * f_n = 0 = xf_nf_0$. For $t = 1$, $f_{n+1} + f_{n-1} = xf_n = xf_nf_1$, so the base case holds. Now, assume that the relationship holds for all $t \leq m - 1 < n$, for some m. Then we have

$$f_{n+m} + f_{n-m} = (xf_{n+m-1} + f_{n+m-2}) + (xf_{n-m+1} + f_{n-m+2})$$

where the second half, $f_{n-m} = xf_{n-m+1} + f_{n-m+2}$ follows from $f_{n-m+2} = xf_{n-m+1} + f_{n-m}$. Then we have

$$f_{n+m} + f_{n-m} = x(f_{n+(m-1)} + f_{n-(m-1)}) + (f_{n+(m-2)} + f_{n-(m-2)})$$

$$f_{n+m} + f_{n-m} = x(xf_nf_{m-1}) + (xf_nf_{m-2}) = xf_nf_m.$$

Finally, the third statement is simply the case where $n = t$, so

$$f_{2n} = f_{n+n} + f_{n-n} = xf_n^2.$$

□

Given the drawbacks to using direct row reduction to determine solvability of an $k \times n$ board, we are primarily interested in extending the method presented by [5] to other sizes of game boards. In order to overcome the difficulties of computing the B_i matrices recursively, we will eventually use results from [2], [3] to simplify the computation of the B_i matrices and to allow for easy generalization on the solvability of an $k \times n$ board for varying k.

3. Generalizations to Arbitrary $\mathbf{k} \times \mathbf{n}$ Boards

For specific values of n and k, it is straightforward to use the method outlined by [5] to determine the solvability of the $k \times n$ board. Recall that the A matrices depend only on n, since the A matrix represents the effects of pushing each button in a row only on the other buttons in the row. Each row will have n buttons, so A will be $n \times n$. In R, the A matrix will then appear k times, down the main diagonal. We can then use the second system, $J\vec{s} = (R + J)\vec{s} + \vec{x}$, where J has $n \times n$ identity matrices just above the main diagonal, with $n \times n$ zero matrices everywhere else. This simplified system leads to the definition of the B_i matrices as polynomials of A, defined recursively by $B_0 = I$, $B_1 = A$, and $B_i = B_{i-2} + A * B_{i-1}$ for $i \geq 2$. The last row of the system then gives us $B_k * \vec{s_1} = gathers(\vec{x})$, so in order to determine the solvability, we need to compute the null space of B_k.

Our goal, however, is to categorize solvability for fixed n columns and varying k rows. In this case, we need to determine the dimension of the null space of $B_i(A_n)$ for many values i. This step towards generalizing the B_i matrices usually requires writing the B_i matrices as Fibonacci polynomials. The $k \times 2$ case, however, simplifies nicely, so we will first consider $k \times 2$ game boards as an example of the application of Martín-Sánchez's approach for rectangular boards.

3.1. $k \times 2$ **Boards.** First, we will specifically consider the 4×2 case, representing a game board with 8 buttons. If we number the buttons as follows,

1	2
3	4
5	6
7	8

we can create the 8×8 button push matrix.

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

We can write this in its simplified form as

$$\begin{bmatrix} A & I & 0 & 0 \\ I & A & I & 0 \\ 0 & I & A & I \\ 0 & 0 & I & A \end{bmatrix}, A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Now, we proceed in exactly the same way as in the 5×5 case. We need to solve the system $R\vec{s} = \vec{x}$ or, equivalently, $J\vec{s} = (R + J)\vec{s} + \vec{x}$. By choosing J as

$$\begin{bmatrix} 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix}, \text{we get the system} \begin{bmatrix} s_2 \\ s_3 \\ s_4 \\ 0 \end{bmatrix} = \begin{bmatrix} A & 0 & 0 & 0 \\ I & A & 0 & 0 \\ 0 & I & A & 0 \\ 0 & 0 & I & A \end{bmatrix} * \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} + \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}.$$

Recall that this system allows us to write the components s_2, s_3, and s_4 in terms of s_1 and $\vec{x}$ which in turn gives the equation $B_4 * s_1 = gathers(\vec{x})$. The B matrices are defined in exactly the same way as in the 5×5 case, so $B_4 = A^4 + A^2 + I$. Note, however, that A^2 is the 2×2 zero matrix, so $B_4 = I$. Thus, the dimension of the null space is 0 and every starting position is solvable. From this result, we can generalize to the $2k \times 2$ case.

Theorem 2. *On a $2k \times 2$ board, $k \in \boldsymbol{Z}$, every initial state is solvable.*

Before we prove this we need the following lemma.

Lemma 2. *For all $k \in \boldsymbol{Z}$, B_{2k} will be a sum of even powers of A and the identity matrix, while B_{2k+1} will be a sum of odd powers of A.*

Proof. We will prove this by induction; if k=0, B_0 and B_1 have been defined to be I and A respectively. Every B_i is defined to be the sum of $B_{(i-2)}$ and $A * B_{(i-1)}$. Thus, $B_2 = A^2 + I$ and $B_3 = A^3 + A$ and the base case is satisfied. Now, assume that for some $k \in \mathbf{Z}$, B_{2k} can be written as the sum of even powers of A and I and B_{2k+1} is the sum of odd powers of A. $B_{2(k+1)} = B_{2k} + A * B_{2k+1}$, so the exponent of every term of B_{2k+1} will become even, while the identity matrix and even powers of A in B_{2k} will remain unchanged. Similarly, B_{2k+3} will be the sum of B_{2k+1}, which contains only odd powers, and $A * B_{2(k+1)}$ which will also be only odd powers of A. □

Using this lemma, the proof of the theorem is straightforward.

Proof of Theorem 2. For any $2k \times 2$ game board, the solvability will be given by B_{2k}. By the previous lemma, B_{2k} will be the sum of even powers of A and the identity matrix; we have already noted that $A_2^2 = 0_2$, so $B_{2k} = I$ and every initial state is solvable. □

Corollary 1. *On a $(2k+1) \times 2$ board, the B_{2k+1} matrix will not be invertible, so some initial states will have no solution.*

The proof of this corollary follows directly from Theorem 2 and Lemma 2. If B_{2k+1} includes an A, the dimension of the null space will be 1. Otherwise, $B_{2k+1} = 0_2$, so the dimension of the null space will be 2.

The $k \times 2$ board is easy to categorize because of the simplicity of A_2 and its powers. Larger values of n lead to more complex A_n matrices that do not simplify as easily. Writing the B_i matrices as Fibonacci polynomials of A, however, allows us to apply the results given by [2], [3] to the B_i matrices.

3.2. **Fibonacci Polynomials and $\mathbf{B_i}$.**

Recall that Fibonacci Polynomials are defined recursively as $f_0(x) = 0$, $f_1(x) = 1$, and $f_i(x) = xf_{i-1}(x) + f_{i-2}(x)$. We have defined B_i matrices as $B_0 = I$, $B_1 = A$ and $B_i = B_{i-2} + A * B_{i-1}$, so we see that the B_i matrices are indeed Fibonacci polynomials with $B_i = f_{i+1}$. Then, using the explicit formula given by [3], we can write each B_k as

$$B_k(A) = \sum_{i=0}^{\lfloor k/2 \rfloor} \binom{k-i}{i} A_n^{k-2i}$$

where A_n is the A matrix corresponding to the $k \times n$ case. The parity of $\binom{n-i}{i}$ determines which terms will appear, since if it is even, the term will disappear and if it is odd, the coefficient of A^{n-2i} will be 1. Thus, we can write:

$$B_n(A) = \sum_{i=0}^{\lfloor n/2 \rfloor} J(n-i, i) A^{n-2i}$$

where

$$J(n,k) = \begin{cases} 1 & : \binom{n}{k} \text{ odd} \\ 0 & : \binom{n}{k} \text{ even} \end{cases} \quad n, k \in \mathbf{Z}^+, n \geq k$$

This formula greatly expedites determining the solvability for a given n and k, since it removes the recursive quality of the definition and allows for computation in one step. We will also use Lemma 1. Recall that if $f_0, f_1, f_2, ...$ is a sequence of Fibonacci polynomials over $GF(2)$, then

(1) f_i is an odd function for i even and an even function not divisible by x for i odd.

(2) $f_{n-t} + f_{n+t} = xf_nf_t$ for $0 \leq t \leq n$

(3) $f_{2n} = xf_n^2$, $n \geq 0$.

Example 4. *The Fibonacci polynomials we will be concerned with are, of course, our B_i matrices. Thus, consider $B_3 = A^3 + 2A = A^3$. Recall that if we write B_3 as a Fibonacci polynomial, we have $B_3 = f_4$. By (3), $f_4 = x * f_2^2$. Translating this back to the B_i matrices gives $B_3 = A * (A^2) = A^3$.*

Using the lemma, we can begin to generalize solvability for fixed n and varying k.

3.3. $\mathbf{k \times n}$ Boards.

In determining solvability, we are most concerned with whether the B_k matrix is invertible. Knowing the invertibility of any A^i, however, will not necessarily impact whether B_k will be invertible, since the sum of invertible matrices may not be invertible. Lemma 1 , however, gives us f_{2n} as the product of A and f_n, since our Fibonacci polynomials are polynomials of A. We know that the product of invertible matrices will be invertible, which allows us to apply this to our B_i matrices. First, we need to be able to categorize A_n by invertibility in order to use it in the f_{2n} formula.

Lemma 3. *A_n is invertible if and only if $n \not\equiv 2 \bmod 3$.*

Proof. We will show this by induction. For the base case, the lemma is easily tested for $n = 1$, $n = 2$ and $n = 3$. For our induction step, we will show that if, for some n, A_{3n-3} and A_{3n-2} are invertible while A_{3n-1} is not, then $|A_{3n}| = 1$, $|A_{3n+1}| = 1$, and $|A_{3n+2}| = 0$. First, consider $|A_{3n}|$.

First, using cofactor expansion on the first row for any m, we can write $|A_m| = |A_{m-1}| + |A_{m-2}|$. Thus, $|A_{3n}| = |A_{3n-1}| + |A_{3n-2}|$. We have $3n - 1 \equiv 2 \bmod 3$ and $3n - 2 \equiv 1 \bmod 3$, so by assumption, $|A_{3n}| = 0 + 1 = 1$ and A_{3n} is invertible. Similarly,

$$|A_{3n+1}| = |A_{3n}| + |A_{3n-1}| = 1 + 0 = 1,$$

$$|A_{3n+2}| = |A_{3n+1}| + |A_{3n}| = 1 + 1 = 0,$$

and therefore, A_{3n+1} is invertible while A_{3n+2} is not. □

Note that if A_n is invertible, then A_n^i is invertible for all i. Therefore, in any case where we know A_n is invertible, we can apply the relationship $f_{2n} = xf_n^2$ to generate an infinite number of entirely solvable $k \times n$ boards.

Theorem 3. *If, for some n and k, the $k \times n$ case is entirely solvable and A_n is invertible, then the $(2k + 1) \times n$ case is also entirely solvable.*

Proof. We know that $B_k(A_n)$ is invertible, by assumption. Note that with our labeling, $B_k(A_n) = f_{k+1}(A_n)$. By Lemma 1 (3) $f_{2k+2} = A_n f_{k+1} f_{k+1}$. This, however, is equivalent to $A_n * B_k * B_k$. Thus, $B_{2k+1} = f_{2k+2} = A_n * B_k * B_k$. A and B_k are invertible by assumption, so B_{2k+1} is invertible by assumption and therefore, the $(2k+1) \times n$ case is entirely solvable. □

By continuing to double the number of columns, we see that if the $k \times n$ case is entirely solvable and A_n is invertible, then the $(4k+2+1) \times n$, $(8k+4+2+1) \times n$, and, in general, boards of size $(2^j k + \sum_{i=0}^{j-1} 2^i) \times n,\ j \geq 1$ cases are also solvable.

Although this theorem gives us relatively select information for any fixed n, we can also apply it more usefully by fixing k, with the following lemma.

Lemma 4. *The $k \times n$ case has the same solvability as the $n \times k$ case.*

Proof. Recall that we generated R, the button push matrix by considering the effect of pushing each button. Our labeling of the buttons, however, was essentially arbitrary. The $k \times n$ board is really just a rotated $n \times k$ board, so the corresponding R matrix will be generated by relabeling the buttons; the relationships between them, however, will remain the same. We can move from R_n, the button push matrix for the $k \times n$ case to R_k, the matrix for the $n \times k$ case through a series of row and column exchanges. Thus, both matrices will have the same null space and therefore will represent systems with identical solvability. □

Example 5. *The 3×2 board can be numbered as*

1	*2*
3	*4*
5	*6*

yielding the button push matrix.

$$R_2 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

Rotating the board, however, gives the following relabeling:

1 → 1	*3→2*	*5→ 3*
2→ 4	*4→ 5*	*6 → 6*

These translate into the following column (and similarly, row) exchanges

$$R_2 = \begin{bmatrix} \vec{c_1} & \vec{c_2} & \vec{c_3} & \vec{c_4} & \vec{c_5} & \vec{c_6} \end{bmatrix} \rightarrow \begin{bmatrix} \vec{c_1} & \vec{c_3} & \vec{c_5} & \vec{c_2} & \vec{c_4} & \vec{c_6} \end{bmatrix}.$$

We can perform the given exchanges, first on the columns of R_2,

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix},$$

and then, on the rows:

$$\begin{bmatrix}1&1&0&1&0&0\\1&0&0&1&1&0\\1&1&1&0&1&0\\0&1&0&1&1&1\\0&1&1&0&0&1\\0&0&1&0&1&1\end{bmatrix} \rightarrow \begin{bmatrix}1&1&0&1&0&0\\1&1&1&0&1&0\\0&1&1&0&0&1\\1&0&0&1&1&0\\0&1&0&1&1&1\\0&0&1&0&1&1\end{bmatrix}.$$

This final matrix is exactly the button push matrix generated by the 2×3 *case.*

With this lemma, we can use the sequences of solvable cases generated in the last theorem to make broader statements about the solvability of game boards for some values of n.

Theorem 4. *For* $j \geq 0$*, the* $n \times (\sum_{i=0}^{j} 2^i)$ *game board is solvable for* $n \not\equiv 2 \bmod 3$

Proof. We know that for $n \not\equiv 2 \bmod 3$, A_n is invertible. Recall that $A_n = B_1(A_n)$. Thus, for $n \not\equiv 2 \bmod 3$, the $1 \times n$ board is entirely solvable. By the previous theorem, the $(\sum_{i=0}^{j} 2^i) \times n$ board will be entirely solvable for $j \geq 1$. This, however, is equivalent to the $n \times (\sum_{i=0}^{j} 2^i)$ case being entirely solvable. □

This theorem gives us, among other results, that the $n \times 3$ board is entirely solvable whenever $n \not\equiv 2 \bmod 3$

3.4. **Solvability Patterns for n $\geq$ 4.** In addition to using Fibonacci polynomials to determine patterns in solvability, we can calculate the solvability

n	$k \times n$ **solvable for**:
4	$k \not\equiv 4 \text{ mod } 5$
5	$k \equiv 0, 4 \text{ mod } 6$,
6	$k \not\equiv 8 \text{ mod } 9$
7	$k \not\equiv 2 \text{ mod } 3$
8	$k \not\equiv 6 \text{ mod } 14$
9	$k \equiv 0, 1, 3, 6, 7, 10, 12, 13 \text{ mod } 15$
10	all k
11	$k \equiv 0, 4 \text{ mod } 6$
12	all k
13	$k \not\equiv 2 \text{ mod } 3$
14	$k \equiv 0, 2, 6, 8 \text{ mod } 10$, $k \not\equiv 16 \text{ mod } 17$
15	$k \not\equiv 2 \text{ mod } 3$
16	all k
18	all k
19	$k \equiv 0, 1, 3, 6, 7, 10, 12, 13 \text{ mod } 15$
20	$k \equiv 0 \text{ mod } 2$, $k \not\equiv 8 \text{ mod } 9$, $k \not\equiv 44 \text{ mod } 64$
21	$k \not\equiv 2 \text{ mod } 3$
22	all k
23	$k \equiv 0, 4 \text{ mod } 6$
24	$k \not\equiv 4 \text{ mod } 5$
25	$k \not\equiv 2 \text{ mod } 3$
26	$k \equiv 0, 2, 4, 8, 10, 12 \text{ mod } 14$
27	$k \not\equiv 2 \text{ mod } 3$
28	all k
30	$k \not\equiv 10 \text{ mod } 11$, $k \not\equiv 30 \text{ mod } 31$

FIGURE 1. Entirely solvable boards for various values of n

for fixed n and various values of k using Matlab. The program used is given in Appendix B. This has led to conjectures about when the $k \times n$ board will be solvable. These results are summarized in Figure 1. Some of these cases, such as $n = 7$ and $n = 15$ follow from the theorem on Fibonacci polynomials. Other cases are far more complex. In order to determine these patterns, we have found the dimension of the null space for the first 200 values of k, starting with $k = 0$. For $n = 14$, these values are

0	1	0	2	4	1	0	2	0	5	0	2	0	1	4	2	**8**	1	0	6
0	1	0	2	4	1	0	2	0	5	0	2	0	**9**	4	2	0	1	0	6
0	1	0	2	4	1	0	2	0	5	**8**	2	0	1	4	2	0	1	0	6
0	1	0	2	4	1	0	**10**	0	5	0	2	0	1	4	2	0	1	0	6
0	1	0	2	**12**	1	0	2	0	5	0	2	0	1	4	2	0	1	0	6
0	**9**	0	2	4	1	0	2	0	5	0	2	0	1	4	2	0	1	**8**	6
0	1	0	2	4	1	0	2	0	5	0	2	0	1	4	**10**	0	1	0	6
0	1	0	2	4	1	0	2	0	5	0	2	**8**	1	4	2	0	1	0	6
0	1	0	2	4	1	0	2	0	**13**	0	2	0	1	4	2	0	1	0	6
0	1	0	2	4	1	**8**	2	0	5	0	2	0	1	4	2	0	1	0	6

Based on this, we predict that for $k \equiv 0, 2, 6, 8 \bmod 10$, $k \not\equiv 16 \bmod 17$, the $k \times n$ board will be entirely solvable. The patterns of solvability for $n = 14$, however, turn out to be relatively simple. If $n = 29$, we see that the dimensions of the null spaces follow a pattern that repeats in sets of 120, except for elements congruent to 15 and 16 mod17, which differ between the repeated sets. In other words, a $k \times 29$ board has the same solvability as a $(k+120) \times 29$ board, $k \not\equiv 15, 16, \text{mod}17$. Within each set of 120 values of k, however, the values for which the null space has dimension zero do not follow a clear pattern. Taking $n = 17$ produces similar results, with the dimensions of the null spaces appearing to repeat in sets of 167.

There is also a great deal of variance in the actual dimensions of the null spaces for $k \times n$. Using the dimension of the null space, we will examine the proportion of solvable initial states for various game boards. [2] gives Theorem 1 on the proportion of solvable initial states. We can derive the same result, however, by considering the number of unique strategies. By definition, an initial state $\vec{x}$ is solvable if and only if there exists some $\vec{s}$ such that $R\vec{s} = \vec{x}$. Thus, the number of solvable $\vec{x}$ is less than or equal to the number of $\vec{s}$, since we do not know that each $R\vec{s}$ is unique. Therefore, an upper bound on the number of solvable initial states for an $k \times n$ board is 2^{nk}, since $\vec{s}$ has nk elements in $\mathbf{Z}_2$. In order to find the exact number of solvable initial configurations, $\vec{x}$, we need to determine how many of these $\vec{s}$ vectors are unique.

If j is the dimension of the null space of the $nk \times nk$ button push matrix, we can find an orthogonal basis for the null space, $\{\vec{r_1}, \vec{r_2}, ..., \vec{r_j}\}$. Using this basis, we can proceed similarly to the solution algorithm given by [1]. If we choose some $\vec{s}$, we know that $R\vec{s}$ gives us a solvable initial state $\vec{x}$. We can then compute

$$R(\vec{s} + \vec{r_i}) = R\vec{s} + R\vec{r_i} = R\vec{s} + \vec{0} = \vec{x}.$$

So adding any one of the basis vectors to $\vec{s}$ produces another strategy vector that solves the same initial state, $\vec{x}$. This holds, of course, for multiple basis vectors — for example, $R(\vec{s} + \vec{r_1} + \vec{r_2} + \cdots + \vec{r_j}) = \vec{x}$. Thus, any linear combination of the basis vectors added to $\vec{s}$ produces a non-distinct strategy vector. To count the number of these non-distinct vectors, note that we have $\vec{s}$ itself, j vectors of the form $\vec{s} + \vec{r_i}$, $\binom{j}{2}$ vectors of the form $\vec{s} + \vec{r_i} + \vec{r_k}$ and so on. Therefore, given any $\vec{s}$, there are $\sum_{i=1}^{j} \binom{j}{i}$ other non-distinct strategy vectors, or, if we include $\vec{s}$, a collection of

$$\sum_{i=0}^{j} \binom{j}{i}$$

vectors, all of which solve the same initial configuration.

Finally, consider two strategy vectors, $\vec{s}$ and $\vec{t}$ such that there is no combination of null space basis vectors such that $\vec{s} + \sum_{i=1}^{k} \vec{r_i} = \vec{t}$. We wish to show that $\vec{s}$ and $\vec{t}$ are distinct. Since we have an orthogonal basis for the null space, we know that we can choose some combination of basis vectors such that when added to $\vec{s}$, the sum is a non-distinct vector $\vec{s'}$ with zeros as the last j entries. We can find another such combination that produces $\vec{t'}$ with zeros as the last j entries. By definition, $\vec{s'} \neq \vec{t'}$. Then, if E is the fully row-reduced matrix equivalent to R, with F the $nk \times nk$ matrix such that $FR = E$, we have $R\vec{s'} = E\vec{s'} = \vec{s'}$ and $R\vec{t'} = E\vec{t'} = \vec{t'}$. Therefore, $R\vec{s} = R\vec{s'} \neq R\vec{t'} = R\vec{t}$, so $\vec{t}$ and $\vec{s}$ are distinct.

We have shown that the set of all strategy vectors, $\vec{s}$ can be partitioned into groups of $\sum_{i=0}^{j} \binom{j}{i}$ strategy vectors, where each group corresponds to a distinct solvable initial state. Therefore, there are

$$\frac{2^{nk}}{\sum_{i=0}^{j} \binom{j}{i}}$$

solvable initial states and

$$\frac{1}{\sum_{i=0}^{j} \binom{j}{i}}$$

of the possible initial states are solvable. Of course, $\sum_{i=0}^{j} \binom{j}{i} = 2^j$, so these results agree with those presented in [2]. We can now quantify the solvability of a board as the fraction of initial states which are solvable, with a 1 corresponding to an entirely solvable board.

Recall that we have previously shown that on an $k \times n$ game board, the null spaces of B_k and R will have the same dimension. Using this, and Matlab programs to compute the dimensions of the null space of B_k for various values of $k \times n$, we have produced the following results on the number of solutions on some sizes of game board.

One question that has come up frequently in previous work is which square boards will be solvable. Using the program given in Appendix A, we have found the results given in Figure 2 for solvability on square boards. We can create similar charts for fixed values of n and varying k. Some values, such as $n = 2$ display clear patterns, as in Figure 3.

This pattern in solvability follows from our original theorem on the $k \times 2$ cases. Clearly, for $2k \times 2$, the board is entirely solvable, so our solvability value is 1. For $2k + 1 \times 2$, we have the following theorem.

Theorem 5. *The* $4k + 1 \times 2$ *game board has solvability* $\frac{1}{2}$ *while the* $4k - 1 \times 2$ *game board has solvability* $\frac{1}{4}$.

n	**Dim(null space)**	**Solvability**:
2	0	1
3	0	1
4	4	1/16
5	2	1/4
6	0	1
7	0	1
8	0	1
9	8	1/256
10	0	1
11	6	1/64
12	0	1
13	0	1
14	4	1/16
15	0	1
16	8	1/256
17	2	1/4
18	0	1
19	16	1/65536
20	0	1

Figure 2. Solvability of $n \times n$ boards

k	**Dim(null space)**	**Solvability**:
1	1	1/2
2	0	1
3	2	1/4
4	0	1
5	1	1/2
6	0	1
7	2	1/4
8	0	1
9	1	1/2
10	0	1
11	2	1/4

Figure 3. Solvability on $k \times 2$ boards

Proof. The base cases follow from the chart above, with $k = 0$ and $k = 1$. Now, assume the theorem holds for all $k \leq n - 1$ for some n. $B_{4n-1} = A * B_{4n-2} + B_{4n-3}$ by definition; recall from previous theorems and calculations that $B_{4n-2} = I$ and B_{4n-3} will either be A or 0. Since $B_{4n-3} = B_{4(n-1)+1}$ by assumption, we know B_{4n-3} has solvability $\frac{1}{2}$ and therefore, a null space of dimension 1. Thus, $B_{4n-3} = A$. We now have $B_{4n-1} = A + A = 0_2$, so B_{4n-1} has a 2 dimensional null space, giving $4k - 1 \times 2$ solvability $\frac{1}{4}$. Similarly, we can write $B_{4n+1} = A * B_{4n} + B_{4n-1} = A + 0_2 = A$, so B_{4n+1} has a one dimensional null space and the $4k + 1 \times 2$ has solvability $\frac{1}{2}$. □

There are similar patterns for other values of n, including 3 and 4. In some cases, however, the solvability is difficult to predict. We summarize solvability for many values of n and k in Figure 4.

4. Future Directions

There are a number of areas of future work, both in determining the solvability of $k \times n$ rectangular game boards and also in other areas of the Lights Out game. As is indicated by Figure 4 and Figure 1, many values of n produce very complicated patterns of solvability. Applications of Fibonacci polynomials may well be able to clarify those cases. In this work, we have used only one result from research on Fibonacci polynomials, to prove the relationship between the $k \times n$ and $2k + 1 \times 2$ cases. Doubtless, there are other results that

k	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 11$
2	1/4	1	1/2	1	1/4	1	1/2	1/4
3	1	1	1/8	1	1	1/4	1	1/8
4	1	1/16	1	1	1	1	1/16	1
5	1/8	1	1/4	1	1/16	1/2	1/2	1/16
6	1	1	1	1	1	1/64	1	1
7	1	1	1/16	1	1	1/4	1	1/128
8	1/4	1	1/2	1/64	1/4	1	1/2	1/4
9	1	1/16	1/2	1	1	1/2	1/256	1/2
10	1	1	1	1	1	1	1	1
11	1/8	1	1/16	1	1/128	1/4	1/2	1/64
12	1	1	1	1	1	1	1	1
13	1	1	1/2	1	1	1/128	1	1/2
14	1/4	1/16	1/2	1	1/4	1	1/32	1/4
15	1	1	1/16	1	1	1/4	1	1/256
16	1	1	1	1	1	1	1	1
17	1/8	1	1/4	1/64	1/16	1/2	1/2	1/16
18	1	1	1	1	1	1	1	1
19	1	1/16	1/8	1	1	1/4	1/256	1/8
20	1/4	1	1/2	1	1/4	1/64	1/2	1/4
21	1	1	1/2	1	1	1/2	1	1/2
22	1	1	1	1	1	1	1	1
23	1/8	1	1/32	1	1/128	1/4	1/2	1/1024
24	1	1/16	1	1	1	1	1/16	1
25	1	1	1/2	1	1	1/2	1	1/2
26	1/4	1	1/2	1/64	1/4	1	1/2	1/4
27	1	1	1/8	1	1	1/256	1	1/8
28	1	1	1	1	1	1	1	1
29	1/8	1/16	1/4	1	1/16	1/2	1/512	1/16
30	1	1	1	1	1	1	1	1

FIGURE 4. Solvability for various values of n

could be applied to the B_i matrices. The fraction of solvable initial states also indicates a need for further research. Fibonacci polynomials may not apply here, although the fraction can be determined using the B_i matrices. The difficulty here rests in the formation of the B_i matrices. Since each is the sum of various powers of A, we cannot determine the dimension of the null space purely by looking at the A_n^m matrices. Finding other expressions for the B_i matrices may be important to proving the patterns seen in the fraction of solvable states. Another possibility is using the button push matrices instead of the smaller $B_i's$. This may be computationally more difficult, but would remove the problem of sums.

Another variation on Lights Out is considering different rules, rather than different game board sizes. For example, if the game board was a torus, pushing a button on one edge would change the state of a button on the other edge. This would result in a slightly different button push matrix, but would allow for very similar solution methods. Alternatively, one could consider a game where the buttons had more than two possible states- for example, if the buttons could be off, red, blue, or green. This case is less analogous in terms of solution methods, because the additional state means that operations must be performed modulo 4. $\mathbf{Z}_4$, however, is not a field, complicating the matrix operations. In this case, Smith Normal Form decomposition might be useful, since it relies on the matrices having elements from principle ideal domains, rather than fields.

Appendices

Appendix A. Square Boards

This program determines the dimension of the null space for the first p $n \times n$ boards, starting with $n = 1$. It does so by computing the B_n and reducing it mod2.

```
p=100;

NULL=zeros(2,p);

for m=1:p

n=m;

X= eye(n,n);

W=eye(n-1,n-1);

z=zeros(1, n); %row vector

q=zeros(n-1,1); %column vector

Y= [ z ; W q] ;

T= [q W; z] ;

A= X + Y + T ;

D=zeros(n,n^2+n); % D will store the B matrices

D(:,1:n)=X;

D(:,(n+1):2*n)=A;

D(:,(2*n+1):3*n)=mod(X+(A*A), 2);
```

```
for i=3:(n)

    D(:,(i*n +1):(i+1)*n)=

     mod( D(:,((i-2)*n +1):(i-1)*n)+ A*D(:,((i-1)*n +1):(i)*n), 2) ;

end

B=D(:,((n)*n +1):n^2+n); %defines B_n as the last n columns of D

R=zeros(n,n);

R=B;

E=eye(n,n);

H=eye(n,n);

for j=1:n % This loop performs row reduction modulo 2

[C,I] = max(R(j:n,j));

F=eye(n,n);

F(j,:)=E((I+j-1),:);

F((I+j-1),:)=E(j,:);

R=mod(R*F,2);

H=mod(H*F,2);

for i=(j+1):n

    if R(i,j)>0,

    R(i,:)=mod(R(j,:)+R(i,:),2);

    H(i,:)=mod(H(j,:)+H(i,:),2);
    end

end

end

R; % R is the fully reduced matrix equivalent to B

H; % H is a matrix such that HB=R

V=null(R);

size(V);

NULL(1,m)=m;

NULL(2,m)=m-rank(R);

end

NULL
```

Appendix B. Fixed n Solvability

This program fixes n and varies k from 1 to p, printing the dimension of the null space as well as the value of k in the matrix NULL. Again, in order to determine the dimension of the null space, for each k, the program finds B_k and then reduces mod2.

```
p=100;

 n=11;

NULL=zeros(2,p);
```

```
D=zeros(n,n*p+n); % D stores the B matrices

X= eye(n,n);

W=eye(n-1,n-1);

z=zeros(1, n);

q=zeros(n-1,1);

Y= [ z ; W q] ;

T= [q W; z] ;

A= X + Y + T ;  % these steps form the A matrix

D(:,1:n)=X;

D(:,(n+1):2*n)=A; D(:,(2*n+1):3*n)=mod(X+(A*A), 2);

for i=3:p

    D(:,(i*n +1):(i+1)*n)=

      mod( D(:,((i-2)*n +1):(i-1)*n)

        + A*D(:,((i-1)*n +1):(i)*n), 2);

end

for m=1:p

    B=D(:,((m-1)*n +1):(m*n));

    R=zeros(n,n);

    R=B;

    E=eye(n,n);

    H=eye(n,n);
```

```
        for j=1:n % performs row reduction modulo 2
            [C,I] = max(R(j:n,j));
            F=eye(n,n);
            F(j,:)=E((I+j-1),:);
            F((I+j-1),:)=E(j,:);
            R=mod(R*F,2);
            H=mod(H*F,2);
                for i=(j+1):n
                    if R(i,j)>0,
                       R(i,:)=mod(R(j,:)+R(i,:),2);
                       H(i,:)=mod(H(j,:)+H(i,:),2);

                    end
                end
        end
R; %R here is the fully reduced matrix equivalent to B_n
H; % H is a matrix that satisfies H*B= R
V=null(R);
size(V);
NULL(1,m)=m-1;
NULL(2,m)=n-rank(R);
```

```
end

NULL
```

References

[1] M. Anderson and T. Feil, Turning Lights Out with Linear Algebra, *Mathematics Magazine*, 71:4 (Oct. 1998), 300-303

[2] J. Goldwasser, W. Klostermeyer, G. Trapp, and C.Q. Zhang, Setting Switches on a Grid, Technical Report 95-20, Department of Statistics and Computer Science, WVU, 1995, manuscript.

[3] J. Goldwasser, W. Klostermeyer, G. Trapp, Characterizing Switch Setting Problems, *Linear and Multilinear Algebra*, vol. 43:1-3, (1997), 121-136

[4] A. T. Benjamin, J. Quinn, *Proofs that Really Count: The Art of Combinatorial Proof*, The Mathematical Association of America, (2003)

[5] O. Martín-Sánchez and C. Pareja-Flores, Two Reflected Analysis of Lights Out, *Mathematics Magazine*, 74:4 (2001), 295-304