A Minimal Time Solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2

Kathryn Boddie

University of Wisconsin-Milwaukee

Follow this and additional works at: https://dc.uwm.edu/etd

Part of the Applied Mathematics Commons, and the Computer Sciences Commons

Recommended Citation
https://dc.uwm.edu/etd/2161

This Dissertation is brought to you for free and open access by UWM Digital Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of UWM Digital Commons. For more information, please contact open-access@uwm.edu.
A MINIMAL TIME SOLUTION TO THE FIRING SQUAD SYNCHRONIZATION PROBLEM WITH VON NEUMANN NEIGHBORHOOD OF EXTENT 2

by

Kathryn A. Boddie

A Dissertation Submitted in
Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy
in Mathematics

at
The University of Wisconsin-Milwaukee
August 2019
Cellular automata provide a simple environment in which to study global behaviors. One example of a problem that utilizes cellular automata is the Firing Squad Synchronization Problem, first proposed in 1957. This paper provides an overview of the standard Firing Squad Synchronization Problem and a commonly used technique in solving it. This paper also provides a statement of a new extension of the Standard Firing Squad Synchronization Problem to a different neighborhood definition - a Von Neumann neighborhood of extent 2. An 8 state 651 rule minimal time solution to the extended problem is described, presented and proven, along with Python code used in running simulations of the solution.
# TABLE OF CONTENTS

1 Introduction to The Firing Squad Synchronization Problem  
   1.1 Background on Cellular Automata  1  
   1.2 The Standard FSSP  2  
   1.3 The Halving Strategy  4  
   1.4 Waksman’s 16-state Solution  5  
   1.5 Balzer’s and Noguchi’s 8-state Solutions  10  
   1.6 Mazoyer’s 6 State Minimal Time Solution  12  

2 A New Extension to the FSSP  16  
   2.1 Problem Statement  16  
   2.2 Length of a Minimal Time Solution  17  

3 A Minimal Time Solution  19  
   3.1 Motivation  19  
   3.2 Description of the Solution  19  
   3.3 Transition Tables for the Solution  24  

4 Proof of Minimal Time Solution  28  
   4.1 Image Solutions  28  
   4.2 Proof of 651 rule 8 state Minimal Time Solution  31  

5 Conclusion and Suggestions for Future Work  67  

6 Simulation Program Code  69  

REFERENCES  97  

Curriculum Vitae  99
LIST OF FIGURES

1.1 Length 11 Squad Simulation - Waksman Solution ........................................ 6
1.2 Length 22 Squad Simulation - Noguchi Solution ........................................... 10
1.3 Length 25 Squad Simulation - Mazoyer Solution ........................................... 13

2.1 Neighborhood Comparison ............................................................................. 17

3.1 Boddie Solution - \( n = 75 \) ........................................................................ 20
3.2 Visualizing the Intersections of Waves .......................................................... 22
3.3 Boddie Solution - \( n = 24 \) ........................................................................ 24
3.4 Boddie Solution - \( n = 25 \) ........................................................................ 25

4.1 Examples of Closed Curves ........................................................................... 29
4.2 Length 16 Squad Simulation, \( i = 11 \) ............................................................... 38
4.3 Length 16 Squad Simulation, \( i = 12 \) ............................................................... 39
4.4 Induction for backwards wave - \( N + 1 \) even .................................................. 43
4.5 Induction for backwards wave - \( N + 1 \) odd ..................................................... 44
4.6 First “slow” wave meeting the backwards wave along \( D_1 \) and \( D_2 \) .......... 45
4.7 Second “slow” wave meeting \( D_1 \) and \( D_2 \) ....................................................... 46
4.8 \( q + 1 \)st Slow Wave meeting \( D_1 \) and \( D_2 \) for \( q \geq 2 \) ........................................ 47
4.9 Finding Middle Points with \( z_j = 0 \) ................................................................. 55
4.10 Finding Middle Points with \( z_j = 1 \) ................................................................. 56
4.11 Diagonals for Lemma 4 if \( z_i = 0 \) ................................................................. 57
4.12 Diagonals for Lemma 4 if \( z_i = 1 \) ................................................................. 58
4.13 \( z_i = 1 \) and \( z_{i+1} = 0 \) reflection of waves ..................................................... 60
4.14 \( z_i = 1 \) and \( z_{i+1} = 1 \) reflection of waves ..................................................... 62
## LIST OF TABLES

1.1 Waksman’s Transition Tables ......................................................... 8
1.2 Noguchi’s Transition Tables .......................................................... 11
1.3 Mazoyer’s Transition Tables ........................................................... 15

3.1 Transition Tables for a Minimal Time Solution to the FSSP with Von Neu-
mann Neighborhood of Extent 2 ....................................................... 25

4.1 Additional Rules for $R_1$ ................................................................. 31
ACKNOWLEDGMENTS

I would like to thank my entire committee, Dr. Suzanne Boyd, Dr. Christine Cheng, Dr. Istvan Lauko, Dr. Jeb Willenbring, and with special thanks to Dr. Bruce Wade for supporting me and encouraging me to pursue applied mathematics and cellular automata.

I would also like to thank Nicholas Boddie for being there and supporting me through everything. Your never-ending support and encouragement is what brought me to this point. Thank you.
1 Introduction to The Firing Squad Synchronization Problem

1.1 Background on Cellular Automata

A finite automaton is a machine which follows a set of fixed rules, changing its state according to an evolution in discrete time. By *state* we mean the behavior of the machine at that time step. The states can be represented by numbers, or as a description of some property of the machine. For example, in modeling disease, a “state” could be “infected” or “vaccinated.” Finite automata can be used to model complex systems in areas such as language recognition and biological modeling, among others [3].

One type of network of finite automata is called *cellular automata*. Cellular automata consist of an $d$-dimensional array of finite automata, a definition of local neighborhood, and a fixed set of rules. Each finite automaton, often called a *cell*, uses the rule set to update its state given its current state and the states of its neighbors. One common neighborhood type found in various cellular automata is the Von Neumann neighborhood. The *Von Neumann neighborhood* of a particular cell consists of all cells with taxi-cab distance 1 from the given cell. Recall that the *taxi-cab distance* is defined by

$$D(p, q) = \sum_{i=0}^{d-1} |p_i - q_i|$$

where $p = (p_0, \ldots, p_{d-1})$ and $q = (q_0, \ldots, q_{d-1})$ are cells in the $d$-dimensional array.

Cellular automata are interesting to study, not because of the simple rule set, but because of the varieties of global behavior that can be observed across the entire array [3]. Some examples of problems using cellular automata, including the Firing Squad Synchronization Problem (FSSP), have consistent global behavior, regardless of the size of the array, even though the system runs only on local information [3]. In the case of the FSSP, globally the array synchronizes with every cell in the same state, regardless of the size of the array.
1.2 The Standard FSSP

The Firing Squad Synchronization Problem was originally proposed in 1957 by John Myhill [3]. The standard FSSP uses a finite one dimensional array with Von Neumann neighborhoods in which each cell in the cellular automata is identical, except possibly the cells at the ends [3]. The end cells could be different due to the fact that they only have neighboring cells on one side, rather than on both sides. All cells begin in a state called the Quiescent state, except for one of the end cells, usually taken to be the left end cell, which is in a state referred to as the General. Then, using a fixed finite number of states and a finite rule set involving those states, each finite automaton in the array is updated based on the current state of that cell and the current states of its two neighboring cells [3].

In the standard FSSP, there is a required rule that a Quiescent cell with two Quiescent neighbors must remain in the Quiescent state in the next time step [3]. This condition guarantees that there is no spontaneous state-changing activity - in order for a cell to change states, some form of information must be received by the finite automaton in question. The end goal of the standard problem is to have every finite automaton in the array, regardless of the number of cells in the array, enter a special state referred to as Fire at the same discrete time step, with no cell entering this Fire state prior to that time step [3]. Having every cell enter this Fire state at the same time step with no prior firings is called synchronizing the array. Ideally, this should be done in the smallest number of discrete time steps and with the least amount of cell states.

A solution to the FSSP is a fixed finite rule set using a fixed finite set of states which synchronizes all cells in the finite one dimensional array of length \( n \) for any size \( n \in \mathbb{N} \). The set of states used in the FSSP can differ by solution, but all solutions have the states of General, Quiescent and Fire amongst others. A minimal time solution is a solution
in which all cells synchronize in the smallest number of discrete time steps possible. In the case of the standard FSSP, a minimal time solution is one that finishes in $2n - 2$ time steps, where $n$ is the length of the one dimensional array [3]. The first minimal time solution for the standard FSSP was found in 1962 by Goto and had over one thousand states in its fixed set of states [4]. Shortly thereafter, the number of required states for a minimal time solution was drastically reduced. In 1966, Waksman produced a 16 state minimal time solution and in 1967 Balzer independently produced an 8 state minimal time solution, both using similar ideas involving the so-called halving strategy [1, 12]. In 1987, Mazoyer presented a minimal time solution to the standard FSSP using only 6 states in its fixed set of states [6]. Mazoyer’s 6 state minimal time solution, which used a technique of dividing the one dimensional array into thirds rather than halving, remains the minimal time solution with least number of states discovered as of today [3, 6]. Many years later, in 2004, Noguchi examined Waksman’s and Balzer’s minimal time solutions and, using the same halving strategy, presented a different 8 state minimal time solution which used fewer rules than Balzer’s previous 8 state solution [7]. Although Noguchi’s 8 state minimal time solution uses two more states than Mazoyer’s 6 state minimal time solution, it required the same number of rules (119) [7]. All of the above mentioned are true minimal time solutions which work for any finite one dimensional array, regardless of the length $n$ of the array.

Work has also been done to find the minimum number of states required in a minimal time solution for the standard FSSP. Balzer wrote an algorithm and claimed to show there was no 4 state minimal time solution, but the algorithm was found to be flawed at a later date [1]. However, Sanders reworked the algorithm in 1993 and the program indicated, through a brute-force simulation of every possible combination of arranging 4 states into rules, that there was no 4 state minimal time solution, although a formal proof was not provided [8]. In 1999, Settle proved that no 3 state solution to the standard FSSP exists, regardless if considering only minimal time solutions or not, and formally proved the non-existence of 4
state minimal time solutions with three particular properties [9]. Yunès discovered a 4-state non-minimal time partial solution for squads of size \( n = 2^k \) which requires \( 2^{k+1} - 1 \) time steps [13]. Umeo and Yanagihara discovered a 5-state non-minimal time partial solution for squads of size which requires \( 3n - 3 \) time steps [11]. It is unknown whether a complete 5-state solution (either minimal time or not) exists, however Settle also proved that there is no minimal time 5-state solution which has six particular properties [3, 9].

1.3 The Halving Strategy

In 1966 and 1967, Waksman and Balzer independently found minimal time solutions to the standard Firing Squad Synchronization Problem, with 16 and 8 states, respectively, using the same technique of the halving strategy [1, 12]. Later, in 2004, Noguchi found a different 8-state minimal time solution with different, and fewer, rules than Balzer [7].

The general idea of the halving strategy is to repeatedly find the middle cell (if \( n \) is odd) or middle cells (if \( n \) is even) of the one-dimensional array of \( n \) finite automata. In general, the finding of the middle cell(s) is achieved by generating a “fast” signal and a “slow” signal, both generated by the original General [1, 2, 12]. These signals can be visualized as a “wave” starting from the General and propagating cell by cell across the array. The “fast” signal travels from the General to the finite automaton at the opposite end of the array. When this “fast” signal reaches the far end, that end cell enters the General state and then sends a return “fast” signal back to the original General. On the return path, the “fast” signal then intersects the “slow” signal at the middle of the array [1, 2, 12]. When the “fast” and “slow” waves intersect, the middle cell(s) enters the General state. At this time, we have the original General on the left end, the General created on the far right end, and the General or Generals in the middle.
After the first division, the same technique is repeated on the two smaller sub-arrays of size \(\left\lceil \frac{n}{2} \right\rceil\), using the new *Generals* to generate new “fast” and “slow” signals, to find the middle cell(s) of the sub-arrays \([1, 2, 12]\). This is continued iteratively until no more divisions can be made. At each point of division - the half, quarter, eighth and so on points of division, the cell(s) enters the *General* state \([1, 2, 12]\). After the array has been successively divided completely, every finite automaton will be in the *General* state and then the following discrete time step will result in all cells simultaneously entering the *Fire* state \([1, 2, 12]\).

### 1.4 Waksman’s 16-state Solution

Waksman’s minimal time solution to the standard FSSP uses 16 states in its fixed set of states and its rule set contains a rule that many other solutions also use - each cell will enter the *Fire* state if and only if its current state is *General* and both its left and right neighbors are also in the *General* state (or in the case of being on the end of the array, that its only neighbor is also in the *General* state) \([12]\).

Waksman’s solution uses three types of signals- the *A* signal, which moves one cell per unit of time, the *R* signal, which is generated by the *A* signals and move in the opposite direction one cell per unit of time, and the *B* signal, which initially moves one cell per every three units of time after encountering an *R* signal \([12]\).

Waksman *A* signals are represented by eight different \(A_{ijk}\) states, four for a variation which generates an *R* signal with no delay and four for a variation which generates an *R* signal after a one time unit delay\([12]\). Different \(A_{ijk}\) states are used to denote whether the signal is moving to the right or the left and also indicating whether the cell being occupied is an even or odd distance from the *General* which generated it \([12]\). The subscript \(i \in \{0, 1\}\) represents the time delay in generating an *R* signal. The subscript \(j \in \{0, 1\}\) represents the direction.
the signal is traveling across the array with 0 representing left and 1 representing right. The subscript $k \in \{0, 1\}$ represents whether the current finite automaton is an even ($k = 0$) distance or odd ($k = 1$) distance from the General which generated it - starting the count on the generating machine. For instance, a finite automaton directly next to the General which generated the $A_{ijk}$ signal received would have the subscript $k = 0$ - the General itself is the first machine in this count and the cell receiving the signal is the second [12]. For example, a finite automaton in state $A_{011}$ indicates that this cell will have a 0 unit time delay in generating an $R$ signal, that this $A$ signal is moving to the right, and that the cell
containing this signal is an odd distance from the General which generated the A signal [12].

The solution has two different $R$ signal states, $R_0$ and $R_1$ indicating the direction of movement - 0 represents movement to the left and 1 represents movement to the right [12]. There also are two different $B$ signal states, one which moves forward upon meeting an $R$ signal and allows the $R$ signal to continue, $B_0$, and one which moves forward upon meeting an $R$ signal without allowing the $R$ signal to continue, $B_1$ [12]. The first $B$ signal will remain in one position for three units of time prior to meeting an $R$ signal, due to the $A$ signals having a delay in generating $R$ signals every other time step [12]. Upon meeting an $R$ signal, a $B$ signal will switch from its current type of $B$ signal state to the other type of $B$ signal state. Since only one of the $B$ signal states allows the $R$ signals to continue to propagate, this results in progressively “slower” $B$ signals, with the $k$-th $B$ signal remaining in place $2^{k+1} - 1$ units of time prior to meeting an $R$ signal [12].

Waksman’s solution is unusual in that it has two different General states, which he denotes as $P_0$ and $P_1$ and refers to as “preterminal states” [12]. We will refer to these two General states as $G_0$ and $G_1$, to more closely resemble the same notation of other solutions. The only difference between the different General states is which type of $A$ signal they generate - $G_i$ generates the $A$ signal with an $i$ time unit delay in $R$ signal generation. The remaining two states are the Fire, or “terminal state”, which we will denote as $F$, and the Quiescent state, denoted by $Q$ [12].

When an $A$ signal intersects a $B$ signal, a new General is generated at the intersection. These signals find the midpoint of the firing squad array as the $A$ signals move one cell per unit of time and the $B$ signals move one cell per three units of time, resulting in the signals meeting at the midpoint as the $A$ signal is returning from the endpoint opposite the original General, when the $A$ signal has traveled $\frac{3}{2}$ the length of the squad. Since the $B$ signal
travels at one third of the speed of the A signal, it has traveled \( \frac{1}{2} \) the length of the squad. Once a new General is created, the process repeats with the new General creating its own signals on the new sub-array.

Waksman’s transition tables are found below. Rows are labeled by left neighbor states, columns are labeled by right neighbor states. Note that * refers to the end of the array (no machine) and that X will stand for all other states not explicitly mentioned in that particular table. If only the main letter of a state with multiple variations is listed, then the rule applies for all variations of that state (ex: listing B instead of both \( B_0 \) and \( B_1 \)). For an example on how to read the tables, to find the new state of a machine currently in state \( Q \) with left neighbor \( R_1 \) and right neighbor \( Q \), move across the \( R_1 \) row until you reach the \( Q \) column and read that the new state should be \( R_1 \). This can be denoted by \((R_1, Q, Q) \rightarrow R_1\).

Table 1.1: Waksman’s Transition Tables

<table>
<thead>
<tr>
<th>Q</th>
<th>( A_{000} )</th>
<th>( A_{001} )</th>
<th>( A_{100} )</th>
<th>( A_{101} )</th>
<th>( A_{010} )</th>
<th>( A_{011} )</th>
<th>( A_{110} )</th>
<th>( A_{111} )</th>
<th>( B )</th>
<th>( R_0 )</th>
<th>( R_1 )</th>
<th>( G_0 )</th>
<th>( G_1 )</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Q )</td>
<td>( A_{001} )</td>
<td>( A_{000} )</td>
<td>( A_{101} )</td>
<td>( A_{100} )</td>
<td>( R_0 )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( R_0 )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( A_{000} )</td>
</tr>
<tr>
<td>( B )</td>
<td>( A_{001} )</td>
<td>( A_{000} )</td>
<td>( A_{101} )</td>
<td>( A_{100} )</td>
<td>( R_0 )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( R_0 )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( A_{000} )</td>
</tr>
<tr>
<td>( R_0 )</td>
<td>( A_{001} )</td>
<td>( A_{000} )</td>
<td>( A_{101} )</td>
<td>( A_{100} )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( A_{000} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( R_1 )</td>
<td>( A_{001} )</td>
<td>( A_{000} )</td>
<td>( A_{101} )</td>
<td>( A_{100} )</td>
<td>( R_1 )</td>
<td>( R_1 )</td>
<td>( R_1 )</td>
<td>( R_1 )</td>
<td>( R_1 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G_0 )</td>
<td>( B_0 )</td>
<td>( B_0 )</td>
<td>( A_{000} )</td>
<td>( A_{010} )</td>
<td>( Q )</td>
<td>( R_0 )</td>
<td>( Q )</td>
<td>( G_0 )</td>
<td>( G_0 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( G_1 )</td>
<td>( A_{000} )</td>
<td>( A_{110} )</td>
<td>( Q )</td>
<td>( R_0 )</td>
<td>( Q )</td>
<td>( G_0 )</td>
<td>( G_0 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{000} )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( Q )</td>
<td>( B_0 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{001} )</td>
<td>( Q )</td>
<td>( Q )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{100} )</td>
<td>( Q )</td>
<td>( Q )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{101} )</td>
<td>( Q )</td>
<td>( Q )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{010} )</td>
<td>( A_{011} )</td>
<td>( A_{011} )</td>
<td>( G_0 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{011} )</td>
<td>( A_{010} )</td>
<td>( A_{010} )</td>
<td>( G_1 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{110} )</td>
<td>( A_{111} )</td>
<td>( A_{111} )</td>
<td>( G_0 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( A_{111} )</td>
<td>( A_{110} )</td>
<td>( A_{110} )</td>
<td>( G_1 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>( G_0 )</td>
<td>( G_1 )</td>
<td>( G_0 )</td>
<td>( G_1 )</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_0$</td>
<td>$X$</td>
<td>$G$</td>
<td>$A_{000}$</td>
<td>$A_{100}$</td>
<td>$A_{001}$</td>
<td>$A_{101}$</td>
<td>$R_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----</td>
<td>----</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>-----</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$X$</td>
<td>$B_0$</td>
<td>$B_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$R_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G$</td>
<td>$B_0$</td>
<td>$G_0$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_0$</td>
<td>$R_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_{010}$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$R_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_{110}$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_0$</td>
<td>$R_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_{011}$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$R_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_{111}$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$R_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$B_1$</th>
<th>$X$</th>
<th>$G$</th>
<th>$A_{000}$</th>
<th>$A_{100}$</th>
<th>$A_{001}$</th>
<th>$A_{101}$</th>
<th>$R_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$B_1$</td>
<td>$G_0$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_0$</td>
<td>$Q$</td>
<td></td>
</tr>
<tr>
<td>$G$</td>
<td>$B_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$Q$</td>
<td></td>
</tr>
<tr>
<td>$A_{010}$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$Q$</td>
<td></td>
</tr>
<tr>
<td>$A_{110}$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_0$</td>
<td>$Q$</td>
<td></td>
</tr>
<tr>
<td>$A_{011}$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$Q$</td>
<td></td>
</tr>
<tr>
<td>$A_{111}$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$Q$</td>
<td></td>
</tr>
<tr>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td>$R_1$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$R_0$</th>
<th>$X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$Q_0$</td>
</tr>
<tr>
<td>$B_0$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$B_1$</td>
<td>$B_0$</td>
</tr>
<tr>
<td>$G$</td>
<td>$B_0$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$G_0$</th>
<th>$X$</th>
<th>$G$</th>
<th>$*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$G_0$</td>
<td>$G_0$</td>
<td>$G_0$</td>
</tr>
<tr>
<td>$G$</td>
<td>$G_0$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$*$</td>
<td>$G_0$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$G_1$</th>
<th>$X$</th>
<th>$G$</th>
<th>$*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$G_1$</td>
<td>$G_1$</td>
<td>$G_1$</td>
</tr>
<tr>
<td>$G$</td>
<td>$G_1$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$*$</td>
<td>$G_1$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A_{000}$</th>
<th>$X$</th>
<th>$P_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$Q$</td>
<td>$B_0$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A_{001}$</th>
<th>$X$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$Q$</td>
</tr>
<tr>
<td>$X$</td>
<td>$R_1$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A_{110}$</th>
<th>$X$</th>
<th>$B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$Q$</td>
<td>$R_0$</td>
</tr>
<tr>
<td>$X$</td>
<td>$Q$</td>
<td>$G_0$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A_{011}$</th>
<th>$X$</th>
<th>$B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$Q$</td>
<td>$R_0$</td>
</tr>
<tr>
<td>$X$</td>
<td>$Q$</td>
<td>$G_0$</td>
</tr>
</tbody>
</table>
1.5 Balzer’s and Noguchi’s 8-state Solutions

Both Balzer and Noguchi followed the same general strategy as Waksman, with slight changes. Balzer’s 8 state solution was found in 1967 independently of Waksman’s 16 state solution with the primary differences being that different signals were not used to indicate whether the current machine was an even or odd distance from the General which generated it and that there is only one General state \[ \square \]. Balzer’s 8 state minimal time solution used 182 rules in its transition tables.

Figure 1.2: Length 22 Squad Simulation - Noguchi Solution
In 2004, Noguchi found that Balzer’s solution had unnecessary rules in the transition tables [7]. He found that of the 182 rules listed in the tables, 17 were not used upon running the solution’s algorithm. Of these 17 unnecessary rules, 6 were used to run the solution with the General beginning on the other end of the array, 1 was for a firing squad of length 1, and the others were simply not used. After eliminating these unnecessary rules in the transition tables, Noguchi was able to further reduce the number of required rules by combining some rules and simplifying the relationship between the Generals and their neighbors, by utilizing two Quiescent states, two states for waves of the A type, and two states for B waves, resulting in a final minimal time solution using 8 states and only 119 rules [7]. Noguchi’s transition tables can be found below.

Table 1.2: Noguchi’s Transition Tables

<table>
<thead>
<tr>
<th>$Q_1$</th>
<th>$Q_1$</th>
<th>$A_1$</th>
<th>$A_2$</th>
<th>$B_1$</th>
<th>$B_2$</th>
<th>$G$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
</tr>
<tr>
<td>$A_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
</tr>
<tr>
<td>$B_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
</tr>
<tr>
<td>$B_2$</td>
<td>$Q_1$</td>
<td>$B_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$B_2$</td>
<td>$G$</td>
</tr>
<tr>
<td>$B_2$</td>
<td>$B_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$G$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$Q_2$</th>
<th>*</th>
<th>$Q_2$</th>
<th>$A_2$</th>
<th>$B_1$</th>
<th>$B_2$</th>
<th>$G$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td></td>
</tr>
<tr>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$B_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$B_2$</td>
<td></td>
</tr>
<tr>
<td>$B_1$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td></td>
</tr>
<tr>
<td>$B_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td></td>
</tr>
<tr>
<td>$G$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A_1$</th>
<th>$Q_1$</th>
<th>$Q_2$</th>
<th>$B_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
</tr>
<tr>
<td>$B_1$</td>
<td>$Q_1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2$</td>
<td>$B_1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A_2$</th>
<th>$Q_2$</th>
<th>$B_1$</th>
<th>$B_2$</th>
<th>$G$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$</td>
<td>$Q_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$B_1$</td>
<td></td>
</tr>
<tr>
<td>$B_1$</td>
<td>$Q_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2$</td>
<td>$B_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$B_2$</th>
<th>$Q_1$</th>
<th>$Q_2$</th>
<th>$A_1$</th>
<th>$B_2$</th>
<th>$G$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$</td>
<td>$B_2$</td>
<td>$A_1$</td>
<td>$G$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2$</td>
<td>$B_2$</td>
<td></td>
<td>$B_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_2$</td>
<td>$A_2$</td>
<td></td>
<td>$Q_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2$</td>
<td>$G$</td>
<td></td>
<td>$G$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G$</td>
<td>$B_2$</td>
<td>$Q_1$</td>
<td>$G$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$G$</th>
<th>*</th>
<th>$Q_1$</th>
<th>$Q_2$</th>
<th>$A_1$</th>
<th>$B_2$</th>
<th>$G$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$*$</td>
<td>$G$</td>
<td>$G$</td>
<td>$G$</td>
<td>$G$</td>
<td>$F$</td>
<td></td>
</tr>
<tr>
<td>$Q_1$</td>
<td>$G$</td>
<td>$G$</td>
<td>$G$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2$</td>
<td>$G$</td>
<td>$G$</td>
<td>$G$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_2$</td>
<td>$G$</td>
<td></td>
<td>$G$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2$</td>
<td>$G$</td>
<td></td>
<td>$G$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$G$</td>
<td>$F$</td>
<td>$G$</td>
<td>$G$</td>
<td>$G$</td>
<td>$G$</td>
<td>$F$</td>
</tr>
</tbody>
</table>
1.6 Mazoyer’s 6 State Minimal Time Solution

In 1987, Mazoyer discovered a 6 state minimal time solution to the FSSP, which remains the minimal time solution with least number of states today. Mazoyer also used a strategy of dividing the length \( n \) array into sections, however unlike Waksman, Balzer, and Noguchi, he divided the array into uneven sections, creating the first division point in such a way as to divide the array into two sections, with one twice as long as the other (up to one unit according to \( n \mod (3) \)) [6].

Mazoyer’s strategy for his 6 state minimal time solution to the FSSP involves iteratively dividing the length \( n \) firing squad array into thirds [6]. After creating the first division at \( \frac{2}{3}n \), the right part of the array is treated as an initial array of size \( \frac{n + i}{3} \) where \( i \in \{0, 1, 2\} \) [6]. This is iteratively repeated, creating divisions at \( \frac{2}{3}n \), \( \left(\frac{2}{3}\right)^2 n \), \( \left(\frac{2}{3}\right)^3 n \), and so on. By dividing the array into pieces as described, each sub-array will have its General at the left most position [6], unlike the previous solutions described where the General could be on the right or the left side of the sub-arrays.

It should be noted that Mazoyer counts his time steps differently than the previously mentioned papers. Mazoyer considers time step \( t = 1 \) to be the initial set up of the firing squad array, with the first machine on the left-hand side being in the General state and the remaining \( n - 1 \) machines in the Quiescent state, whereas the previously mentioned works consider the initial set up of the firing squad array to be time step \( t = 0 \). Because of this difference in initialization, Mazoyer’s paper [6] states that his solution finishes in \( 2n - 1 \) time steps, whereas using the initialization of the other papers, this would be \( 2n - 2 \) time steps. Here, we adjust the times mentioned in Mazoyer’s work to match with the other works in the field.
It is easiest to describe the overarching theme of the strategy using an ideal case where the array is perfectly divided each time with no conflicts involving remainders when dividing into thirds. In actuality, this does not happen, but this ideal case will still be described so that the general idea of the algorithm can be explained simply. Once divided at the $\frac{2}{3}n$ machine at $t = \frac{4}{3}n - 2$, the newly created sub-array consisting of the right third of the original array, from that point in time until completion, behaves similarly to the original length $n$ array’s behavior from the beginning of the algorithm until completion [6]. The remaining piece of the original array is also divided similarly to the original array, thus at the machine at spot $\frac{2}{3} \left( \frac{2}{3}n \right)$ [6]. Once divided at the machine at spot $\frac{2}{3} \left( \frac{2}{3}n \right)$, the process is repeated on the new remaining sub-array and the newly created sub-array, until the entire array is
In actual practice, with any integer $n$ as the length of the initial firing squad, the above described procedure has to be adjusted based on the remainder modulo 3 of $n$. Let $(i, j)$ indicate position $i$ at time $j$. The initial wave including the original General will occupy spots $(\ell, \ell - 1)$ and so its reflection will occupy spots $(\ell, 2n - \ell - 1)$ for $\ell \in \{1, \ldots, n\}$. Say that the new General, created around the $\frac{2}{3}n$ machine as described above, is formed at $(k, 2n - k - 1)$. This results in the reflection of the initial wave meeting the original General at time $(2n - k - 1) + (k - 1) = 2n - 2$, as needed for a minimal time solution. The newly created General will generate a new wave, with a delay depending on the remainder modulo 3 of the length of the original array. Say that the new General generates a new wave after a delay of $j$ time steps with $j \in \{0, 1, 2\}$. Then the new initial wave generated by this new General will start at $(k, 2n - k - 1 + j)$ and will travel to the right-most machine, reaching it at $(n, 2n - k - 1 + j + n - k)$, or simplified at $(n, 3n - 2k - 1 + j)$ [6]. This new wave, once reflected, would then return to the new General at time $3n - 2k - 1 + j + n - k = 4n - 3k - 1 + j$.

In order for the firing squad to be synchronized, the original wave must return to the original general at the same time that the new wave returns to the new general. So, it would be required that $2n - 2 = 4n - 3k - 1 + j$, so it must be that $3k = 2n + j + 1$, which is solvable provided that $j \in \{0, 1, 2\}$ [6].

Consider the length of the original array as $n = 3p + i$ with integer $p \geq 1$ and $i \in \{1, 2, 3\}$. If $i = 1$, then $n = 3p + 1$ and so $3k = 2(3p+1)+j+1=6p+3+j$. Thus $j = 0$ and so $k = 2p+1$. Similarly, if $i = 2$, we have $j = 1$, $k = 2p + 2$ and if $i = 3$, we have $j = 2$, $k = 2p + 3$ [6]. The variable $j$ will be called the delay for the new General’s activation. Regardless of the value of $j$, the new sub-array will have $p + 1$ machines, consisting of machines $k$ to $n$ with a delay of $j = i - 1$. 

Mazoyer constructed his 6 state minimal time solution by first creating an 8 state minimal time solution using the method described above. Then, he combined some rules and states to eliminate two of the least used states, resulting in his 6 state solution. His solution uses three states, $A, B, C$ to provide information in the waves in regards to how much of a delay each new General should have, as well as the General, Quiescent, and Fire states. The construction of the transition tables and the induction proof of the correctness of the solution, both of which are quite lengthy, can be found in [6]. Mazoyer’s transition tables can be found below.

Currently, there are many different minimal time solutions to the standard FSSP, many of which were not outlined here. The minimal time solution with the smallest number of states is currently Mazoyer’s 6 state solution outlined above [3, 6]. Settle proved there is no 3 state minimal time solution to the standard FSSP and Sanders claimed there is no 4 state minimal time solution to the standard FSSP using a brute-force computer simulation [8, 9]. It is currently unknown whether a 5 state minimal time solution to the standard FSSP can exist [3].
2 A New Extension to the FSSP

2.1 Problem Statement

We introduce a new extension to the Firing Squad Synchronization Problem using a different neighborhood definition than the standard problem.

Definition 1. The Von Neumann neighborhood of extent \( n \) of a cell is the set of all cells with taxi-cab distance no more than \( n \) from the given cell.

Consider an extension of the standard FSSP as follows. Let the firing squad be represented by a one dimensional array with Von Neumann neighborhoods of extent 2 in which each cell in the cellular automata is identical, except possibly the 4 cells, 2 at each of the ends, which will not have a full neighborhood consisting of other cells in the array. All cells begin in a state called the Quiescent state except for one of the end cells, which will be taken to be the left end cell, which is in a state referred to as the General. Then, using a fixed finite number of states and a rule set, each cell in the array is updated based on the current state of that cell and its four neighbors (the far left, the near left, the near right, and the far right neighbors).

In the standard FSSP, there is a required rule that a Quiescent cell with two Quiescent neighbors must remain in the Quiescent state in the next time step \[3\] and we extend this concept of no spontaneous state change activity by requiring that a Quiescent cell with four Quiescent neighbors must remain Quiescent in the next time step. The end goal of the problem is to have every cell in the array, regardless of the size of the array, enter a special state called Fire at the same time step, with no cell firing prior to that time step. Ideally, this should be done in the least amount of discrete time steps and with as few cell states as possible.
2.2 Length of a Minimal Time Solution

In the standard FSSP, the length of a minimal time solution is $2n - 2$, where $n$ is the length of the squad, due to the conditions of no spontaneous state changes without a received signal and Von Neumann neighborhoods of extent 1 which restrict waves of information to moving only 1 cell per time step. This results in a minimal time solution needing $2n - 2$ time steps since it is necessary for a signal from the General to travel $n - 1$ units to reach the opposite end of the firing squad and then return to the General - $(n - 1) + (n - 1) = 2n - 2$ time steps. It should be noted that, due to the neighborhood definition in the standard FSSP, no interior cell has information about whether either of its neighbors is an end cell.

In the FSSP with Von Neumann neighborhood of extent 2, two interior cells - the one directly to the right of the General and the one directly to the left of the end cell on the far end, depicted in orange in Figure 2.1 - have information about one of their 4 neighbors being an end cell. This is due to the radius of the neighborhoods being 2, so on one side of these particular cells, they have a cell as a neighbor and can “see” that there is no cell on the other side of their neighbor. This bit of information influences the time needed in a minimal time solution.

Figure 2.1: Neighborhood Comparison
Extent 1 neighborhood in green, extent 2 neighborhood in purple

In the FSSP with Von Neumann neighborhood of extent 2, two interior cells - the one directly to the right of the General and the one directly to the left of the end cell on the far end, depicted in orange in Figure 2.1 - have information about one of their 4 neighbors being an end cell. This is due to the radius of the neighborhoods being 2, so on one side of these particular cells, they have a cell as a neighbor and can “see” that there is no cell on the other side of their neighbor. This bit of information influences the time needed in a minimal time solution.

**Theorem 1** (Length of Minimal Time Solution to the FSSP with Von Neumann neighborhood of extent 2). A minimal time solution to the FSSP with Von Neumann neighborhood
of extent 2 synchronizes the firing squad in $n - 1$ time steps, where $n$ is the number of cells in the squad.

Proof. Assume $n$ is odd. Then $n - 1$ is even. The original signal, call it $A_1$, generated by the General must travel $n - 1$ cells to the far end. This signal can move 2 cells per time step at its quickest speed and so can reach the far end in $k = \frac{n - 1}{2}$ time steps. Similarly, on return to the General, the return signal, call it $A_2$, can travel the $n - 1$ cells in $k$ time steps resulting in a minimal time solution of $2k = n - 1$ time steps.

Now assume $n$ is even. Then $n - 1$ is odd. The original signal $A_1$ generated by the General can travel to the cell directly to the left of the far end cell in $\ell = \frac{n - 2}{2}$ time steps. In the next time step, the original signal will reach the end cell, but the cell directly to the left of this end cell is aware that its neighbor is an end cell and will thus send back a signal. So in that same time step the cell directly to the left of the far end cell can enter a state of $A_2$ for the return signal. This $A_2$ signal would need to travel another $n - 2$ cells back to the General, which can be done in another $\ell$ time steps. Thus a minimal time solution will take $2\ell + 1 = n - 1$ time steps. \qed
3 A Minimal Time Solution

3.1 Motivation

Our goal was to discover a minimal time solution to the Firing Squad Synchronization Problem with Von Neumann neighborhoods of extent 2 using the common technique of halving. Motivated by Waksman’s, Balzer’s, and Noguchi’s previous halving technique solutions to the standard problem, we were able to construct a similar halving technique solution performed at double speed. This was achieved by first creating a simulation program in python (source code found in Chapter 6) and beginning to create rules involving a “fast” wave and “slow wave” and halving using a small, but non-trivial, array of size \( n = 15 \). After successfully attaining a working minimal time simulation for size \( n = 15 \), additional rules were added successively in order for the solution to work for all odd \( n \), beginning with \( n = 3 \) and increasing in size. After 638 rules were included in the transition tables, no more rules were needed for odd \( n \) after size \( n = 211 \). Then an additional 13 rules were added in order to account for the differences that occur when \( n \) is even and the fast return wave begins one discrete time step “early.” At this stage, the simulation program confirmed that the solution was a minimal time solution for a firing squad of any size up to and including \( n = 500 \), as well as a few choice large \( n \) examples including \( n = 17, 675 \) and \( n = 1,357,899 \). Once these large examples were confirmed through simulation, we moved on to the formal proof of the solution.

3.2 Description of the Solution

We will describe a 651 rule, 8 state, minimal time solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2 that uses the technique of halving.
Let $S$ be the set of states for the cellular automata where $S = \{G, Q_1, Q_2, A_1, A_2, B_1, B_2, F\}$.

$G$ represents the General state, $Q_i$ are Quiescent states, $A_i$ are waves generated by a General that move at the maximum speed of 2 cells per time step, $B_i$ are waves that are generated and travel at slower speeds (as will be described), and $F$ represents the terminal Fire state.

At time $t = 0$, the 1-dimensional array of length $n$ representing the firing squad has the first cell, cell 0, in state $G$ and the remaining cells, cells 1 through $n - 1$, in state $Q_1$. 

Figure 3.1: Boddie Solution - $n = 75$
The general idea of the minimal time solution follows the halving method. The cell in position 0 starts in the state $G$ and emits a forward signal $A_1$ which moves at the maximal speed of two cells per time step. If $n$ is odd, then the $A_1$ signal will reach the cell in position $n - 1$ after $\frac{n - 1}{2}$ time steps, and at that time the cell in position $n - 1$ will enter the $G$ state. This new $G$ will then send a return signal $A_2$ which moves at the maximal speed of two cells per time step. While this is occurring, the cell at position 0 in state $G$ will also emit slower moving $B_i$ signals which will serve to intersect the return signal $A_2$ at the $\frac{n - 1}{2}$, $\frac{n - 1}{4}$, $\frac{n - 1}{8}$ and so on division points of the firing squad array. At each intersection point, the cell or cells at the intersection will enter state $G$ and the procedure is repeated on the smaller sub-array. As the sub-arrays are repeatedly halved, the sub problems will eventually involve small enough sub-arrays to be trivial, at which time, $t = n - 1$, all cells in the entire array will synchronize in the $F$ state. The only difference with an even $n$ is that if $n$ is even, then the next-to-last machine in position $n - 2$ will enter the $A_2$ state in the same time step as the cell in position $n - 1$ will enter the $G$ state - essentially beginning the return wave one time step “early.”

The first $B_i$ wave will travel 2 machines every 3 times steps in order to intersect the return wave $A_2$ at the $\frac{n - 1}{2}$ division point. This can be visualized by imagining a space-time diagram of our firing squad with the initial states of the cells at time $t = 0$ printed in a row and the states of the cells in each successive time step printed in rows below the initial row. As the solution is a minimal time solution, there will be $n$ rows in this diagram - one for each of the $n - 1$ times steps and one for the initial set up at $t = 0$. Thinking of placing this diagram on the Cartesian plane in Quadrant 1 with the bottom left corner at the origin, the return $A_2$ wave can be modeled by the linear equation $y = \frac{1}{2}x$. The first $B_i$ wave can be modeled by the linear equation $y = -x + (n - 1)$, and these two lines intersect when $x = \frac{n - 1}{2}$. Successive $B_i$ waves are emitted and will move 2 cells every $2^{m+1} - 1$ time steps,
thus modeled by the linear equation \( y = -\frac{2^{m+1} - 1}{2} x + (n - 1) \) to meet the return wave \( A_2 \) at the \((n - 1) \left(\frac{1}{2}\right)^m\) division point.

![Figure 3.2: Visualizing the Intersections of Waves](image)

The initial \( A_1 \) wave emitted from the General will emit a “backwards” \( A_1 \) wave, traveling at the maximum speed of 2 units per time step in the direction of the General. These “backwards” \( A_1 \) waves serve to inform the slow \( B_i \) waves of when to move forward, resulting in the progressively slower \( B_i \) waves.

The slow waves progress as follows. In time step 1, machine 1 will enter the \( Q_2 \) Quiescent
state. The slow waves consist of states $B_1$, $B_2$, and the occasional $Q_2$ state. In time step 2, machine 1 will enter the $B_1$ state by the rule $(\ast, G, Q_2, A_1, Q_1) \rightarrow B_1$. Then, the slow wave progresses using these general properties: 1) If a cell is in state $B_1$ with its near-right or far-right neighbor in state $A_1$, the signal will progress one spot to the right as $B_1$ or $B_2$, provided that there is no $A_2$ or $G$ as a near or far-right neighbor of the $A_1$ cell, as laid out in the transition tables. 2) If a cell is in state $B_1$ with no $A_1$ neighbor near or far-right, the signal will remain in place as $B_1$, provided that no near neighbor (either side) is in $B_1$, no left neighbor is in $A_2$, there is not at least one $G$ neighbor on each side, it is not about to intersect an $A_2$ far-right neighbor, and its left neighbors are not $A_1Q_1$ in that order. 3) If a machine is in state $B_2$ with near or far-right neighbor in state $A_1$, the signal will stay in place but change to $B_1$. 4) If a cell is in state $B_2$ with no $A_1$ or $A_2$ neighbor near or far-right, the signal will remain in place as $B_2$, provided that there is not at least one $G$ neighbor on each side, there is no $A_2$ neighbor near or far-left, and there is no $B_1$ or $B_2$ neighbor far-right while a left neighbor is $A_1$.

An $A_2$ wave is sent from the new General created in cell $n - 1$ after the primary $A_1$ wave reaches the end of the array. When this $A_2$ wave meets the slow $B_i$ wave and the length of the squad is odd, the intersection will create one new General. If the length of the squad is even the waves will intersect and form two Generals, side by side. This new General or Generals are created at the middle cell or cells of the firing squad array and will emit their own signals to further divide the array. The process is iteratively repeated until the squad is sufficiently divided into small enough sub-arrays to enter the Fire state simultaneously at time $t = n - 1$. 

23
3.3 Transition Tables for the Solution

A state transition rule of a cell in state $X$ will be denoted by $(V, W, X, Y, Z) \rightarrow U$ where $V, W, Y, Z \in S \cup \{\ast\}$, with $\ast$ representing a dummy state (no actual cell) and $X, U \in S$. This state transition rule represents that at time $t$, if a cell has state $X$ and its far left neighbor has state $V$, near left neighbor has state $W$, near right neighbor has state $Y$, and far right neighbor has state $Z$, (where by default if any of the neighbors do not exist in the array, the state is taken to be the dummy state $\ast$), then at time $t + 1$ the cell will enter state $U$. In the tables to follow, the state transition rule $(V, W, X, Y, Z) \rightarrow U$ will be represented in the table with state $X$ in the upper left corner. To locate the particular rule, locate $VW$ in the left column and $YZ$ on the top row. Move horizontally across the row starting with $VW$ until you reach the column starting with $YZ$. In this location in the array will be the state $U$ - the output of the rule.
Figure 3.4: Boddie Solution - \( n = 25 \)

Table 3.1: Transition Tables for a Minimal Time Solution to the FSSP with Von Neumann Neighborhood of Extent 2

<table>
<thead>
<tr>
<th>( G )</th>
<th>( Q_1Q_2 )</th>
<th>( Q_4Q_3 )</th>
<th>( B_3A_1 )</th>
<th>( B_1A_1 )</th>
<th>( B_3B_2 )</th>
<th>( B_2B_2 )</th>
<th>**</th>
<th>( Q_1Q_3 )</th>
<th>( Q_1Q_2 )</th>
<th>( B_1A_1 )</th>
<th>( B_1B_1 )</th>
<th>( G_1B_1 )</th>
<th>( Q_1Q_2 )</th>
<th>( Q_1Q_2 )</th>
<th>( Q_1Q_2 )</th>
<th>( G_1Q_1 )</th>
<th>( Q_1B_2 )</th>
<th>( GQ_2 )</th>
<th>( GQ_1 )</th>
<th>( Q_2G )</th>
<th>( GQ_1 )</th>
<th>( Q_1G )</th>
<th>( B_2Q_2 )</th>
<th>( B_2Q_1 )</th>
<th>( Q_1G )</th>
<th>( A_2G )</th>
<th>( Q_1Q_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( F )</td>
<td>( F )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
<td>( G )</td>
</tr>
<tr>
<td>$Q_1$</td>
<td>$Q_1Q_1$</td>
<td>$Q_1*$</td>
<td>$A_2B_1$</td>
<td>$A_2Q_1$</td>
<td>$B_1Q_1$</td>
<td>$G*$</td>
<td>$A_2B_2$</td>
<td>$B_1G$</td>
<td>$GQ_2$</td>
<td>$B_2Q_1$</td>
<td>$GG$</td>
<td>$Q_1B_1$</td>
<td>$Q_1A_2$</td>
<td>$Q_1B_2$</td>
<td>$B_2A_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>--------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>-----</td>
<td>----------</td>
<td>--------</td>
<td>-------</td>
<td>--------</td>
<td>-----</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>---------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$F$</td>
<td>$F$</td>
<td>$F$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$GQ_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$G$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_1Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2A_2$</td>
<td>$B_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$B_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2A_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$B_1$</td>
<td>$Q_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2A_2$</td>
<td>$B_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$B_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_1Q_2$</td>
<td>$Q_1A_2$</td>
<td>$B_1G$</td>
<td>$Q_1B_1$</td>
<td>$GA_1$</td>
<td>$B_2A_2$</td>
<td>$Q_1G$</td>
<td>$B_1Q_1$</td>
<td>$Q_1B_2$</td>
<td>$B_2Q_1$</td>
<td>$GG$</td>
<td>$Q_1Q_1$</td>
<td>$G^*$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>--------</td>
<td>----------</td>
<td>--------</td>
<td>----------</td>
<td>--------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>------</td>
<td>----------</td>
<td>------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_2Q_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_1B_2$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$G$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A_2B_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$GQ_1$</td>
<td>$A_1$</td>
<td>$G$</td>
<td>$A_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_1Q_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_1Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$B_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_1Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_1B_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2B_2$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2Q_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2Q_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td>$A_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B_2Q_1$</td>
<td>$B_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$B_2$</th>
<th>$A_1Q_2$</th>
<th>$Q_2A_1$</th>
<th>$Q_2Q_2$</th>
<th>$A_2Q_1$</th>
<th>$Q_2B_1$</th>
<th>$Q_2G$</th>
<th>$Q_1B_1$</th>
<th>$Q_2B_2$</th>
<th>$Q_2A_2$</th>
<th>$Q_1Q_1$</th>
<th>$Q_2B_1$</th>
<th>$Q_1B_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$Q_2$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
</tr>
<tr>
<td>$Q_2A_1$</td>
<td>$A_2$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$Q_2B_2$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$Q_2Q_2$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$A_2$</td>
<td>$Q_1$</td>
</tr>
<tr>
<td>$A_1B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$A_1Q_2$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$A_1Q_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$A_2B_1$</td>
<td>$B_2$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$B_2$</td>
<td>$B_2$</td>
<td>$B_2$</td>
<td>$B_2$</td>
</tr>
<tr>
<td>$A_2Q_2$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$A_2B_2$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
<td>$Q_1$</td>
</tr>
<tr>
<td>$A_2Q_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$A_1B_2$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$A_1Q_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$B_1Q_2$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
<tr>
<td>$B_2Q_2$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_2$</td>
<td>$G$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
<td>$B_1$</td>
</tr>
</tbody>
</table>

27
4 Proof of Minimal Time Solution

4.1 Image Solutions

In [2], Balzer defined a special type of solution, called an Image Solution, to the standard Firing Squad Synchronization Problem and proved properties that Image Solutions exhibit. We extend the definition of an Image Solution and prove the corresponding properties for the Firing Squad Synchronization Problem with Von Neumann Neighborhoods of Extent 2.

**Definition 2.** Let $S$ be the set of states in a solution. A solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2 is called an Image Solution if for every state transition rule $(V, W, X, Y, Z) \rightarrow U$ in the solution there exists a state transition rule $(\text{Im}(Z), \text{Im}(Y), \text{Im}(X), \text{Im}(W), \text{Im}(V)) \rightarrow \text{Im}(U)$, where $\text{Im}: S \rightarrow S$ and $\text{Im}(\text{Im}(A)) = A$ for every state $A \in S$. $\text{Im}(A)$ is called the image of $A$.

**Definition 3.** A curve is defined to be a connected set of machines in the two dimensional representation of the squad (successive time steps printed on rows beneath the original squad). Each connected set of machines must be a portion of a horizontal row, a vertical column, or two adjacent diagonals at $\pm \arctan(0.5)$ degrees (i.e. diagonals with slope $\pm \frac{1}{2}$).

**Definition 4.** Two curves are said to be equal if they have the same shape and orientation and if elements along one curve are identical to the corresponding elements of the other curve.
Definition 5. Two curves are said to be images of each other if they have the same shape and orientation, except the horizontal direction is reversed, and elements along one curve are images of the corresponding elements of the other curve.

Definition 6. A closed curve consists of a vertical curve adjoined at the top to a curve consisting of two adjacent diagonals at $\pm \arctan(0.5)$ degrees and closed off by either a diagonal at $\pm \arctan(0.5)$ degrees extending from the end of the adjacent diagonals to the bottom end of the vertical curve or by a horizontal curve connecting the bottom end of the adjacent diagonals to the bottom end of the vertical curve.

Definition 7. The closure of two curves, a vertical curve adjoined at the top to a curve consisting of two adjacent diagonals, is the closed curve consisting of these curves and the horizontal curve or diagonal curve needed to form a closed curve.
Definition 8. The parents of an element are the elements which determine its value. For example, if $S(x,t)$ refers to the state of the machine $x$ units from the left end at time $t$, then the parents of the element $S(x,t)$ are $S(x-2,t-1), S(x-1,t-1), S(x,t-1), S(x+1,t-1), S(x+2,t-1)$.

In [2], Balzer proved a property of image solutions stating that if a pair of curves in separate simulations are equal (or images) of each other then all cells in the interior of the closure of these curves are also equal (or images) of each other. We prove that this same property holds with Von Neumann neighborhoods of extent 2.

Theorem 2. Consider two pairs of curves, as defined in Definition 3, each consisting of a vertical curve adjoined at the top to a curve consisting of two adjacent diagonals. If the corresponding curves in the pairs are equal (or images) then all elements in the interior or on the bottom border of the closures are equal (or images).

Proof. Assume there is at least one element in the interior or bottom border of the closure of one of the pairs that is not equal (or the image) of the corresponding element in the other pair’s closure. Then, since the transitions in the transition tables are deterministic, at least one of that element’s parents is not equal (or the image) to its corresponding element. Similarly, one of this element’s parents must also not be equal (or the image) to its corresponding element. Continuing this process until we reach the border of the closure, we find an element on one of our original curves is not equal (or the image) to the corresponding element of the original curve in the other pair. This contradicts the assumptions of the theorem, and hence all elements in the interior or on the border of the closures are equal. 

□
4.2 Proof of 651 rule 8 state Minimal Time Solution

Let $R_0$ be the set consisting of the 651 rules outlined in the transition tables found in Table 3.1. Let $R_1$ be the set consisting of the union of $R_0$ and the following additional rules in Table 4.1:

Table 4.1: Additional Rules for $R_1$

<table>
<thead>
<tr>
<th>$Q_2$</th>
<th>$B_1A_1$</th>
<th>$B_2A_1$</th>
<th>$G*$</th>
<th>$Q_2G$</th>
<th>$Q_2Q_2$</th>
<th>$A_2Q_1$</th>
<th>$Q_2A_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>$B_1$</td>
<td>$Q_2$</td>
<td>$F$</td>
<td>$G$</td>
<td>$Q_2$</td>
<td>$G$</td>
<td></td>
</tr>
<tr>
<td>$*G$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$*Q_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Q_2Q_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Let $S^* = \{G, Q_1, Q_2, A_1, A_2, B_1, B_2, F, *\}$ be the set of states used in $R_0$ and $R_1$, including the nonexistent state *. Let $\text{Im} : S^* \rightarrow S^*$ be defined by

$$\text{Im}(s) = \begin{cases} 
    s & s = G, F, B_1, B_2, * \\
    Q_1 & s = Q_2 \\
    Q_2 & s = Q_1 \\
    A_1 & s = A_2 \\
    A_2 & s = A_1 
\end{cases}$$

Using $\text{Im}(s)$, we can see that for every state transition rule $(V, W, X, Y, Z) \rightarrow U \in R_1$ there exists a state transition rule $(\text{Im}(Z), \text{Im}(Y), \text{Im}(X), \text{Im}(W), \text{Im}(V)) \rightarrow \text{Im}(U) \in R_1$.
and that $Im(Im(s)) = s$ for every $s \in S^*$. 

We will show that the set of states $S^*$ with rule set $R_1$ forms a minimal time solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2 by using properties of image solutions. Then, we will show that the rules in rule set $R_0$ are necessary and sufficient, meaning that $S^*$ with $R_0$ is a minimal time solution to the problem.

In Noguchi’s minimal time solution to the standard FSSP, he noted that his “slow” waves formed a virtual binary counter which counted the number of cells that backwards waves had traveled [7]. He then continued to use binary in his proof due to this virtual counter and due to the binary nature of repeated halving of the array [7]. Our solution to the FSSP with Von Neumann Neighborhood of Extent 2 does not exhibit this same behavior of forming a binary counter, however we will also prove the correctness of the solution using binary, as inspired by Noguchi. Using binary is a convenient way to express dividing in half repeatedly, as dividing by 2 in binary is represented by simply removing the bit farthest to the right (rounded down as needed). We utilize this in the statements and proofs needed to prove the correctness of the solution to express the half point, quarter point, eighth point and so on in a concise way.

**Theorem 3.** The set of states $S^*$, as on page 31, with rule set $R_1$, outlined in Tables 3.1 and 4.1 is a minimal time solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2. That is, for any size $n$, the one-dimensional array of size $n$ will synchronize with $S(i, n-1) = F$ for $0 \leq i \leq n-1$ and $S(i, t) \neq F$ for $0 \leq i \leq n-1$ and $t < n - 1$.

The proof of this theorem follows multiple lemmas.
Lemma 1. (Propagation of the initial $A_1$ wave)

With set of states $S^*$ as defined on page 31, rule set $R_1$, outlined in Tables 3.1 and 4.1, and $n \geq 6$

a. $S(2i, i) = A_1$ with $1 \leq i \leq \frac{n-3}{2}$ if $n$ is odd and $1 \leq i \leq \frac{n-2}{2}$ if $n$ is even

b. $S(2i, i+1) = A_1$ with $1 \leq i \leq \frac{n-3}{2}$ if $n$ is odd and $1 \leq i \leq \frac{n-4}{2}$ if $n$ is even

c. $S(2i+1, i+1) = Q_2$ with $1 \leq i \leq \frac{n-3}{2}$ if $n$ is odd and $1 \leq i \leq \frac{n-2}{2}$ if $n$ is even

d. $S(j, i) = Q_1$ for $2i < j \leq n - 1$

e. $S\left(n - 2, \frac{n}{2}\right) = A_2$ for even $n$

Proof. We use induction on $i$ to prove a. – d.

Recall that at time $t = 0$, the array is as follows:

$$** GQ_1Q_1Q_1 \ldots Q_1 **$$

That is, the initial array satisfies $S(0, 0) = G$ and $S(i, 0) = Q_1$ for $1 \leq i \leq n - 1$, as laid out in the problem statement.

Assume $i = 1$. We need to verify:

a. $S(2, 1) = A_1$

b. $S(2, 2) = A_1$

c. $S(3, 2) = Q_2$

d. $S(j, 1) = Q_1$ for $2 < j \leq n - 1$

The machine in position 2 is in state $Q_1$ at time $t = 0$, and applying the rule $(G, Q_1, Q_1, Q_1, Q_1) \rightarrow A_1$ gives $S(2, 1) = A_1$.

To verify b. we must find the states of the neighbors of the cell in position 2 at time $t = 1$ in order to apply rules to find the state of cell 2 at time $t = 2$. Applying the
rules \((\ast, \ast, G, Q_1, Q_1) \rightarrow G\), \((\ast, G, Q_1, Q_1, Q_1) \rightarrow Q_2\), \((Q_1, Q_1, Q_1, Q_1, Q_1) \rightarrow Q_1\), \((Q_1, Q_1, Q_1, Q_1, \ast) \rightarrow Q_1\) and \((Q_1, Q_1, Q_1, \ast, \ast) \rightarrow Q_1\) places the array at time \(t = 1\) in the following states:

\[
**GQ_2A_1Q_1 \ldots Q_1**
\]

Then applying the rule \((G, Q_2, A_1, Q_1, Q_1) \rightarrow A_1\) results in \(S(2, 2) = A_1\).

To verify c., apply the rule \(Q_2, A_1, Q_1, Q_1, Q_1) \rightarrow Q_2\) to find \(S(3, 2) = Q_2\).

To verify d., the rules \((Q_1, Q_1, Q_1, Q_1, Q_1) \rightarrow Q_1\), \((Q_1, Q_1, Q_1, Q_1, \ast) \rightarrow Q_1\) and \((Q_1, Q_1, Q_1, \ast, \ast) \rightarrow Q_1\) result in \(S(j, 1) = Q_1\) for all \(2 < j \leq n - 1\) - meaning there is no spontaneous state changes.

Now assume statements a. – d. hold for \(i = k\). That is, assume the following are true:

a. \(S(2k, k) = A_1\) where \(k \leq \frac{n - 5}{2}\) if \(n\) is odd and \(k \leq \frac{n - 4}{2}\) if \(n\) is even

b. \(S(2k, k + 1) = A_1\) where \(k \leq \frac{n - 5}{2}\) if \(n\) is odd and \(k \leq \frac{n - 6}{2}\) if \(n\) is even

c. \(S(2k + 1, k + 1) = Q_2\) where \(k \leq \frac{n - 5}{2}\) if \(n\) is odd and \(k \leq \frac{n - 4}{2}\) if \(n\) is even

d. \(S(j, k) = Q_1\) for \(2k < j \leq n - 1\)

We will verify these four statements for \(i = k + 1\). That is, we need to show:

a. \(S(2(k + 1), k + 1) = S(2(k + 2), k + 1) = A_1\) where \(k + 1 \leq \frac{n - 5}{2} + 1 = \frac{n - 3}{2}\) if \(n\) is odd and \(k + 1 \leq \frac{n - 4}{2} + 1 = \frac{n - 2}{2}\) if \(n\) is even

b. \(S(2(k + 1), (k + 1) + 1) = S(2(k + 2), k + 2) = A_1\) where \(k + 1 \leq \frac{n - 5}{2} + 1 = \frac{n - 3}{2}\) if \(n\) is odd and \(k + 1 \leq \frac{n - 6}{2} + 1 = \frac{n - 4}{2}\) if \(n\) is even

c. \(S(2(k + 1) + 1, (k + 1) + 1) = S(2(k + 3), k + 2) = Q_2\) where \(k + 1 \leq \frac{n - 5}{2} + 1 = \frac{n - 3}{2}\)
if \( n \) is odd and \( k + 1 \leq \frac{n - 4}{2} + 1 = \frac{n - 2}{2} \) if \( n \) is even

d. \( S(j, k + 1) = Q_1 \) for \( 2(k + 1) = 2k + 2 < j \leq n - 1 \)

By the assumptions made above, the squad at time \( t = k \), where \( I \) means a cell in an irrelevant state, is one of the following:

\[
\ast\ast I \ldots A_1 Q_1 Q_1 Q_1 Q_1 \ldots \ast\ast \quad \ast\ast I \ldots A_1 Q_1 Q_1 Q_1 \ast\ast
\]

where the first cell indicated after the irrelevant cells is in position \( 2k \). In both cases, we find that \( S(2k + 2, k + 1) = A_1 \) by either \((A_1, Q_1, Q_1, Q_1, Q_1) \rightarrow A_1\) or \((A_1, Q_1, Q_1, Q_1, *) \rightarrow A_1\).

To show \( b. \) and \( c. \) we must observe what happens to the array at time \( t = k + 2 \). At time \( t = k + 1 \), with \( k + 1 \) satisfying the restrictions given, the squad is in one of the following:

\[
\ast\ast I \ldots A_1 Q_2 A_1 Q_1 Q_1 Q_1 \ldots \ast\ast \quad \ast\ast I \ldots A_1 Q_2 A_1 Q_1 Q_1 \ast\ast
\]

Note that \( S(2k; k + 1) = A_1 \) and \( S(2k + 1, k + 1) = Q_2 \) were given by assumption and that we already showed \( S(2k + 2, k + 1) = A_1 \). Then at time \( t = k + 2 \), we have from the rules \((A_1, Q_2, A_1, Q_1, Q_1) \rightarrow A_1\), \((Q_2, A_1, Q_1, Q_1, Q_1) \rightarrow Q_2\), and \((Q_2, A_1, Q_1, Q_1, *) \rightarrow Q_2\) that \( S(2k + 2, k + 2) = A_1 \) and \( S(2k + 3, k + 2) = Q_2 \).

Part \( d. \) can be seen by observing the squad at time \( t = k \), which again is as follows:

\[
\ast\ast I \ldots A_1 Q_1 Q_1 Q_1 Q_1 \ldots \ast\ast \quad \ast\ast I \ldots A_1 Q_1 Q_1 Q_1 \ast\ast
\]

where the first cell not in an irrelevant state is in position \( 2k \). We want \( S(j, k + 1) = Q_1 \) where \( 2k + 2 < j \leq n - 1 \). The cells in positions \( 2k + 3 \) and greater, by following rules
\((Q_1, Q_1, Q_1, Q_1, Q_1) \rightarrow Q_1\), \((Q_1, Q_1, Q_1, Q_1, *) \rightarrow Q_1\), and \((Q_1, Q_1, Q_1, *, *) \rightarrow Q_1\) all enter state \(Q_1\) at time \(k + 1\), so \(S(j, k + 1) = Q_1\) where \(2k + 2 < j \leq n - 1\) - no spontaneous state changes have occurred.

By the principles of mathematical induction, parts \(a. - d.\) hold for all \(i\) as stated.

It remains to show part \(e.\). By parts \(a. - d.\) we know that the squad at time \(t = \frac{n - 2}{2}\) for even \(n\) is as follows:

\[** I \ldots A_1Q_2A_1Q_1**\]

By the rule \((A_1, Q_2, A_1, Q_1, *) \rightarrow A_2\), we have that \(S\left(n, \frac{n}{2}\right) = A_2\). This is the cell in position \(n - 2\) “seeing” that the \(A_1\) signal will reach the end in the next time step and thus beginning to send the return signal \(A_2\) in that next time step.

\[\square\]

**Lemma 2.** (Progress of slow waves and propagation of backwards \(A_1\) waves)

Let \((x_mx_{m-1} \ldots x_1x_0)_2\) be the binary representation of \(i + 1\), where \(m \geq 1\) and \(x_m = 1\). That is, \(i + 1 = \sum_{k=0}^{m} x_k2^k\) where \(x_k \in \{0, 1\}\), \(m \geq 1\), and \(x_m = 1\). Consider the space-time diagram representing the execution of the FSSP with time \(t = 0\) as the first row and time increasing as we read vertically down. If \(i \in 2\mathbb{Z}\), let \(C_1\) be the arctan \(\left(\frac{1}{2}\right)\) degree diagonal curve beginning at \(\left(i, \frac{i}{2}\right)\) and ending at \((0, i)\) and let \(C_2\) be the adjacent arctan \(\left(\frac{1}{2}\right)\) degree diagonal curve beginning at \(\left(i - 1, \frac{i}{2} + 1\right)\) and ending at \((1, i)\). If \(i \in \mathbb{Z} \setminus 2\mathbb{Z}\), let \(C_1\) be the arctan \(\left(\frac{1}{2}\right)\) degree diagonal curve beginning at \(\left(i - 1, \frac{i + 1}{2}\right)\) and ending at \((0, i)\) and let \(C_2\) be the adjacent arctan \(\left(\frac{1}{2}\right)\) degree diagonal curve beginning at \(\left(i, \frac{i + 1}{2}\right)\) and ending at \((1, i)\). The notation \(C(j)\) refers to the state of the machine in position \(j\) that lies on the curve \(C\).
Using the set of states $S^*$ (page 31) and rule set $R_1$ (Tables 3.1 and 4.1), for $n \geq 4$ and for all $i$ such that $2 \leq i \leq n-2$ the following are true:

a. For all $j$ such that $1 + \sum_{k=1}^{m} x_k 2^{k-1} \leq j < i$ we have $C_1(j) = A_1$ if $j \in 2\mathbb{Z}$ and $C_2(j) = Q_2$ if $j \in \mathbb{Z} \setminus 2\mathbb{Z}$.

b. For all $\ell$ such that $0 \leq \ell \leq m - 1$
   
i. If $x_\ell = 1$ then $C_1 \left( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \right) = B_1$ if $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \in 2\mathbb{Z}$ and $C_2 \left( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \right) = B_1$ if $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \in \mathbb{Z} \setminus 2\mathbb{Z}$.
   
   ii. If $x_\ell = 1$ and $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \geq 3$, then for all $j$ such that $1 + \sum_{k=\ell+2}^{m} x_k 2^{k-(\ell+2)} \leq j \leq -1 + \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)}$ we have $C_1(j) = Q_2$ if $j \in 2\mathbb{Z}$ and $C_2(j) = Q_2$ if $j \in \mathbb{Z} \setminus 2\mathbb{Z}$.
   
   iii. If $x_\ell = 0$ and there exists $p$ such that $0 \leq p < \ell$, $x_p = 1$, and $\ell < m - 1$, then $C_1 \left( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \right) = B_2$ if $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \in 2\mathbb{Z}$ and $C_2 \left( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \right) = B_2$ if $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \in \mathbb{Z} \setminus 2\mathbb{Z}$.
   
   iv. If $x_\ell = 0$ and there exists $p$ such that $0 \leq p < \ell$, $x_p = 1$, $\ell < m - 1$ and $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \geq 3$ then for all $j$ such that $1 + \sum_{k=\ell+2}^{m} x_k 2^{k-(\ell+2)} \leq j \leq -1 + \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)}$ we have that $C_1(j) = Q_2$ if $j \in 2\mathbb{Z}$ and $C_2(j) = Q_2$ if $j \in \mathbb{Z} \setminus 2\mathbb{Z}$.
   
   v. If $x_\ell = 0$, $x_p = 0$ for all $p$ such that $0 \leq p \leq \ell$, and $\ell < m - 1$ then $C_1 \left( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \right) = A_1$ if $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \in 2\mathbb{Z}$ and $C_2 \left( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \right) = B_2$ if $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \in \mathbb{Z} \setminus 2\mathbb{Z}$.
   
   vi. If $x_\ell = 0$, $x_p = 0$ for all $p$ such that $0 \leq p \leq \ell$, $\ell < m - 1$, and $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} \geq 3$, then for all $j$ such that $1 + \sum_{k=\ell+2}^{m} x_k 2^{k-(\ell+2)} \leq j \leq -1 + \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)}$ we have that $C_1(j) = A_1$ if $j \in 2\mathbb{Z}$ and $C_2(j) = Q_2$ if $j \in \mathbb{Z} \setminus 2\mathbb{Z}$.
   
   vii. If $x_{m-1} = 0$ then $C_2(1) = Q_2$.  

37
Proof. We will prove using induction. For a base case, all statements were verified by direct verification for squads of any length $N \leq N_0 = 85$. As an illustration, the direct verification for a squad of length 16 and $i = 11$ as well as a squad of length 16 and $i = 12$ are shown below.

As seen in Figure 4.2 for $n = 16$ and $i = 11$, and thus $i + 1 = 12 = (1100)_2$ and $m = 3$, the statements of the lemma read as follows:

a. For all $j$ such that $7 \leq j \leq 11$, $C_1(j) = A_1$ if $j \in 2\mathbb{Z}$ and $C_2(j) = Q_2$ if $j \in \mathbb{Z} \setminus 2\mathbb{Z}$.

b. For all $\ell$ such that $0 \leq \ell \leq 2$

i. $x_\ell = 1$ implies $\ell = 2$, then since $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} = \sum_{k=3}^{3} x_k 2^{k-3} = 1 \in \mathbb{Z} \setminus 2\mathbb{Z}$, we have $C_2(1) = B_1$.

ii. $x_\ell = 1$ implies $\ell = 2$, and since $\sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} = \sum_{k=3}^{3} x_k 2^{k-3} = 1 \not\geq 3$, this part does not apply.

iii. $x_\ell = 0$ implies $\ell = 0, 1$. For either of these there is no $p$ such that $0 \leq p < \ell$ with $x_p = 1$ so this part does not apply.
iv. \( x_\ell = 0 \) implies \( \ell = 0, 1 \). For either of these there is no \( p \) such that \( 0 \leq p < \ell \) with \( x_p = 1 \) so this part does not apply.

v. \( x_\ell = 0 \) implies \( \ell = 0, 1 \). For \( \ell = 0 \), we do have that \( x_p = 0 \) for all \( p \) such that \( 0 \leq p \leq 0 \).

Then since \( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} = \sum_{k=1}^{3} x_k 2^{k-1} = 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 = 6 \in 2\mathbb{Z} \), we see that \( C_1(6) = A_1 \).

For \( \ell = 1 \), we also have that \( x_p = 0 \) for all \( p \) such that \( 0 \leq p \leq 1 \).

Then since \( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} = \sum_{k=2}^{3} x_k 2^{k-2} = 1 \times 2^0 + 1 \times 2^1 = 3 \in \mathbb{Z} \setminus 2\mathbb{Z} \), we see that \( C_2(3) = B_2 \).

vi. \( x_\ell = 0 \) implies \( \ell = 0, 1 \) and both satisfy that \( x_p = 0 \) for all \( p \) such that \( 0 \leq p \leq \ell \). If \( \ell = 0 \), then \( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} = \sum_{k=1}^{3} x_k 2^{k-1} = 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 = 6 \geq 3 \), we have to check for all \( j \) such that \( 4 \leq j \leq 5 \) that \( C_1(j) = A_1 \) if \( j \in 2\mathbb{Z} \) and \( C_2(j) = Q_2 \) if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \). We do in fact have \( C_1(4) = A_1 \) and \( C_2(5) = Q_2 \). If \( \ell = 1 \), then \( \sum_{k=\ell+1}^{m} x_k 2^{k-(\ell+1)} = \sum_{k=2}^{3} x_k 2^{k-2} = 1 \times 2^0 + 1 \times 2^1 = 3 \geq 3 \) and we must check the conclusion for all \( j \) such that \( 2 \leq j \leq 2 \) and we do in fact have \( C_1(2) = A_1 \).

vii. \( x_{m-1} = x_2 = 1 \neq 0 \), so this does not apply.

c. \( C_1(0) = G \) can be seen in the simulation.

![Image](image.png)

Figure 4.3: Length 16 Squad Simulation, \( i = 12 \)

\( C_1 \) depicted in blue and \( C_2 \) depicted in gold

As seen in Figure 4.3 for \( n = 16 \) and \( i = 12 \), and thus \( i + 1 = 13 = (1101)_2 \) and \( m = 3 \), the...
statements of the lemma read as follows:

a. For all \( j \) such that \( 7 \leq j \leq 12 \), \( C_1(j) = A_1 \) if \( j \in 2\mathbb{Z} \) and \( C_2(j) = Q_2 \) if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \).

b. For all \( \ell \) such that \( 0 \leq \ell \leq 2 \)

i. \( x_\ell = 1 \) implies \( \ell = 0, 2 \). If \( \ell = 0 \) then since \( \sum_{k=\ell+1} x_k 2^{k-(\ell+1)} = \sum_{k=1}^3 x_k 2^{k-1} = 6 \in 2\mathbb{Z} \), we have \( C_1(6) = B_1 \). If \( \ell = 2 \) then since \( \sum_{k=\ell+1} x_k 2^{k-(\ell+1)} = \sum_{k=3}^3 x_k 2^{k-3} = 1 \in \mathbb{Z} \setminus 2\mathbb{Z} \), we have \( C_2(1) = B_1 \).

ii. \( x_\ell = 1 \) implies \( \ell = 0, 2 \), and for \( \ell = 0 \) we have \( \sum_{k=\ell+1} x_k 2^{k-(\ell+1)} = \sum_{k=1}^3 x_k 2^{k-1} = 6 \geq 3 \).

Then we must have that for all \( j \) such that \( 4 \leq j \leq 5 \), \( C_1(j) = Q_2 \) if \( j \in 2\mathbb{Z} \) and \( C_2(j) = Q_2 \) if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \). We do indeed have \( C_1(4) = Q_2 \) and \( C_2(5) = Q_2 \). If \( \ell = 2 \), then \( \sum_{k=\ell+1} x_k 2^{k-(\ell+1)} = \sum_{k=3}^3 x_k 2^{k-3} = 1 \notin 3 \), and this part does not apply.

iii. \( x_\ell = 0 \) implies \( \ell = 1 \). There does exist \( p \) such that \( 0 \leq p < \ell \) with \( x_p = 1 \) (namely \( p = 0 \)). Then since \( \sum_{k=\ell+1} x_k 2^{k-(\ell+1)} = \sum_{k=2}^3 x_k 2^{k-2} = 3 \in \mathbb{Z} \setminus 2\mathbb{Z} \), we verify that \( C_2(3) = B_2 \) as shown in the simulation.

iv. \( x_\ell = 0 \) implies \( \ell = 1 \). Since \( x_0 = 1 \) and \( \sum_{k=\ell+1} x_k 2^{k-(\ell+1)} = \sum_{k=2}^3 x_k 2^{k-2} = 3 \geq 3 \), we must verify that for all \( j \) such that \( 2 \leq j \leq 2 \) we have \( C_1(j) = Q_2 \) if \( j \in 2\mathbb{Z} \) and \( C_2(j) = Q_2 \) if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \). Indeed, we have \( C_1(2) = Q_2 \).

v. \( x_\ell = 0 \) implies \( \ell = 1 \). But since \( x_0 = 1 \), this part does not apply.

vi. \( x_\ell = 0 \) implies \( \ell = 1 \). But since \( x_0 = 1 \), this part does not apply.

vii. \( x_{m-1} = x_2 = 1 \neq 0 \), so this does not apply.

c. \( C_1(0) = G \) can be seen in the simulation.

Now consider a squad of length \( n \) where \( N_0 - 2 < n - 2 \). For a squad of length \( n \), the lemma holds for all \( i \) such that \( 2 \leq i \leq N_0 - 2 < n - 2 \) using properties of image solutions.

If \( i \) is even, then consider the triangular region within the simulation diagram for a squad of size \( N_0 \) enclosed by the vertical curve at \( x = -1 \), the pair of adjacent \( -arctan \left( \frac{1}{2} \right) \) degree
diagonal curves beginning at \((0,0)\) and \((1,1)\) and ending at \((i, \frac{i}{2})\) and \((i - 1, \frac{i}{2})\) respectively, and the pair of adjacent \(arctan \left( \frac{1}{2} \right)\) degree diagonal curves \(C_1\) and \(C_2\) as described in the lemma. Looking at the triangular area enclosed by the same curves in the simulation diagram for a squad of size \(n\), we see that the vertical curves at \(x = -1\) in both diagrams are equal (these curves are on the boundary and thus in state \(\ast\) at each time step). Also, by Lemma 1, the adjacent diagonals beginning at \((0,0)\) and \((1,1)\) and ending at \((i, \frac{i}{2})\) and \((i - 1, \frac{i}{2})\) respectively are equal in both diagrams. Then by Theorem 2 and using the fact that the set of states \(S^\ast\) with rule set \(R_1\) is an image solution under the function \(Im(s)\) defined on page 31, we see that \(C_1\) and \(C_2\) are the same in each simulation diagram, and hence they satisfy the properties laid out in the lemma for all even \(i\) such that \(2 \leq i \leq N_0 - 2 < n - 2\) in the simulation diagram for a squad of length \(n\).

If \(i\) is odd, then consider the triangular region within the simulation diagram for a squad of size \(N_0\) enclosed by the vertical curve at \(x = -1\), the pair of adjacent \(-arctan \left( \frac{1}{2} \right)\) degree diagonal curves beginning at \((0,0)\) and \((1,1)\) and ending at \((i + 1, \frac{i + 1}{2})\) and \((i, \frac{i + 1}{2})\) respectively, and the pair of adjacent \(arctan \left( \frac{1}{2} \right)\) degree diagonal curves \(C_1\) and \(C_2\) as described in the lemma. Looking at the triangular area enclosed by the same curves in the simulation diagram for a squad of size \(n\), we see that the vertical curves at \(x = -1\) in both diagrams are equal (these curves are on the boundary and thus in state \(\ast\) at each time step). Also, by Lemma 1, the adjacent diagonals beginning at \((0,0)\) and \((1,1)\) and ending at \((i + 1, \frac{i + 1}{2})\) and \((i, \frac{i + 1}{2})\) respectively are equal in both diagrams. Then by Theorem 2 and using the fact that the set of states \(S^\ast\) with rule set \(R_1\) is an image solution under the function \(Im(s)\) (page 31), we see that \(C_1\) and \(C_2\) are the same in each simulation diagram and thus satisfy the conclusions of this lemma for all odd \(i\) such that \(2 \leq i \leq N_0 - 2 < n - 2\), even in the size \(n\) diagram. So the conclusions of the lemma are satisfied for all \(i\) such that \(2 \leq i \leq N_0 - 2 < n - 2\).
Now, assume that the conclusions hold for all \( i \) such that \( 2 \leq i \leq N \) where \( N_0 - 2 \leq N \leq n - 3 \). We will show that the conclusions then also hold for \( N + 1 \).

Let \((x_m x_{m-1} \ldots x_1 x_0)_2\) be the binary representation of \( N + 1 \) and let \((y_M y_{M-1} \ldots y_1 y_0)_2\) be the binary representation of \( N + 2 \). That is, \( N + 1 = \sum_{k=0}^{m} x_k 2^k \) and \( N + 2 = \sum_{k=0}^{M} y_k 2^k \) and \( M = m \) or \( M = m + 1 \). If \( N + 1 \in 2\mathbb{Z} \), let \( D_1 \) be the \( \arctan\left(\frac{1}{2}\right) \) degree diagonal curve beginning at \( \left( N + 1, \frac{N + 1}{2} \right) \) and ending at \((0,N + 1)\) and let \( D_2 \) be the adjacent \( \arctan\left(\frac{1}{2}\right) \) degree diagonal curve beginning at \( \left( N, \frac{N + 1}{2} + 1 \right) \) and ending at \( (1,N + 1) \). If \( N + 1 \in \mathbb{Z} \setminus 2\mathbb{Z} \) then let \( D_1 \) begin at \( \left( N, \frac{N + 2}{2} \right) \) and end at \( (0,N + 1) \) and let \( D_2 \) begin at \( \left( N + 1, \frac{N + 2}{2} \right) \) and end at \( (1,N + 1) \).

First consider if \( N + 1 \) is even. We know by Lemma 1 that \( S\left( N + 1, \frac{N + 1}{2} \right) = A_1 \), \( S\left( N + 1, \frac{N + 1}{2} + 1 \right) = A_1 \), \( S\left( N + 2, \frac{N + 1}{2} + 1 \right) = Q_2 \), and \( S\left( k, \frac{N + 1}{2} \right) = Q_1 \) for all \( k \) such that \( N + 1 < k \leq n - 1 \). All of these are depicted in green in Figure 4.4.

By our inductive hypothesis, we know that for all \( j \) such that \( 1 + \sum_{k=1}^{m} x_k 2^{k-1} \leq j \leq N \), we have \( C_1(j) = A_1 \) if \( j \in 2\mathbb{Z} \) and \( C_2(j) = Q_2 \) if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \), as shown in gold in Figure 4.4. We also know the corresponding information for \( i = N - 1 \) as shown in blue in Figure 4.4. Then by applying the rules \((A_1, Q_2, A_1, Q_2, A_1) \rightarrow A_1 \), \((Q_2, A_1, Q_2, A_1, Q_1) \rightarrow Q_2 \), and \((Q_2, A_1, Q_2, A_1, Q_2) \rightarrow Q_2 \) and noting that since \( N + 1 \) is even, we know \( x_0 = 0 \) and thus \( \sum_{k=1}^{m} x_k 2^{k-1} = \sum_{k=1}^{M} y_k 2^{k-1} \), we have that for all \( j \) such that \( 1 + \sum_{k=1}^{M} y_k 2^{k-1} \leq j \leq N + 1 \) we have \( D_1(j) = A_1 \) if \( j \in 2\mathbb{Z} \) and \( D_2(j) = Q_2 \) if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \).

Now consider if \( N + 1 \) is odd. We know by Lemma 1 that \( S\left( N, \frac{N}{2} \right) = A_1 \), \( S\left( N, \frac{N + 2}{2} \right) =
Figure 4.4: Induction for backwards wave - $N + 1$ even

Green boxes have information from Lemma 1, blue and gold boxes have information from the assumptions for the induction step for $i = N - 1$ and $i = N$ respectively.

$A_1, S \left(N + 1, \frac{N + 2}{2}\right) = Q_2$, and $S \left(k, \frac{N}{2}\right) = Q_1$ for all $k$ such that $N < k \leq n - 1$. All of these are depicted in green in Figure 4.5.

By our inductive hypothesis, we know that for all $j$ such that $1 + \sum_{k=1}^{m} x_k 2^{k-1} \leq j \leq N$, we have $C_1(j) = A_1$ if $j \in 2\mathbb{Z}$ and $C_2(j) = Q_2$ if $j \in \mathbb{Z} \setminus 2\mathbb{Z}$, as shown in gold in Figure 4.5. We also know the corresponding information for $i = N - 1$ as shown in blue in Figure 4.5. Then by applying the rules $(A_1, Q_2, A_1, Q_2, A_1) \rightarrow A_1$ and $(Q_2, A_1, Q_2, A_1, Q_2) \rightarrow Q_2$ and noting that since $N + 1$ is odd, we know $x_0 = 1$ and thus $1 + \sum_{k=1}^{m} x_k 2^{k-1} = \sum_{k=1}^{M} y_k 2^{k-1}$, we have that for all $j$ such that $1 + \sum_{k=1}^{M} y_k 2^{k-1} \leq j \leq N + 1$ we have $D_1(j) = A_1$ if $j \in 2\mathbb{Z}$ and $D_2(j) = Q_2$ if $j \in \mathbb{Z} \setminus 2\mathbb{Z}$. Thus the conclusions of Lemma 2 part a are proven.

Now we examine the movement of the “slow” $B$ waves, starting with the first “slow” $B$ wave.
Figure 4.5: Induction for backwards wave - $N + 1$ odd

Green boxes have information from Lemma 1, blue and gold boxes have information from the assumptions for the induction step for $i = N - 1$ and $i = N$ respectively through the $(p + 1)$st “slow wave.” Since smaller length $n$ squads can be verified by hand simply using the transition tables, we assume that

$$\sum_{k=p+1}^{m} x_k 2^{k-(p+1)} - \sum_{k=p+2}^{m} x_k 2^{k-(p+2)} \geq 3.$$ 

This assumption assures that the $(p + 1)$st and $(p + 2)$nd “slow” $B$ waves are separated by at least 2 cells along our pair of $arctan \left( \frac{1}{2} \right)$ diagonal curves $C_1$ and $C_2$.

When the “backward” $A_1$ wave traveling along $D_1$ and $D_2$ meets the first “slow” wave coming from $C_1$ and $C_2$, the assumed conditions of this lemma for all $i$ such that $2 \leq i \leq N$ where $N_0 - 2 \leq N \leq n - 3$ result in one of four possible configurations of states, as seen in Figure 4.6. Configurations 1 and 2 arise for $N + 1 \in 2\mathbb{Z}$ and configurations 3 and 4 arise for $N + 1 \in \mathbb{Z} \setminus 2\mathbb{Z}$.

Note that in 1 and 2 in Figure 4.6 the backward $A_1$ wave on $D_1$ does not continue after the first “slow” $B$ wave meets the $D_1, D_2$ pair. However, in 3 and 4 in Figure 4.6 the
Figure 4.6: First “slow” wave meeting the backwards wave along $D_1$ and $D_2$. $D_1$ in blue and $D_2$ in gold. 1 and 2 are for even $N+1$ while 3 and 4 are for odd $N+1$. 

backwards $A_1$ wave on $D_1$ continues to propagate even after the “slow” $B$ wave meets the $D_1, D_2$ pair.

When examining the second “slow” $B$ wave, 8 different transition configurations, depicted in Figure 4.7, are possible, depending on the binary expansion $(x_m x_{m-1} \ldots x_1 x_0)_2$ of $N+1$. These configurations are found due to our induction hypothesis of the conclusions of this lemma holding for all $i$ such that $2 \leq i \leq N$ where $N_0 - 2 \leq N \leq n - 3$.

If in the binary expansion of $N+1 = (x_m x_{m-1} \ldots x_1 x_0)_2$ we have that $x_1 = 0 = x_0$, the possible configurations are 1 or 2 in Figure 4.7. If $x_1 = 1$ and $x_0 = 0$, we have 3 or 4 in Figure 4.7. If $x_1 = 0$ and $x_0 = 1$, we have 5 or 6 in Figure 4.7. If $x_1 = 1 = x_0$, the
Figure 4.7: Second “slow” wave meeting $D_1$ and $D_2$
$D_1$ in blue and $D_2$ in gold. 1 and 2 are for $(x_1x_0) = (00)$, 3 and 4 are for $(x_1x_0) = (10)$, 5 and 6 are for $(x_1x_0) = (01)$, and 7 and 8 are for $(x_1x_0) = (11)$.

possible transition configurations are as in 7 or 8 in Figure 4.7. Note that in this last case, the “backwards” $A_1$ wave along $D_1$ and $D_2$ continues to propagate after meeting the second “slow” wave.

Now consider the third “slow” $B$ wave. 12 different transition configurations, depicted in Figure 4.8 for $q = 2$, are possible, depending on the binary expansion $(x_mx_{m-1} \ldots x_1x_0)_2$ of $N + 1$. These configurations are found due to our induction hypothesis of the conclusions of this lemma holding for all $i$ such that $2 \leq i \leq N$ where $N_0 - 2 \leq N \leq n - 3$. In 9 and 10 in Figure 4.8 the state $I$ refers to an irrelevant state - it can be in $Q_2$ or in $A_1$, but this does not change the states along $D_1$ and $D_2$.

If in the binary expansion of $N + 1 = (x_mx_{m-1} \ldots x_1x_0)_2$ we have $(x_2x_1x_0) = (000)$, then the transition pattern is that of 1 or 2 in Figure 4.8. If $(x_2x_1x_0) = (100)$, the transition pattern is either 3 or 4 in Figure 4.8. If $(x_2x_1x_0) = (011)$, the transition pattern is either 7 or 8 in Figure 4.8. If $(x_2x_1x_0) = (010)$ or $(001)$, the transition pattern is either 9 or 10 in Figure 4.8. Lastly, if
Figure 4.8: $q + 1$st Slow Wave meeting $D_1$ and $D_2$ for $q \geq 2$

$D_1$ depicted in blue, $D_2$ depicted in gold. 1, 2: $x_0 = x_1 = \ldots = x_q = 0$ (and so $y_0 = 1$ and $y_1 = y_2 = \ldots = y_q = 0$). 3, 4: $x_0 = x_1 = \ldots = x_{q-1} = 1$ and $x_q = 0$ (and so $y_0 = y_1 = \ldots = y_q = 1$). 5, 6: $x_0 = x_1 = \ldots = x_q = 1$ (and so $y_0 = y_1 = \ldots = y_q = 0$). 7, 8: $x_0 = x_1 = \ldots = x_q = 1$ (and so $y_0 = y_1 = \ldots = y_q = 0$). 9, 10: $x_q = y_q = 0$ (except for cases already covered by 1, 2). 11, 12: $x_q = y_q = 1$ (except for cases already covered by 3, 4).

$(x_2 x_1 x_0) = (110)$ or $(101)$, the transition pattern is that of 11 or 12 in Figure 4.8. Note that in the case of $(x_2 x_1 x_0) = (111)$, so transition pattern 7 or 8, the backward wave along $D_1$ and $D_2$ still continues to propagate after meeting the “slow” wave.

Now, for $q$ such that $2 \leq q \leq p$, we show that the possible transition configurations for the $(q + 1)$st “slow” $B$ wave are the same as those depicted in Figure 4.8.

If the transition configuration for the $q$th “slow” wave is pattern 1 or 2, then we know...
that the binary expansion of $N + 1 = (x_mx_{m-1} \ldots x_1x_0)_2$ is $(x_mx_{m-1} \ldots x_q00 \ldots 0)_2$ and the binary expansion of $N + 2$ is $(y_{M}y_{M-1} \ldots y_q00 \ldots 01)_2$. Since $x_q$ is either 0 or 1 and $y_q$ can be determined by adding 1 to the binary expansion of $N + 1$, the possibilities for the transition configuration of the $(k + 1)$st “slow” wave are given by $(x_mx_{m-1} \ldots x_q+1000 \ldots 0)_2$ and $(y_{M}y_{M-1} \ldots y_q+1000 \ldots 1)_2$, resulting in configuration 1 or 2 in Figure 4.8 or by $(x_mx_{m-1} \ldots x_q+1100 \ldots 0)_2$ and $(y_{M}y_{M-1} \ldots y_q+1100 \ldots 1)_2$, resulting in configuration 3 or 4.

If the transition configuration for the $q$th “slow” wave is pattern 3 or 4, then we have $(x_mx_{m-1} \ldots x_q100 \ldots 0)_2$ and $(y_{M}y_{M-1} \ldots y_q100 \ldots 01)_2$. Then one possibility for the transition configuration of the $(k + 1)$st “slow” wave is given by $(x_mx_{m-1} \ldots x_q+10100 \ldots 0)_2$ and $(y_{M}y_{M-1} \ldots y_q+10100 \ldots 1)_2$. In this case, since $x_q = y_q = 0$ but not every bit $x_i$ with $0 \leq i < q$ is 0, we are in configuration 9 or 10 in Figure 4.8. The other possibility is $(x_mx_{m-1} \ldots x_q+11100 \ldots 0)_2$ and $(y_{M}y_{M-1} \ldots y_q+11100 \ldots 1)_2$. In this case, since $x_q = y_q = 1$ but not every bit $x_i$ with $0 \leq i < q$ is 0, we are in configuration 11 or 12.

If the transition configuration for the $q$th “slow” wave is pattern 5 or 6, then we have $(x_mx_{m-1} \ldots x_q011 \ldots 1)_2$ and $(y_{M}y_{M-1} \ldots y_q100 \ldots 0)_2$. Then one possibility for the transition configuration of the $(k + 1)$st “slow” wave is given by $(x_mx_{m-1} \ldots x_q+10011 \ldots 1)_2$ and $(y_{M}y_{M-1} \ldots y_q+10100 \ldots 0)_2$. In this case, since $x_q = y_q = 0$ but not every bit $x_i$ with $0 \leq i < q$ is 0, we are in configuration 9 or 10 in Figure 4.8. The other possibility is $(x_mx_{m-1} \ldots x_q+11011 \ldots 1)_2$ and $(y_{M}y_{M-1} \ldots y_q+11100 \ldots 0)_2$. In this case, since $x_q = y_q = 1$ but not every bit $x_i$ with $0 \leq i < q$ is 0, we are in configuration 11 or 12.

If the transition configuration for the $q$th “slow” wave is pattern 7 or 8, then we have $(x_mx_{m-1} \ldots x_q11 \ldots 1)_2$ and $(y_{M}y_{M-1} \ldots y_q00 \ldots 0)_2$. Then one possibility for the transition configuration of the $(k + 1)$st “slow” wave is given by $(x_mx_{m-1} \ldots x_q+1011 \ldots 1)_2$ and $(y_{M}y_{M-1} \ldots y_q+1100 \ldots 0)_2$. In this case, we are in configuration 5 or 6 in Figure 4.8. The
other possibility is \((x_m x_{m-1} \ldots x_{q+1}11 \ldots 1)_2\) and \((y_M y_{M-1} \ldots y_{q+1}00 \ldots 0)_2\). In this case, we are in configuration 7 or 8.

If the transition configuration for the \(q\)th “slow” wave is pattern 9 or 10, then we have \((x_m x_{m-1} \ldots x_q0x_{q-2} \ldots x_0)_2\) where at least one \(x_i\) with \(0 \leq i \leq q - 2\) is not 0 and \((y_M y_{M-1} \ldots y_q0y_{q-2} \ldots y_0)_2\). Then one possibility for the transition configuration of the \((k + 1)\)st “slow” wave is given by \((x_m x_{m-1} \ldots x_{q+1}0x_{q-2} \ldots x_0)_2\) where at least one \(x_i\) with \(0 \leq i \leq q - 2\) is not 0 and \((y_M y_{M-1} \ldots 00y_{q-2} \ldots y_0)_2\). In this case, since \(x_q = y_q = 0\) but not every bit \(x_i\) with \(0 \leq i < q\) is 0, we are in configuration 9 or 10 in Figure 4.8. The other possibility is given by \((x_m x_{m-1} \ldots x_{q+1}10x_{q-2} \ldots x_0)_2\) where at least one \(x_i\) with \(0 \leq i \leq q - 2\) is not 0 and \((y_M y_{M-1} \ldots 10y_{q-2} \ldots y_0)_2\). In this case, since \(x_q = y_q = 1\) but not every bit \(x_i\) with \(0 \leq i < q\) is 0, we are in configuration 11 or 12.

Lastly, if the transition configuration for the \(q\)th “slow” wave is pattern 11 or 12, then we have \((x_m x_{m-1} \ldots x_q1x_{q-2} \ldots x_0)_2\) where at least one \(x_i\) with \(0 \leq i \leq q - 2\) is not 0 and \((y_M y_{M-1} \ldots y_q1y_{q-2} \ldots y_0)_2\). Then one possibility for the transition configuration of the \((k + 1)\)st “slow” wave is given by \((x_m x_{m-1} \ldots x_{q+1}101x_{q-2} \ldots x_0)_2\) where at least one \(x_i\) with \(0 \leq i \leq q - 2\) is not 0 and \((y_M y_{M-1} \ldots 01y_{q-2} \ldots y_0)_2\). In this case, since \(x_q = y_q = 0\) but not every bit \(x_i\) with \(0 \leq i < q\) is 0, we are in configuration 9 or 10 in Figure 4.8. The other possibility is given by \((x_m x_{m-1} \ldots x_{q+1}11x_{q-2} \ldots x_0)_2\) where at least one \(x_i\) with \(0 \leq i \leq q - 2\) is not 0 and \((y_M y_{M-1} \ldots 11y_{q-2} \ldots y_0)_2\). In this case, since \(x_q = y_q = 1\) but not every bit \(x_i\) with \(0 \leq i < q\) is 0, we are in configuration 11 or 12.

In summary, following the numbering in Figure 4.8, if the \(q\)th “slow” wave meets \(D_1\) and \(D_2\) in configuration 1 or 2, the \((q + 1)\)st slow wave will meet \(D_1\) and \(D_2\) in configuration 1, 2, 3 or 4. If in 7 or 8, the next will meet in 5, 6, 7 or 8. If in 3, 4, 5, 6, 9, 10, 11 or 12, the next will meet in 9, 10, 11 or 12.
Hence the conclusions of Lemma 2 part b hold for \( i = N + 1 \) and for all \( q \) such that \( 2 \leq q \leq p \). We must still show that the “slow” waves after the \( p \)th one still satisfy the conclusions. Let \( p = 3 \). Then the assumption of \( \sum_{k=p+1}^{m} x_k 2^{k-(p+1)} - \sum_{k=p+2}^{m} x_k 2^{k-(p+2)} \geq 3 \) means that \( \sum_{k=4}^{m} x_k 2^{k-4} - \sum_{k=5}^{m} x_k 2^{k-5} \geq 3 \) where \( N + 1 = \left(x_m x_{m-1} \ldots x_1 x_0 \right)_2 \). The smallest such \( N + 1 \) which satisfies this assumption is \( N + 1 = 80 = \left(1010000\right)_2 \), thus \( N \geq 79 \). Since we already verified by hand that the lemma is true for all squads up to length 85 and \( 85 - 2 > 79 \), we take \( N_0 = 85 \).

If \( N' \in 2\mathbb{Z} \), let \( C'_1 \) be the \( \text{arctan} \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left(N', \frac{N'}{2}\right) \) and ending at \( (0, N') \) and let \( C'_2 \) be the adjacent \( \text{arctan} \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left(N' - 1, \frac{N'}{2} + 1\right) \) and ending at \( (1, N') \). If \( N' \in \mathbb{Z} \setminus 2\mathbb{Z} \), let \( C'_1 \) be the \( \text{arctan} \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left(N' - 1, \frac{N' + 1}{2}\right) \) and ending at \( (0, N') \) and let \( C'_2 \) be the adjacent \( \text{arctan} \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left(N', \frac{N' + 1}{2}\right) \) and ending at \( (1, N') \).

If \( N' \in 2\mathbb{Z} \), then let \( D'_1 \) begin at \( \left(N', \frac{N' + 2}{2}\right) \) and end at \( (0, N' + 1) \) and let \( D'_2 \) begin at \( \left(N' + 1, \frac{N' + 2}{2}\right) \) and end at \( (1, N' + 1) \). If \( N' \in \mathbb{Z} \setminus 2\mathbb{Z} \) let \( D'_1 \) be the \( \text{arctan} \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left(N' + 1, \frac{N' + 1}{2}\right) \) and ending at \( (0, N' + 1) \) and let \( D'_2 \) be the adjacent \( \text{arctan} \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left(N', \frac{N' + 1}{2} + 1\right) \) and ending at \( (1, N' + 1) \).

There exists \( N' << N \) such that (I) the pair of curves \( C_1 \) and \( C_2 \) are equal to the pair of curves \( C'_1 \) and \( C'_2 \) when starting the comparison of the curves at cell position \( \sum_{k=4}^{m} x_k 2^{k-4} \) and ending at cell position 0 and (II) when comparing the transitions configurations, laid out in Figure 4.8 of the “slow” waves at cell position \( \sum_{k=4}^{m} x_k 2^{k-4} \) passing from \( C_1 \) and \( C_2 \) to \( D_1 \) and \( D_2 \) to the transition configurations of the “slow” waves in the same cell position.
from $C'_1$ and $C'_2$ to $D'_1$ and $D'_2$, the configurations are the same. We show the existence of such an $N'$ below.

First, if the transition configuration of the 4th “slow” wave, which occurs in cell $\sum_{k=4}^{m} x_k 2^{k-4}$, on the pair $C_1$ and $C_2$ is in configuration 1 or 2 of Figure 4.8, then we know that $N + 1 = (x_m x_{m-1} \ldots x_4 0000)_2$. Then, we can pick $N' + 1 = (x_m x_{m-1} \ldots x_4 0000)_2$, which would have the same transition configuration, then, at the same cell location, although it is the 3rd “slow” wave on $C'_1$ and $C'_2$ (note that $N + 1$ is $m + 1$ bits and $N' + 1$ is $m$ bits - so $N' << N$).

So, for transition configuration 1 or 2, $N' = (x_m x_{m-1} \ldots x_4 000)_2 - 1$ satisfies the conditions.

If the transition configuration of the 4th “slow” wave on the pair $C_1$ and $C_2$ is in configuration 3 or 4 of Figure 4.8 then we know that $N + 1 = (x_m x_{m-1} \ldots x_4 1000)_2$. So $N' + 1 = (x_m x_{m-1} \ldots x_4 100)_2$ would have the same transition configuration at the same cell location on $C'_1$ and $C'_2$. Thus for transition pattern 3 or 4, $N' = (x_m x_{m-1} \ldots x_4 100)_2 - 1$ satisfies the conditions.

If the transition configuration of the 4th “slow” wave on the pair $C_1$ and $C_2$ is in configuration 5 or 6 of Figure 4.8 then we know that $N + 1 = (x_m x_{m-1} \ldots x_4 0111)_2$. So $N' + 1 = (x_m x_{m-1} \ldots x_4 011)_2$ would have the same transition configuration at the same cell location on $C'_1$ and $C'_2$. Thus for transition pattern 5 or 6, $N' = (x_m x_{m-1} \ldots x_4 011)_2 - 1$ satisfies the conditions.

If the transition configuration of the 4th “slow” wave on the pair $C_1$ and $C_2$ is in configuration 7 or 8 of Figure 4.8 then we know that $N + 1 = (x_m x_{m-1} \ldots x_4 1111)_2$. So $N' + 1 = (x_m x_{m-1} \ldots x_4 111)_2$ would have the same transition configuration at the same cell location on $C'_1$ and $C'_2$. Thus for transition pattern 7 or 8, $N' = (x_m x_{m-1} \ldots x_4 111)_2 - 1$ satisfies the conditions.
If the transition configuration of the 4th “slow” wave on the pair $C_1$ and $C_2$ is in configuration 9 or 10 of Figure 4.8, then we know that $N + 1 = (x_m x_{m-1} \ldots x_4 0 x_2 x_1 x_0)_2$, where not all of $x_0, x_1, x_2$ are the same. Of $x_0, x_1, x_2$ select $x_a$ and $x_b$ so that $x_a \neq x_b$. Then, $N' + 1 = (x_m x_{m-1} \ldots x_4 0 x_a x_b)_2$ would have the same transition configuration at the same cell location on $C'_1$ and $C'_2$. Thus for transition pattern 9 or 10, $N' = (x_m x_{m-1} \ldots x_4 0 x_a x_b)_2 - 1$ satisfies the conditions.

Lastly, if the transition configuration of the 4th “slow” wave on the pair $C_1$ and $C_2$ is in configuration 11 or 12 of Figure 4.8, then we know that $N + 1 = (x_m x_{m-1} \ldots x_4 1 x_2 x_1 x_0)_2$, where not all of $x_0, x_1, x_2$ are the same. Of $x_0, x_1, x_2$ select $x_a$ and $x_b$ so that $x_a \neq x_b$. Then, $N' + 1 = (x_m x_{m-1} \ldots x_4 1 x_a x_b)_2$ would have the same transition configuration at the same cell location on $C'_1$ and $C'_2$. Thus for transition pattern 9 or 10, $N' = (x_m x_{m-1} \ldots x_4 1 x_a x_b)_2 - 1$ satisfies the conditions.

Note that, as stated when describing the situation where the 4th “slow” $B$ wave on $C_1$ and $C_2$ is in configuration 1 or 2, in all of these situations, the “slow” $B$ wave on $C'_1$ and $C'_2$ is the 3rd “slow” wave and that the number of bits in $N'$ is $m$, whereas the number of bits in $N$ is $m + 1$.

Now we compare the curves $D_1$ and $D_2$ to $D'_1$ and $D'_2$. Since $C_1$ and $C_2$ are equal to $C'_1$ and $C'_2$ from cell $\sum_{k=4}^{m} x_k 2^{k-4}$ to cell 0, given the above description of finding the appropriate $N'$, and since the transition patterns of the “slow” waves from $C_1$ and $C_2$ to $D_1$ and $D_2$ are the same as those of the “slow” waves from $C'_1$ and $C'_2$ to $D'_1$ and $D'_2$, also from finding the appropriate $N'$ as above, then $D_1$ and $D_2$ are equal to $D'_1$ and $D'_2$ from cell $\sum_{k=4}^{m} x_k 2^{k-4}$ to cell 0 since we are applying the same set of transition rules.
Thus, the position of the 4th “slow” wave on $D_1$ and $D_2$ is in the same position as the 3rd “slow” wave on $D'_1$ and $D'_2$ - namely cell \( \sum_{k=4}^{M} y_k 2^{k-4} \) (recall that the binary expansion of $N + 2$ is \((y_M y_{M-1} \ldots y_1 y_0)_2\)). Hence, $N + 2$ and $N' + 2$ have the same binary expansions for $M - 3$ bits, though they may differ for the remaining 4 bits. Then, since we know from assumption that Lemma 2 part b holds for $i = N' + 1$ and since for all $q$ such that $4 \leq q \leq M - 1$, we have matching conditions regarding the bit patterns for the $q$th “slow” wave for $i = N + 1$ and the $(q-1)$st “slow” wave for $i = N' + 1$, we have that Lemma 2 part b holds for $i = N + 1$.

It only remains to show part c. But as a result of the above, we know that $D_1(0) = D'_1(0)$ and since we assumed in our induction hypothesis that all of Lemma 2 holds for all $i$ such that $2 \leq i \leq N$, we know that $D'_1(0) = G$ and thus $D_1(0) = G$, so part c holds for $i = N + 1$.

Hence by the principle of mathematical induction, Lemma 2 holds for squads of any finite length $n$.

\[ \square \]

**Lemma 3. (Finding the middle points of the squad array)**

Consider a firing squad of length $n \geq 6$. Let the binary representation of $n - 1$ be $(z_m z_{m-1} \ldots z_1 z_0)_2$, where $m \geq 1$ and $z_m = 1$. That is, let $n - 1 = \sum_{k=0}^{m} z_k 2^k$ with $z_m = 1$, $m \geq 1$.

If $n - 1 \in 2\mathbb{Z}$, let $C_3$ be the arctan $\left( \frac{1}{2} \right)$ degree diagonal curve beginning at \( (n - 1, \frac{n-1}{2}) \) and ending at $(2, n - 2)$ and let $C_4$ be the adjacent arctan $\left( \frac{1}{2} \right)$ degree diagonal curve beginning at \( (n - 2, \frac{n-1}{2} + 1) \) and ending at $(3, n - 2)$. If $n - 1 \in \mathbb{Z} \setminus 2\mathbb{Z}$, then let $C_3$ be the arctan $\left( \frac{1}{2} \right)$ degree diagonal curve beginning at \( (n - 2, \frac{n}{2}) \) and ending at $(2, n - 2)$ and let $C_4$ be the adjacent arctan $\left( \frac{1}{2} \right)$ degree diagonal curve beginning at \( (n - 1, \frac{n}{2}) \) and ending at $(3, n - 2)$.

Using the set of states $S^*$ (defined on page 31) and rule set $R_1$ (made up of the rules found
in Tables 3.1 and 4.1), and for all \( i \) such that \( 0 \leq i \leq m - 1 \) the following hold:

a. If \( \sum_{k=i}^{m} z_k 2^{k-i} \in 2\mathbb{Z} \), then \( C_3 \left( \sum_{k=i}^{m} z_k 2^{k-i} \right) = G \). If \( \sum_{k=i}^{m} z_k 2^{k-i} \in \mathbb{Z} \setminus 2\mathbb{Z} \), then \( C_4 \left( \sum_{k=i}^{m} z_k 2^{k-i} \right) = G \).

b. If \( z_i = 0 \) and \( i < m - 1 \) then for all \( j \) such that \( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \leq j \leq -1 + \sum_{k=i}^{m} z_k 2^{k-i} \), if \( j \in 2\mathbb{Z} \) then \( C_3(j) = A_2 \) and if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \) then \( C_4(j) = Q_1 \).

Note that when \( z_i = 0 \), the cell in position \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \) is the middle cell of cells 0 through \( \sum_{k=i}^{m} z_k 2^{k-i} \).

c. If \( z_i = 1 \) and \( i < m - 1 \), then

i. For all \( j \) such that \( 2 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \leq j \leq -1 + \sum_{k=i}^{m} z_k 2^{k-i} \), if \( j \in 2\mathbb{Z} \) then \( C_3(j) = A_2 \) and if \( j \in \mathbb{Z} \setminus 2\mathbb{Z} \) then \( C_4(j) = Q_1 \).

ii. If \( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in 2\mathbb{Z} \) then \( C_3 \left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) = B_1 \) and if \( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in \mathbb{Z} \setminus 2\mathbb{Z} \) then \( C_4 \left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) = G \)

Note that when \( z_i = 1 \), the two cells in positions \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \) and \( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \) are the middle cells of cells 0 through \( \sum_{k=i}^{m} z_k 2^{k-i} \).

Proof. Consider a firing squad array of length \( n \geq 6 \). We will prove by induction.

First, we show that Lemma 3 part a. holds when \( i = 0 \). Assume that \( n \) is even and so \( n - 1 \) is odd. We have \( C_3 \) beginning at \( \left( n - 2, \frac{n}{2} \right) \) and \( C_4 \) beginning at \( \left( n - 1, \frac{n}{2} \right) \). Since \( n \) is even, we know from Lemma 1 that the squad at time \( t = \frac{n - 2}{2} \) is as follows, where \( I \) stands for an irrelevant state:

\[ **II\ldotsIQ_2A_1Q_1** \]

54
Then applying the rule \((Q_2 A_1 Q_1 **) \rightarrow G\) we see that \(C_4 \left( \sum_{k=0}^{m} z_k 2^k \right) = C_4(n - 1) = G\).

Now assume that \(n\) is odd and so \(n - 1\) is even. We have \(C_3\) beginning at \(\left( n - 1, \frac{n - 1}{2} \right)\) and \(C_4\) beginning at \(\left( n - 2, \frac{n - 1}{2} + 1 \right)\). Since \(n\) is odd, we know from Lemma 1 that the squad at time \(t = \frac{n - 3}{2}\) is as follows, where \(I\) stands for an irrelevant state:

\[**II \ldots IA_1Q_1Q_1**\]

Then applying the rule \((A_1Q_1Q_1**) \rightarrow G\) we see that \(C_3 \left( \sum_{k=0}^{m} z_k 2^k \right) = C_3(n - 1) = G\).

Thus Lemma 3 part a. holds for \(i = 0\).

Now assume that part a. holds for \(i \leq j\). First, assume that \(z_j = 0\). Then, by applying Lemma 2 for \(i = n - 2\) and \(i = n - 3\), and by using rules \((Q_2, A_1, Q_2, G, Q_1) \rightarrow Q_1, (A_1, Q_2, A_1, Q_2, G) \rightarrow A_2, (Q_2, A_1, Q_2, A_2, Q_1) \rightarrow Q_1, (A_1, Q_2, A_1, Q_2, A_2) \rightarrow A_2, (B_1, Q_2, A_1, Q_2, A_2) \rightarrow A_2, (Q_2, Q_2, B_2, A_2, Q_1) \rightarrow Q_1, (Q_2, A_1, B_2, A_2, Q_1) \rightarrow G, (Q_2, A_1, Q_2, A_2, G) \rightarrow Q_1, (Q_2, B_1, A_1, Q_2, A_2) \rightarrow G, (B_1, A_1, Q_2, A_2, Q_1) \rightarrow Q_1, (B_2, Q_2, A_1, Q_2, A_2) \rightarrow A_2, (Q_2, B_2, Q_2, A_2, Q_1) \rightarrow Q_1, (Q_2, Q_2, B_2, Q_2, A_2) \rightarrow G\) we have the possibilities in Figure 4.9.

So if \(z_j = 0\), we have that part a. holds for \(i = j + 1\) and that part b. holds for \(i = j\).
Now assume that \( z_j = 1 \). Then, by applying Lemma 2 for \( i = n - 2 \) and \( i = n - 3 \), and by Figure 4.10: Finding Middle Points with \( z_j = 1 \)

\( C_3 \) and \( C_4 \) are dark and light orange, respectively. \( C_1 \) and \( C_2 \) for \( i = n - 2 \) are blue and gold, respectively.

\( S \) refers to a state that is either \( A_1 \) or \( Q_2 \), \( T \) refers to a state that is \( A_1 \) or \( B_1 \), \( R \) refers to a state that is \( B_1 \) or \( B_2 \)

using rules \( (A_1, Q_2, A_1, Q_2, A_2) \to A_2 \), \( (Q_2, A_1, Q_2, A_2, G) \to Q_1 \), \( (Q_2, A_1, Q_2, A_2, Q_1) \to Q_1 \), \( (B_2, Q_2, A_1, Q_2, A_2) \to A_2 \), \( (Q_2, Q_2, B_1, Q_2, A_2) \to G \), \( (A_1, Q_2, B_1, Q_2, A_2) \to G \), \( (Q_2, B_1, Q_2, A_2, Q_1) \to G \), \( (B_2, A_1, Q_2, A_2, Q_1) \to Q_1 \), \( (A_1, B_2, A_1, Q_2, A_2) \to B_1 \), \( (Q_2, B_2, A_1, Q_2, A_2) \to B_1 \), and \( (Q_2, Q_2, B_1, B_1, Q_1) \to G \), we have the possibilities in Figure 4.10.

So if \( z_j = 1 \), we have that part a. holds for \( i = j + 1 \) and that part c. holds for \( i = j \). Thus by the principle of mathematical induction, Lemma 3 holds for all \( i \) with \( 0 \leq i \leq m - 1 \).

**Lemma 4. (Reflection of return wave at middle points)**

Consider a firing squad of length \( n \geq 6 \). Let the binary representation of \( n - 1 \) be \((z_m z_{m-1} \ldots z_1 z_0)_2\), where \( m \geq 2 \) and \( z_m = 1 \). That is, let \( n - 1 = \sum_{k=0}^{m} z_k 2^k \) with \( z_m = 1 \), \( m \geq 2 \).

If \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in 2\mathbb{Z} \), then let \( C_{3,i} \) be the \( \arctan \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - 1 - \frac{1}{2} \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \) and ending at \((2, n - 2)\) and let \( C_{4,i} \) be the adjacent \( \arctan \left( \frac{1}{2} \right) \) degree diagonal curve beginning at \( \left( -1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - 1 - \frac{1}{2} \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \) and ending at \((3, n - 2)\). If \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in \mathbb{Z} \setminus 2\mathbb{Z} \), then let \( C_{3,i} \) be the \( \arctan \left( \frac{1}{2} \right) \) degree diagonal curve beginning at...
\[
\left(-1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \left(1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}\right)\right) \text{ and ending at } (2, n-2) \text{ and let } C_{4,i} \text{ be the adjacent } \arctan \left(\frac{1}{2}\right) \text{ degree diagonal curve beginning at } \\
\left(\sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \left(1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}\right)\right) \text{ and ending at } (3, n-2).
\]

Using the set of states \(S^*\) (defined on page 37) and rule set \(R_1\) (containing the rules from Tables 3.1 and 4.1), the following are true:

a. If \(z_k = 0\), and \(\sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in 2\mathbb{Z}\), then let \(C_{5,i}\) be the \(-\arctan \left(\frac{1}{2}\right)\) degree diagonal curve beginning at \((m \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \left(1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}\right))\) and ending at \((3, n-2)\).

\[\text{Figure 4.11: Diagonals for Lemma } 4\text{ if } z_i = 0\]

\(C_{3,i}\) in blue, \(C_{4,i}\) in yellow, \(C_{5,i}\) in purple, \(C_{6,i}\) in orange. Green indicates a cell that is both in \(C_{4,i}\) and \(C_{6,i}\)
If \( z_k = 0 \) and \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in \mathbb{Z} \setminus 2\mathbb{Z} \), then let \( C_{5,i} \) begin at 
\[
\left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \right) \]
and end at 
\[
\left( -2 + \sum_{k=i}^{m} z_k 2^{k-i}, n - 2 \right) \]
and let \( C_{6,i} \) begin at 
\[
\left( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \right) \]
and end at 
\[
\left( -3 + \sum_{k=i}^{m} z_k 2^{k-i}, n - 2 \right). \]

Then, \( C_{5,i} \) and \( C_{6,i} \) are images (as in Definition 5) of \( C_{3,i} \) and \( C_{4,i} \).

b. If \( z_i = 1 \) and \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in 2\mathbb{Z} \), then let \( C_{5,i} \) be the \(-\arctan\left(\frac{1}{2}\right)\) degree diagonal

curve beginning at 
\[
\left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \]
and ending at 
\[
\left( -2 + \sum_{k=i}^{m} z_k 2^{k-i}, n - 2 \right) \]
and let \( C_{6,i} \) be the adjacent \(-\arctan\left(\frac{1}{2}\right)\) degree diagonal curve
beginning at 
\[
\left( 2 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \]
and ending at 
\[
\left( -3 + \sum_{k=i}^{m} z_k 2^{k-i}, n - 2 \right). \]

Figure 4.12: Diagonals for Lemma 4 if \( z_i = 1 \) 
\( C_{3,i} \) in blue, \( C_{4,i} \) in yellow, \( C_{5,i} \) in purple, \( C_{6,i} \) in orange.
If \( z_i = 1 \) and \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in \mathbb{Z} \setminus 2\mathbb{Z} \), then let \( C_{5,i} \) begin at
\[
\left( 2 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \right)
\] and end at \( \left( -2 + \sum_{k=i}^{m} z_k 2^{k-i}, n - 2 \right) \) and let \( C_{6,i} \) begin at
\[
\left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)}, n - \frac{1}{2} \left( 1 + \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \right) \right)
\] and end at \( \left( -3 + \sum_{k=i}^{m} z_k 2^{k-i}, n - 2 \right) \).

Then, \( C_{5,i} \) and \( C_{6,i} \) are images (as in Definition 3) of \( C_{3,i} \) and \( C_{4,i} \).

**Proof.** First, let \( z_i = 0 \). So we know that \( \sum_{k=i}^{m} z_k 2^{k-i} \in 2\mathbb{Z} \). Consider the triangular region enclosed by the following curves:

The vertical curve starting at \( \left( \sum_{k=i}^{m} z_k 2^{k-i}, n - 1 - \frac{1}{2} \sum_{k=i}^{m} z_k 2^{k-i} \right) \) and ending at
\[
\left( \sum_{k=i}^{m} z_k 2^{k-i}, n - 2 \right)
\]

The double diagonal curves starting at \( \left( \sum_{k=i}^{m} z_k 2^{k-i}, n - 1 - \frac{1}{2} \sum_{k=i}^{m} z_k 2^{k-i} \right) \) and
\[
\left( -1 + \sum_{k=i}^{m} z_k 2^{k-i}, n - \frac{1}{2} \sum_{k=i}^{m} z_k 2^{k-i} \right)
\] and ending at the starting position for \( C_{3,i} \) and \( C_{5,i} \), respectively, if \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in 2\mathbb{Z} \). If \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in \mathbb{Z} \setminus 2\mathbb{Z} \), have the double diagonal curves end at the starting position for \( C_{3,i} \) and \( C_{4,i} \), respectively.

We know the double diagonal curves follow Lemma 3 and that the vertical curve is in state \( G \) throughout, due to rule set \( R_1 \) requiring that a \textit{General} remain in the \textit{General} state until the few conditions specified in the tables which result in a \textit{General} firing, none of which appear in this situation. Call this triangular area \( T_1 \).

Consider also the triangular area in the space-time diagram for a squad of length \( N = \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \) enclosed by the vertical curve starting at \( (0,0) \) and ending at \( (0,N - 2) \) and the double diagonals starting at \( (0,0) \) and \( (1,1) \) and ending at \( \left( N - 1, \frac{N - 1}{2} \right) \) and
\( \left( N - 2, \frac{N - 1}{2} \right) \), respectively, if \( N \in \mathbb{Z} \setminus 2\mathbb{Z} \). If \( N \in 2\mathbb{Z} \), then have the double diagonals end at \( \left( N - 2, \frac{N - 2}{2} \right) \) and \( \left( N - 1, \frac{N}{2} \right) \), respectively. Call this triangular area \( T_2 \).

We know that the double diagonals of \( T_2 \) follow Lemma 1 and the vertical curve remains in \( G \). So, by Lemmas 1 and 3 we see the double diagonals of \( T_1 \) and \( T_2 \) are images of each other (using the function \( \text{Im}(s) \) defined on page 31). Then, since \( \text{Im}(G) = G \) as well, we know from Theorem 2 that \( T_1 \) and \( T_2 \) are images of each other. Thus the double diagonals that close \( T_1 \) and \( T_2 \) are also images of each other. Note that the closing diagonals of \( T_1 \) are \( C_{5,i} \) and \( C_{6,i} \). Then, due to Lemma 3 we also have that the closing double diagonals for \( T_2 \) are equal to the double diagonals \( C_{3,i} \) and \( C_{4,i} \). Hence, \( C_{5,i} \) and \( C_{6,i} \) are images of \( C_{3,i} \) and \( C_{4,i} \).

Now assume \( z_i = 1 \). We know from Lemma 3 that we have two possibilities for arrangements of states depending on if \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \) is even or odd (see Figure 4.10). First assume that \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in 2\mathbb{Z} \). Then, by Lemmas 2 and 3 and Figure 4.10, we can apply rules from rule set \( R_1 \) to obtain the configuration in Figure 4.13.

In particular, the rules \((B_1, Q_2, A_2, Q_1, I) \rightarrow Q_1\), \((A_1, Q_2, A_2, Q_1, I) \rightarrow A_2\),

\[
\begin{array}{ccccccc}
& & & & & & \text{Q2, A2, Q1, I} \\
& & & & & \text{G} \\
& & & & \text{G} \\
& & & \text{G} \\
& & \text{G} \\
& \text{G} \\
\end{array}
\]

Figure 4.13: \( z_i = 1 \) and \( z_{i+1} = 0 \) reflection of waves
\( C_{3,i} \) in blue, \( C_{4,i} \) in yellow, \( C_{5,i} \) in purple, \( C_{6,i} \) in orange

\( (Q_2, A_2, Q_1, A_2, I) \rightarrow Q_1\), \((G, G, Q_1, Q_1, I) \rightarrow Q_2\), \((A_2, Q_1, A_2, Q_1, I) \rightarrow A_2\),
(G, Q_1, Q_1, A_2, I) \to A_1, (Q_1, A_2, Q_1, A_2) \to Q_1, (Q_1, Q_1, A_2, Q_1, I) \to Q_1, (Q_2, A_1, Q_1, Q_1, I) \to Q_2 are used. Using the function Im(s) as defined on page 31 we can see that C_{5,i} and C_{6,i} are images of C_{3,i} and C_{4,i} so far.

By using Lemma 3 we know that C_{3,i} and C_{4,i} continue on the same way until they meet the i + 2nd slow wave. Since the \ell\textsuperscript{th} slow wave intersects at the cell in position \( \frac{1}{2^\ell}(n-1) \) at time \( t = \frac{2^{\ell+1} - 1}{2^{\ell+1}}(n-1) \), we know the number of time steps between the intersection at middle point \( \sum_{k=i+1}^{m} z_k 2^{k-i} \) and the intersection at middle point \( \sum_{k=i+2}^{m} z_k 2^{k-(i+2)} \) is \( \frac{2^{i+3} - 1}{2^{i+3}}(n-1) - \frac{2^{i+2} - 1}{2^{i+2}}(n-1) = \frac{1}{2^{i+3}}(n-1) \) time steps. So C_{3,i} and C_{4,i} continue in their same pattern until \( \frac{1}{2^{i+3}}(n-1) \) time steps have passed since the intersection at \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \).

Meanwhile, C_{5,i} and C_{6,i} continue unchanged until the first slow wave emitted from the middle point at \( \sum_{k=i}^{m} z_k 2^{k-i} \) intersects. The first slow wave emitted from the cell in position \( \sum_{k=i}^{m} z_k 2^{k-i} \) travels to the midpoint between \( \sum_{k=i+1}^{m} z_k 2^{k-i} \) and \( \sum_{k=i+2}^{m} z_k 2^{k-(i+2)} \). So the slow wave needs to travel \( \frac{1}{2} \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} = \sum_{k=i+2}^{m} z_k 2^{k-(i+2)} = \frac{1}{2^{i+2}}(n-1) \) cells. Since this is the first slow wave emitted from this middle point, it travels 2 cells every 3 time steps, hence will reach this new middle point in a total of \( \frac{3}{2} \left( \frac{1}{2^{i+2}}(n-1) \right) = \frac{3}{2^{i+3}}(n-1) \) time steps. However, by the time the \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \) intersection begins generating our waves C_{3,i}, C_{4,i}, C_{5,i}, C_{6,i}, the first slow wave from the \( \sum_{k=i+2}^{m} z_k 2^{k-(i+2)} \) middle point has already been traveling for \( \frac{1}{2} \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} = \sum_{k=i+2}^{m} z_k 2^{k-(i+2)} = \frac{1}{2^{i+2}}(n-1) \) time steps. Hence, starting when the middle cell \( \sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \) begins generating our C_{3,i}, C_{4,i}, C_{5,i}, C_{6,i}, the first slow wave from position \( \sum_{k=i}^{m} z_k 2^{k-i} \) will meet our waves C_{5,i} and C_{6,i} in another \( \frac{3}{2^{i+3}}(n-1) - \frac{1}{2^{i+2}}(n-1) = \frac{1}{2^{i+3}}(n-1) \) time steps.

Hence the pair C_{3,i} and C_{4,i} meet their slow wave in the same number of time steps as it takes C_{5,i} and C_{6,i} to meet their slow wave. Hence, we can see that C_{5,i} and C_{6,i} are
images of $C_{3,i}$ and $C_{4,i}$.

Now consider if $\sum_{k=i+1}^{m} z_k 2^{k-(i+1)} \in \mathbb{Z} \setminus 2\mathbb{Z}$. Then, by Lemmas 2 and 3 and Figure 4.10, we can apply rules from rule set $R_1$ to obtain the configuration in Figure 4.14.

In particular, the rules $(A_1, Q_2, A_2, Q_1, I) \to A_2$, $(Q_2, A_2, Q_1, A_2, I) \to Q_1$, $(B_1, B_1, Q_1, A_2, Q_1) \to A_1$, $(B_1, Q_1, A_2, Q_1, I) \to Q_1$, $(G, A_1, Q_1, Q_1, I) \to Q_2$, $(A_1, Q_1, Q_1, A_2, I) \to A_1$, $(Q_1, A_2, Q_1, A_2, I) \to Q_1$, $(Q_1, Q_1, A_2, Q_1, I) \to Q_1$, and $(Q_2, A_1, Q_1, Q_1, I) \to Q_2$ are used. Using the function $Im(s)$ as defined on page 31, we can see that $C_{5,i}$ and $C_{6,i}$ are images of $C_{3,i}$ and $C_{4,i}$ so far. However, the same argument as above holds as to when the next midpoints will be found, and thus we can see that $C_{5,i}$ and $C_{6,i}$ are images of $C_{3,i}$ and $C_{4,i}$.

Now we provide the proof for Theorem 3.

**Proof of Theorem 3.** We need to show that, using set of states $S^*$, as on page 31 with rule set $R_1$, outlined in Tables 3.1 and 4.1, for any $n$, the one-dimensional array of size $n$ will synchronize with $S(i, n-1) = F$ for $0 \leq i \leq n-1$ and $S(i, t) \neq F$ for $0 \leq i \leq n-1$ and $t < n-1$. 

62
This was verified directly using the simulation program for squads of any size $n$ with $2 \leq n \leq 500$.

By Lemmas 1, 2, 3, and 4, we have that the curves $C_{3,1}$ and $C_{4,1}$ for a squad of size $n$ are equivalent to the fast return wave of a squad of size $\left\lceil \frac{n}{2} \right\rceil$, which takes $\left\lceil \frac{n-1}{4} \right\rceil$ time steps to fire. $C_{3,1}$ and $C_{4,1}$ are generated at the first middle point of the squad of size $n$ at time $\left\lceil \frac{3}{4} (n-1) \right\rceil$. Adding these times together shows that the left half of the squad of size $n$ fires in $n-1$ time steps. The right side, by Lemma 4, also fires in a total of $n-1$ time steps.

Hence a squad of size $n$ fires in $n-1$ time steps and we have a minimal time solution. □

**Theorem 4.** The set of states $S^*$, as on page 34, with rule set $R_0$, outlined in Table 3.1, is a minimal time solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2. That is, for any size $n$, the one-dimensional array of size $n$ will synchronize with $S(i, n-1) = F$ for $0 \leq i \leq n-1$ and $S(i, t) \neq F$ for $0 \leq i \leq n-1$ and $t < n-1$.

**Proof.** The rule set $R_0$ is contained in the rule set $R_1$. The rules in $R_0$ are necessary rules, as all 651 rules in rule set $R_0$ are used at least once in running simulations of firing squads of size $n$ with $2 \leq n \leq 211$.

We will show the rules in $R_0$ are also sufficient. Of the added rules in Table 4.1 to create rule set $R_1$:

The following 13 rules cannot exist due to the initial set up of the firing squad array $\ast \ast GQ_1Q_1\ldots Q_1 \ast \ast$ and the fact that the *General* state remains in *General* until the conditions are met for firing:
\((*, Q_2, Q_2, G, *) \rightarrow Q_1\)

\((*, *, Q_2, G, *) \rightarrow F\)

\((*, Q_2, Q_2, Q_2, G) \rightarrow A_2\)

\((*, *, Q_2, Q_2, G) \rightarrow G\)

\((*, Q_2, Q_2, Q_2, Q_2) \rightarrow Q_2\)

\((*, *, Q_2, Q_2, Q_2) \rightarrow Q_2\)

\((*, Q_2, Q_2, A_2, Q_1) \rightarrow Q_1\)

\((*, *, Q_2, A_2, Q_1) \rightarrow G\)

\((*, Q_2, Q_2, Q_2, A_2) \rightarrow A_2\)

\((*, *, Q_2, Q_2, A_2) \rightarrow G\)

\((*, Q_2, G, * ,*) \rightarrow F\)

\((*, Q_2, A_2, Q_1, A_2) \rightarrow A_1\)

\((*, Q_2, A_2, Q_1, G) \rightarrow B_1\)

The following 6 rules cannot exist because of the fast \(A_1\) wave traveling 2 cells per time step from the initial \textit{General}: 

\((*, G, A_1, Q_1, A_2) \rightarrow B_1\)

\((*, G, A_1, Q_1, B_1) \rightarrow Q_2\)

64
\((*, G, B_1, Q_2, A_1) \rightarrow Q_2\)

\((*, G, Q_2, B_1, A_1) \rightarrow B_1\)

\((*, G, Q_2, B_2, A_1) \rightarrow Q_2\)

\((*, *, G, A_1, Q_1) \rightarrow G\)

The following 3 rules cannot exist because of the fact that \(A_2\) is the fast return wave and is about to hit the original \textit{General}, which should result in \textit{Fire}:

\((*, G, Q_1, A_2, B_1) \rightarrow Q_2\)

\((*, G, Q_1, A_2, Q_1) \rightarrow Q_2\)

\((*, *, G, Q_1, A_2) \rightarrow G\)

Both \((Q_2, Q_2, Q_2, G, *) \rightarrow Q_1\) and \((Q_2, Q_2, G, *, *) \rightarrow G\) are not necessary as they would be for having an initial set up with the \textit{General} on the right side of the firing squad.

\((*, *, G, Q_1, G) \rightarrow F\) cannot exist as Theorem 3 showed that the squad of size \(n\) has \(C_{3,1}\) and \(C_{4,1}\) equal to the fast return wave of a squad of size \(\lceil \frac{n}{2} \rceil\). Iteratively repeating this, a squad will \textit{Fire} when the states at time \(n - 2\) are repetitions of the second-to-last time step of a squad of size 4, 5, or 6. Examining squads of these sizes, we see the only possibilities for this rule at time \(n - 2\) are \((*, *, G, B_1, B_1) \rightarrow F\) or \((*, *, G, Q_2, G) \rightarrow F\).

The last added rule of Table 4.1 is \((*, G, Q_1, G, *)\), which could only arise in a squad of size 3. Actually running the simulation for a squad of size 3, however, yields the following:
So that last added rule never occurs.

Hence, since Theorem 3 shows that the set of states \( S^* \) with rule set \( R_1 \) is a minimal time solution, and since we have shown that the rules in rule set \( R_0 \) are necessary and sufficient, we have that \( S^* \) with rule set \( R_0 \) consisting of the 651 rules outlined in Table 3.1 is a minimal time solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2.
5 Conclusion and Suggestions for Future Work

We developed a new variation of the Firing Squad Synchronization Problem using a different neighborhood definition - the Von Neumann Neighborhood of extent 2. In solving this new variation of the FSSP, we found a minimal time solution which synchronizes an array of any finite length in half the number of times steps as needed in the standard FSSP. This reduction in discrete time steps is due to the ability of information to travel at twice the speed as in the standard problem, because of the extended radius of the neighborhoods used. The technique used to solve the problem, the halving technique, has been used by many others in solving the standard problem, including Waksman, Balzer, and Noguchi [1, 2, 6, 7].

Although the number of discrete time steps needed is half that of the required time steps in the standard problem, the number of rules in this solution, 651, is larger than the number of rules in minimal time solutions to the standard FSSP that use the same technique of halving and have the same number of possible states, 8. Balzer’s 8 state minimal time solution is stated to have 182 rules, however Noguchi claims that only 165 of these are actually used, and Noguchi’s 8 state minimal time solution with the least number of rules has 119 rules [1, 7]. However, regardless of the number of rules, in implementing any of these solutions using a simulation program similar to our code found on page 69, the complexity of the algorithm remains the same.

The 651 rule 8 state minimal time solution to the Firing Squad Synchronization Problem with Von Neumann Neighborhood of Extent 2 was proven with induction using properties of image solutions and many patterns that arise due to the binary nature of repeated halving of the firing squad array.

More work can be done in examining possible other minimal time solutions to the FSSP
with Von Neumann Neighborhood of Extent 2 to attempt to reduce the number of states in a solution or reduce the number of rules. Different techniques other than halving could also be explored, such as dividing the array in thirds as Mazoyer did in the standard problem [6]. Due to the rules having more inputs when considered a Von Neumann Neighborhood of Extent 2, there are more possible rules that can be defined for a fixed number of states than with the standard Von Neumann Neighborhood. Having a larger number of possible configurations of states in the rule set could feasibly result in solutions with fewer states than what is possible in the standard problem. Exploring that possibility would be of interest.

It is also natural to now consider changing the neighborhood definition used in other variations of the FSSP, including the ring variation as in [10] or in higher dimensions as in [5]. Expanding the radius of the neighborhoods in these variations could result in minimal time solutions that require fewer discrete time steps. Different types of neighborhoods can also be explored, including an even larger radius, neighborhoods defined only on one side of a cell, or neighborhoods with unequal number of neighbors on each side.

Solutions to variations of the FSSP could be used in applications requiring simultaneous action by machines. For example, in a future with all self-driving cars, a solution to the FSSP could be used to allow for simultaneous movement of cars stopped at a traffic light when the signal changes to green, as opposed to the current reality that each car can only move once all preceding cars have already started moving. Solutions could also be used for simultaneous engine firing in rockets or other applications requiring synchronous action.
import sys
import numpy as np
from termcolor import colored
from colored import fg

#prompt user for the size of the firing squad
squad_size = int(input('Enter the length of the firing squad: '))

#initialize 1d array for firing squad with all entries 0 (quiescent state)
squad = np.zeros(squad_size + 4, dtype=np.int)

#set the four end units’ state to boundary (20)
squad[0]=20
squad[1]=20
squad[squad_size+2]=20
squad[squad_size+3]=20

#initialize first member of the squad as the general (state 1)
squad[2]=1

for k in range(2, squad_size+2):
    if squad[k] == 0:
        sys.stdout.write('%s\u2610' %(fg('white')))
    elif squad[k] == 1:
        sys.stdout.write('%s\u2610' %(fg('green')))
elif squad[k] == 2:
    sys.stdout.write('%s\u2610' %(fg('red')))
elif squad[k] == 3:
    sys.stdout.write('%s\u2610' %(fg('yellow')))
elif squad[k] == 4:
    sys.stdout.write('%s\u2610' %(fg('magenta')))
elif squad[k] == 5:
    sys.stdout.write('%s\u2610' %(fg('blue')))
elif squad[k] == 6:
    sys.stdout.write('%s\u2610' %(fg('cyan')))
elif squad[k] == 7:
    sys.stdout.write('%s\u2610' %(fg('orchid')))
else:
    sys.stdout.write('')
sys.stdout.write('
')

# create the array to hold the rule set
rules = np.full((21,21,21,21,21), 21, dtype=np.int)
# define rules
rules[20,20,1,0,0]=1 # general persists
rules[20,1,0,0,0]=3
rules[1,0,0,0,0]=4 # fast wave
rules[0,0,0,0,0]=0 # no instantaneous change from quiescent
rules[0,0,0,0,20]=0 # no instantaneous change from quiescent
rules[0,0,20,20]=0 # no instantaneous change from quiescent
rules[20,20,1,3,4]=1 # general persists
rules[20,1,3,4,0]=7 # start slow wave
rules[1,3,4,0,0]=4
rules[3,4,0,0,0]=3
rules[4,0,0,0,0]=4  # fast wave
rules[20,20,1,7,4]=1  # general persists
rules[20,1,7,4,3]=3
rules[1,7,4,3,4]=7  # slow wave
rules[7,4,3,4,0]=3
rules[4,3,4,0,0]=4
rules[20,20,1,3,7]=1  # general persists
rules[20,1,3,7,3]=3
rules[1,3,7,3,4]=4  # fast wave
rules[3,7,3,4,3]=6  # slow wave
rules[7,3,4,3,4]=4
rules[3,4,3,4,0]=3
rules[20,1,3,4,6]=7  # slow wave
rules[1,3,4,6,4]=3
rules[3,4,6,4,3]=7  # slow wave
rules[4,6,4,3,4]=4
rules[6,4,3,4,3]=3
rules[4,3,4,3,4]=4
rules[20,1,7,3,7]=7  # slow wave
rules[1,7,3,7,4]=4  # fast wave
rules[7,3,7,4,3]=3
rules[3,7,4,3,4]=7  # slow wave
rules[7,4,3,4,3]=3
rules[3,4,3,4,3]=3
rules[1,7,4,3,7]=6  # slow wave
rules[7,4,3,7,3]=3
rules[4,3,7,3,4]=4  # fast wave
rules[3,4,0,0,20]=3
rules[4,0,0,20,20]=1  # fast wave hits far end
rules[20,20,1,3,6]=1  # general persists
rules[20,1,3,6,3]=3
rules[1,3,6,3,4]=7
rules[3,6,3,4,6]=3
rules[6,3,4,6,4]=3
rules[4,3,4,3,1]=5  # fast reverse wave
rules[3,4,3,1,20]=0
rules[4,3,1,20,20]=1  # general persists
rules[1,3,7,3,3]=7  # slow wave
rules[3,7,3,3,7]=3
rules[7,3,7,4]=4
rules[3,3,7,4,3]=3
rules[7,3,4,3,7]=3
rules[3,4,3,7,3]=3
rules[7,3,4,3,5]=5  # fast reverse wave
rules[4,3,5,0,5]=5
rules[3,5,0,5,7]=0
rules[5,0,5,7,1]=7
rules[0,5,7,1,20]=0
rules[5,7,1,20,20]=1  # general persists
rules[1,3,4,6,3]=3
rules[3,4,6,3,3]=6  # slow wave
rules[4,6,3,3,4]=4  # fast wave
rules[6,3,3,4,6]=3
rules[3,3,4,6,5]=5  # fast reverse wave
rules[3,4,6,5,0]=1  # intersection of reverse fast and forward slow waves
rules[4,6,5,0,5]=4  # fast wave
rules[6,5,0,5,0]=0
rules[5,0,5,0,7]=5
rules[0,5,0,7,0]=6
rules[5,0,7,0,1]=5  # fast reverse wave
rules[0,7,0,1,20]=0
rules[7,0,1,20,20]=1  # general persists
rules[20,20,1,7,3]=1  # general persists
rules[20,1,7,3,6]=7  # slow wave
rules[1,7,3,6,4]=3
rules[7,3,6,4,3]=7  # slow wave
rules[3,6,4,3,5]=7  # slow wave
rules[6,4,3,5,1]=0
rules[4,3,5,1,4]=7  # slow wave
rules[3,5,1,4,0]=1  # general persists
rules[5,1,4,0,5]=7  # slow wave
rules[1,4,0,5,6]=3
rules[4,0,5,6,5]=7  # slow wave
rules[0,5,6,5,0]=7  # slow wave
rules[5,6,5,0,1]=0
rules[6,5,0,1,20]=7
rules[4,3,4,3,5]=5  # fast reverse wave
rules[3,4,3,5,0]=0
rules[4,3,5,0,1]=5
rules[3,5,0,1,20]=7  #slow wave
rules[5,0,1,20,20]=1  #general persists
rules[1,7,3,7,7]=7
rules[7,3,7,7,0]=1
rules[3,7,7,0,7]=1
rules[7,7,0,7,1]=7
rules[7,0,7,1,7]=7
rules[0,7,1,7,3]=1  #general persists
rules[7,1,7,3,7]=7
rules[7,0,7,1,20]=7
rules[0,7,1,20,20]=1  #general persists
rules[20,20,1,7,7]=2
rules[20,1,7,7,1]=2
rules[1,7,7,1,1]=2
rules[7,7,1,1,7]=2
rules[7,1,1,7,7]=2
rules[1,1,7,7,1]=2
rules[1,7,7,1,7]=2
rules[7,7,1,7,7]=2
rules[7,1,7,7,1]=2
rules[1,7,7,1,20]=2
rules[7,7,1,20,20]=2
rules[1,7,4,3,1]=1  #intersection of waves
rules[7,4,3,1,20]=0
rules[20,20,1,3,1]=2
rules[20,1,3,1,0]=2
rules[1,3,1,0,1]=2
rules[3,1,0,1,20]=2
rules[1,0,1,20,20]=2
rules[7,3,4,3,1]=5
rules[1,3,4,6,5]=7
rules[4,6,5,0,1]=7
rules[3,7,4,3,5]=1  # intersection of waves
rules[7,4,3,5,0]=0
rules[7,4,3,1,0]=0
rules[4,3,1,0,5]=1  # general persists
rules[3,1,0,5,7]=3
rules[1,0,5,7,1]=1
rules[3,1,0,1,3]=2
rules[1,0,1,3,1]=2
rules[0,1,3,1,0]=2
rules[6,3,4,6,5]=5
rules[6,5,0,5,7]=0
rules[1,3,7,3,5]=1
rules[3,7,3,5,1]=1
rules[7,3,5,1,4]=0
rules[5,1,4,0,7]=3
rules[1,4,0,7,0]=1
rules[4,0,7,0,1]=1
rules[20,1,3,1,1]=2
rules[1,3,1,1,0]=2
rules[3,1,1,0,1]=2
rules[1,1,0,1,3]=2
rules[0,1,3,1,1]=2
rules[1,1,0,1,20]=2
rules[3,4,3,1,0]=0
rules[3,1,0,5,0]=3
rules[1,0,5,0,7]=4
rules[6,5,0,1,3]=7
rules[5,0,1,3,4]=1
rules[0,1,3,4,6]=7
rules[20,1,0,0,20]=3
rules[1,0,0,20,20]=1
rules[20,1,3,1,20]=2
rules[1,3,1,20,20]=2
rules[3,3,4,6,4]=3
rules[3,6,4,3,3]=3
rules[6,4,3,3,7]=3
rules[4,3,3,7,4]=4
rules[3,5,0,5,0]=0
rules[1,7,3,7,3]=3
rules[7,3,7,3,3]=7
rules[3,7,3,3,4]=4
rules[7,3,3,4,3]=3
rules[3,3,4,3,1]=5
rules[1,0,5,0,5]=4
rules[0,5,0,5,6]=0
rules[5,0,5,6,5]=5
rules[3,5,0,1,3]=7
rules[0,1,3,4,0]=7
rules[1,3,4,0,5]=4
rules[3,4,0,5,7]=3
rules[4,0,5,7,0]=1  #intersection
rules[0,5,7,0,7]=0
rules[5,7,0,7,1]=5
rules[0,5,7,1,7]=0
rules[5,7,1,7,4]=1  #general persists
rules[7,1,7,4,3]=3
rules[3,3,4,3,7]=3
rules[3,7,4,3,3]=6
rules[7,4,3,3,3]=3
rules[4,3,3,3,4]=4
rules[3,3,3,4,6]=3
rules[5,0,5,0,5]=5
rules[1,7,4,3,6]=6
rules[7,4,3,6,3]=3
rules[4,3,6,3,4]=7
rules[3,6,3,4,3]=3
rules[6,3,4,3,5]=5
rules[3,4,3,5,1]=0
rules[1,4,0,5,0]=3
rules[4,0,5,0,5]=4
rules[0,5,0,5,7]=0
rules[5,0,5,7,0]=7
rules[1,3,6,3,7]=6
rules[3,6,3,7,3]=3
rules[6,3,7,3,5]=1  #intersection
rules[3,7,3,5,0]=1  #intersection
rules[7,3,5,0,7] = 0
rules[3,5,0,7,1] = 6
rules[5,0,7,1,7] = 0
rules[7,1,7,3,4] = 3
rules[1,7,3,4,0] = 6
rules[7,3,4,0,7] = 3
rules[3,4,0,7,0] = 1  # intersection
rules[4,0,7,0,5] = 1  # intersection
rules[0,7,0,5,7] = 0
rules[7,0,5,7,1] = 6
rules[1,3,6,3,1] = 1  # intersection
rules[3,6,3,1,1] = 0
rules[6,3,1,1,0] = 1  # general persists
rules[3,1,1,0,6] = 1  # general persists
rules[1,1,0,6,0] = 3
rules[1,0,6,0,1] = 1  # intersection
rules[0,6,0,1,3] = 0
rules[6,0,1,3,6] = 1  # general persists
rules[0,1,3,6,3] = 3
rules[0,6,0,1,20] = 0
rules[6,0,1,20,20] = 1  # general persists
rules[3,1,0,1,1] = 2
rules[1,0,1,1,3] = 2
rules[0,1,1,3,1] = 2
rules[1,1,3,1,0] = 2
rules[6,3,4,3,3] = 3
rules[3,4,3,3,7] = 3
rules[6,3,7,3,3]=7  
rules[3,7,3,3,3]=3  
rules[7,3,3,3,4]=4  
rules[3,3,3,4,3]=3  
rules[0,5,0,5,0]=0  
rules[6,3,7,3,4]=4  
rules[3,4,0,5,0]=3  
rules[4,0,5,0,7]=4  
rules[5,0,7,0,5]=5  
rules[7,4,3,4,6]=3  
rules[4,3,4,6,5]=5  
rules[4,6,5,0,6]=4  
rules[6,5,0,6,0]=0  
rules[5,0,6,0,1]=7  
rules[0,7,0,1,3]=0  
rules[0,1,3,7,3]=3  
rules[7,0,1,3,7]=1  
rules[7,3,4,3,3]=3  
rules[3,4,3,3,3]=3  
rules[6,3,4,6,3]=3  
rules[6,3,3,4,3]=3  
rules[3,3,4,3,5]=5  
rules[3,7,3,3,6]=3  
rules[7,3,3,6,4]=3  
rules[3,3,6,4,3]=7  
rules[4,3,5,0,7]=5  
rules[6,4,3,5,0]=0
rules[7,3,4,0,5]=4
rules[3,4,0,5,6]=3
rules[5,6,5,0,6]=0
rules[7,3,3,7,7]=5
rules[3,3,7,7,0]=1
rules[3,7,7,0,5]=1
rules[7,7,0,5,6]=4
rules[7,0,5,6,0]=0
rules[0,5,6,0,1]=7
rules[5,6,0,1,3]=0
rules[0,1,3,6,4]=3
rules[1,3,6,4,3]=7
rules[3,6,4,3,7]=3
rules[6,4,3,7,7]=5
rules[4,3,7,7,0]=1
rules[3,7,7,0,0]=1
rules[7,7,0,0,7]=4
rules[0,0,7,0,1]=7
rules[7,0,0,7,0]=0
rules[7,3,5,1,1]=0
rules[1,1,4,0,7]=3
rules[3,5,1,1,4]=1
rules[5,1,1,4,0]=1
rules[1,1,3,1,1]=2
rules[1,1,0,1,1]=2
rules[6,4,3,3,3]=3
rules[3,3,4,3,3]=3
rules[3,3,7,3,3]=7
rules[7,3,3,7,3]=3
rules[5,7,0,0,7]=5
rules[4,3,4,0,5]=4
rules[0,5,7,0,0]=0
rules[7,3,4,3,6]=3
rules[3,4,3,6,3]=3
rules[4,6,3,3,7]=3
rules[6,3,3,7,3]=3
rules[0,7,0,5,0]=0
rules[7,0,5,0,7]=0
rules[7,3,5,0,5]=0
rules[3,5,0,5,6]=0
rules[5,0,5,6,0]=5
rules[3,3,7,3,5]=1
rules[3,6,4,3,4]=4
rules[6,4,3,4,0]=3
rules[4,3,4,0,7]=3
rules[1,7,3,6,3]=3
rules[7,3,6,3,3]=6
rules[3,6,3,3,1]=5
rules[6,3,3,1,1]=0
rules[3,3,1,1,0]=1
rules[3,1,1,0,0]=1
rules[1,0,0,5,7]=4
rules[0,0,5,7,0]=6
rules[0,5,7,0,1]=0
rules[5,7,0,1,3]=7
rules[1,1,0,0,5]=3
rules[0,1,3,7,4]=7
rules[1,3,7,4,3]=3
rules[7,4,3,3,1]=5
rules[4,3,3,1,1]=0
rules[1,1,0,0,6]=3
rules[1,0,0,6,5]=4
rules[0,0,6,5,0]=6
rules[0,6,5,0,1]=0
rules[1,7,3,6,5]=7
rules[3,6,5,0,1]=7
rules[6,5,0,1,1]=7
rules[5,0,1,1,3]=1
rules[0,1,1,3,4]=1
rules[1,1,3,4,6]=7
rules[1,3,4,6,0]=7
rules[6,0,7,1,7]=7
rules[3,4,6,0,7]=1
rules[4,6,0,7,1]=7
rules[6,0,7,1,20]=7
rules[7,1,7,3,6]=7
rules[3,6,3,3,7]=3
rules[3,3,7,3,4]=4
rules[5,0,0,6,5]=5
rules[6,5,0,0,6]=0
rules[4,6,5,0,0] = 4
rules[3,6,3,3,4] = 4
rules[3,4,3,4,6] = 3
rules[5,6,0,7,1] = 0
rules[0,5,6,0,7] = 7
rules[4,0,5,6,0] = 7
rules[4,6,4,3,5] = 7
rules[3,3,4,6,3] = 3
rules[5,6,5,0,0] = 0
rules[6,4,3,3,6] = 3
rules[4,3,3,6,4] = 3
rules[7,3,3,3,7] = 3
rules[3,3,3,7,7] = 5
rules[7,7,0,5,0] = 4
rules[7,0,5,0,5] = 0
rules[3,4,3,7,7] = 5
rules[0,0,5,6,0] = 0
rules[7,0,0,5,6] = 0
rules[7,7,0,0,5] = 4
rules[4,3,4,3,7] = 3
rules[7,3,3,5,1] = 0
rules[0,7,0,7,1] = 0
rules[1,4,0,0,7] = 3
rules[0,0,7,0,7] = 7
rules[4,0,0,7,0] = 7
rules[1,1,4,0,0] = 7
rules[3,7,3,3,5] = 7
rules[3,3,5,1,1]=7
rules[7,0,7,1,1]=7
rules[7,1,1,7,3]=1
rules[0,7,1,1,7]=1
rules[1,1,7,3,7]=7
rules[3,3,3,7,3]=3
rules[3,3,3,7,4]=4
rules[5,7,0,0,5]=5
rules[1,0,5,0,0]=4
rules[0,5,0,0,7]=0
rules[5,0,0,7,0]=5
rules[3,3,4,3,6]=3
rules[4,3,3,3,7]=3
rules[7,0,5,0,0]=0
rules[6,3,3,3,1]=5
rules[3,3,3,1,1]=0
rules[0,0,0,5,7]=0
rules[1,0,0,0,5]=4
rules[1,1,0,0,0]=3
rules[4,3,6,3,3]=6
rules[3,6,3,3,3]=3
rules[1,0,0,5,0]=4
rules[0,0,5,0,7]=0
rules[5,0,7,0,7]=5
rules[7,3,7,3,4]=4
rules[3,4,3,3,1]=5
rules[1,3,6,3,6]=6
rules[3, 6, 3, 6, 3] = 3
rules[3, 6, 3, 5, 0] = 0
rules[6, 3, 5, 0, 1] = 5
rules[3, 5, 0, 1, 1] = 7
rules[6, 3, 6, 3, 5] = 1
rules[6, 0, 5, 7, 1] = 6
rules[0, 6, 0, 5, 7] = 0
rules[1, 1, 3, 4, 0] = 7
rules[1, 3, 4, 0, 6] = 4
rules[3, 4, 0, 6, 5] = 3
rules[4, 0, 6, 0, 5] = 1
rules[3, 4, 0, 6, 0] = 3
rules[4, 0, 6, 5, 0] = 1
rules[6, 5, 0, 7, 1] = 6
rules[0, 6, 5, 0, 7] = 0
rules[1, 7, 3, 4, 6] = 6
rules[7, 3, 4, 6, 3] = 3
rules[3, 4, 6, 3, 5] = 1
rules[4, 6, 3, 5, 0] = 0
rules[3, 6, 3, 1, 0] = 0
rules[3, 1, 0, 6, 0] = 3
rules[6, 3, 1, 0, 5] = 1
rules[4, 3, 1, 0, 6] = 1
rules[0, 5, 7, 1, 1] = 0
rules[7, 1, 1, 7, 4] = 1
rules[1, 1, 7, 4, 3] = 3
rules[5, 7, 1, 1, 7] = 1
rules[6,3,3,3,7]=3
rules[6,3,6,3,3]=6
rules[6,3,3,3,4]=4
rules[6,5,0,0,0]=0
rules[5,0,0,0,5]=5
rules[6,3,6,3,4]=7
rules[4,0,5,0,6]=4
rules[5,0,6,0,5]=7
rules[0,5,0,6,0]=0
rules[4,3,7,3,5]=1
rules[7,0,6,0,1]=6
rules[0,7,0,6,0]=0
rules[4,0,7,0,6]=1
rules[3,4,3,3,6]=3
rules[7,3,3,3,3]=3
rules[3,3,3,3,7]=3
rules[7,0,0,5,0]=0
rules[0,0,5,0,6]=0
rules[7,3,3,3,5]=5
rules[3,3,3,5,1]=0
rules[1,4,0,0,5]=3
rules[4,0,0,5,6]=4
rules[0,0,5,6,5]=0
rules[5,6,5,0,7]=0
rules[7,3,4,6,4]=3
rules[4,6,4,3,3]=3
rules[6,4,3,3,5]=5
rules[4,3,3,5,1]=0
rules[1,4,0,0,0]=3
rules[4,0,0,0,7]=4
rules[0,0,0,7,0]=0
rules[5,0,7,1,1]=0
rules[1,1,7,3,4]=3
rules[0,6,0,1,1]=0
rules[6,0,1,1,3]=1
rules[1,1,3,6,3]=3
rules[0,1,1,3,6]=1
rules[5,0,0,0,7]=5
rules[0,5,0,0,0]=0
rules[5,0,7,0,6]=5
rules[4,6,3,3,3]=3
rules[6,3,3,3,3]=3
rules[3,3,3,3,1]=5
rules[0,0,5,0,5]=0
rules[4,3,4,3,3]=3
rules[0,0,0,5,0]=0
rules[3,3,6,3,3]=6
rules[7,3,3,6,3]=3
rules[6,3,3,5,0]=0
rules[3,3,5,0,1]=5
rules[3,6,3,3,5]=5
rules[3,4,0,0,5]=3
rules[4,0,0,5,7]=4
<table>
<thead>
<tr>
<th>Rule</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>rules[0, 5, 7, 0, 6]</code></td>
<td>0</td>
</tr>
<tr>
<td><code>rules[5, 7, 0, 6, 0]</code></td>
<td>5</td>
</tr>
<tr>
<td><code>rules[3, 6, 3, 7, 4]</code></td>
<td>4</td>
</tr>
<tr>
<td><code>rules[6, 3, 7, 4, 3]</code></td>
<td>3</td>
</tr>
<tr>
<td><code>rules[4, 3, 3, 5, 0]</code></td>
<td>0</td>
</tr>
<tr>
<td><code>rules[7, 4, 3, 3, 5]</code></td>
<td>5</td>
</tr>
<tr>
<td><code>rules[3, 4, 0, 0, 6]</code></td>
<td>3</td>
</tr>
<tr>
<td><code>rules[4, 0, 0, 6, 5]</code></td>
<td>4</td>
</tr>
<tr>
<td><code>rules[0, 6, 5, 0, 6]</code></td>
<td>0</td>
</tr>
<tr>
<td><code>rules[7, 3, 3, 6, 5]</code></td>
<td>5</td>
</tr>
<tr>
<td><code>rules[3, 3, 6, 5, 0]</code></td>
<td>1</td>
</tr>
<tr>
<td><code>rules[3, 6, 5, 0, 5]</code></td>
<td>4</td>
</tr>
<tr>
<td><code>rules[4, 3, 4, 6, 0]</code></td>
<td>5</td>
</tr>
<tr>
<td><code>rules[3, 4, 6, 0, 5]</code></td>
<td>1</td>
</tr>
<tr>
<td><code>rules[4, 6, 0, 5, 6]</code></td>
<td>4</td>
</tr>
<tr>
<td><code>rules[6, 0, 5, 6, 0]</code></td>
<td>0</td>
</tr>
<tr>
<td><code>rules[3, 6, 4, 3, 6]</code></td>
<td>3</td>
</tr>
<tr>
<td><code>rules[6, 4, 3, 6, 5]</code></td>
<td>5</td>
</tr>
<tr>
<td><code>rules[4, 3, 6, 5, 0]</code></td>
<td>1</td>
</tr>
<tr>
<td><code>rules[3, 4, 6, 0, 0]</code></td>
<td>1</td>
</tr>
<tr>
<td><code>rules[4, 6, 0, 0, 7]</code></td>
<td>4</td>
</tr>
<tr>
<td><code>rules[6, 0, 0, 7, 0]</code></td>
<td>0</td>
</tr>
<tr>
<td><code>rules[0, 7, 0, 1, 1]</code></td>
<td>0</td>
</tr>
<tr>
<td><code>rules[1, 1, 3, 7, 3]</code></td>
<td>3</td>
</tr>
<tr>
<td><code>rules[7, 0, 1, 1, 3]</code></td>
<td>1</td>
</tr>
<tr>
<td><code>rules[0, 1, 1, 3, 7]</code></td>
<td>1</td>
</tr>
<tr>
<td><code>rules[3, 3, 3, 3, 4]</code></td>
<td>4</td>
</tr>
</tbody>
</table>
rules[4,0,5,0,0]=4
rules[0,5,0,0,6]=0
rules[0,5,6,0,0]=7
rules[5,6,0,0,7]=0
rules[4,3,3,3,3]=3
rules[0,0,5,0,0]=0
rules[3,3,3,3,5]=5
rules[4,0,0,5,0]=4
rules[3,4,3,3,5]=5
rules[4,0,0,0,5]=4
rules[0,0,0,5,6]=0
rules[7,3,3,5,0]=0
rules[3,3,5,0,7]=5
rules[7,3,4,0,0]=4
rules[3,4,0,0,7]=3
rules[0,0,7,0,5]=7
rules[0,7,0,5,6]=0
rules[6,4,3,7,3]=3
rules[4,3,7,3,3]=7
rules[0,7,0,0,7]=0
rules[0,0,7,0,0]=7
rules[5,6,0,1,1]=0
rules[1,1,3,6,4]=3
rules[3,3,3,3,3]=3
rules[6,3,3,3,5]=5
rules[3,3,3,5,0]=0
rules[4,6,3,3,6]=3
rules[6,3,3,6,3]=3
rules[3,3,6,3,5]=1
rules[6,3,5,0,5]=5
rules[4,3,4,0,6]=4
rules[0,6,0,5,0]=0
rules[6,0,5,0,7]=0
rules[0,6,5,0,0]=0
rules[6,5,0,0,7]=0
rules[7,3,3,4,6]=3
rules[6,3,3,1,0]=0
rules[3,3,1,0,5]=1
rules[4,3,1,0,0]=1
rules[3,1,0,0,5]=3
rules[4,3,3,1,0]=0
rules[3,1,0,0,6]=3
rules[3,3,6,3,4]=7
rules[3,6,3,3,6]=3
rules[0,7,0,0,6]=0
rules[7,0,0,6,5]=0
rules[4,0,7,0,0]=1
rules[4,6,3,3,1]=5
rules[1,0,0,6,0]=4
rules[0,0,6,0,7]=6
rules[0,6,0,7,1]=0
rules[0,7,0,0,5]=0
rules[7,0,0,5,7]=0
rules[7,4,3,3,7]=3
<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>rules[4,3,3,7,3]</td>
<td>3</td>
</tr>
<tr>
<td>rules[7,0,0,6,0]</td>
<td>0</td>
</tr>
<tr>
<td>rules[5,7,0,1,1]</td>
<td>7</td>
</tr>
<tr>
<td>rules[1,1,3,7,4]</td>
<td>7</td>
</tr>
<tr>
<td>rules[6,0,7,1,1]</td>
<td>7</td>
</tr>
<tr>
<td>rules[1,1,7,3,6]</td>
<td>7</td>
</tr>
<tr>
<td>rules[5,0,7,0,0]</td>
<td>5</td>
</tr>
<tr>
<td>rules[5,0,0,6,0]</td>
<td>5</td>
</tr>
<tr>
<td>rules[4,3,3,6,3]</td>
<td>3</td>
</tr>
<tr>
<td>rules[3,3,5,0,5]</td>
<td>5</td>
</tr>
<tr>
<td>rules[7,3,3,3,6]</td>
<td>3</td>
</tr>
<tr>
<td>rules[3,3,3,6,5]</td>
<td>5</td>
</tr>
<tr>
<td>rules[5,0,6,0,7]</td>
<td>7</td>
</tr>
<tr>
<td>rules[4,6,0,5,0]</td>
<td>4</td>
</tr>
<tr>
<td>rules[6,0,5,0,6]</td>
<td>0</td>
</tr>
<tr>
<td>rules[3,4,3,6,5]</td>
<td>5</td>
</tr>
<tr>
<td>rules[7,3,6,3,4]</td>
<td>7</td>
</tr>
<tr>
<td>rules[6,3,4,3,6]</td>
<td>3</td>
</tr>
<tr>
<td>rules[4,6,0,0,5]</td>
<td>4</td>
</tr>
<tr>
<td>rules[6,0,0,5,6]</td>
<td>0</td>
</tr>
<tr>
<td>rules[3,3,5,1,4]</td>
<td>7</td>
</tr>
<tr>
<td>rules[5,1,4,0,0]</td>
<td>7</td>
</tr>
<tr>
<td>rules[3,3,3,6,3]</td>
<td>3</td>
</tr>
<tr>
<td>rules[3,3,3,6,4]</td>
<td>3</td>
</tr>
<tr>
<td>rules[5,6,0,0,5]</td>
<td>0</td>
</tr>
<tr>
<td>rules[4,3,3,7,7]</td>
<td>5</td>
</tr>
<tr>
<td>rules[7,7,0,0,0]</td>
<td>4</td>
</tr>
</tbody>
</table>
rules[7,0,0,0,7]=0
rules[7,0,5,0,6]=0
rules[6,3,4,3,7]=3
rules[0,7,0,0,0]=0
rules[5,7,0,0,0]=5
rules[4,3,3,3,6]=3
rules[6,0,5,0,0]=0
rules[3,3,3,1,0]=0
rules[3,1,0,0,0]=3
rules[6,3,3,3,6]=3
rules[7,0,0,0,5]=0
rules[4,3,3,3,1]=5
rules[1,0,0,0,6]=4
rules[0,0,0,6,0]=0
rules[0,0,6,0,5]=6
rules[4,3,6,3,5]=1
rules[4,0,6,0,6]=1
rules[6,0,6,0,1]=6
rules[0,6,0,6,0]=0
rules[7,0,0,0,6]=0
rules[0,0,0,6,5]=0
rules[0,0,6,0,6]=6
rules[5,0,0,0,6]=5
rules[5,0,6,0,6]=7
rules[3,3,3,3,6]=3
rules[6,0,0,5,0]=0
rules[0,5,6,0,6]=7
<table>
<thead>
<tr>
<th>rules</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>[5,6,0,6,0]</td>
<td>0</td>
</tr>
<tr>
<td>[6,3,6,4,3]</td>
<td>7</td>
</tr>
<tr>
<td>[3,6,3,6,4]</td>
<td>3</td>
</tr>
<tr>
<td>[7,0,0,0,0]</td>
<td>0</td>
</tr>
<tr>
<td>[0,0,0,0,7]</td>
<td>0</td>
</tr>
<tr>
<td>[0,0,0,0,6]</td>
<td>0</td>
</tr>
<tr>
<td>[4,6,3,3,5]</td>
<td>5</td>
</tr>
<tr>
<td>[4,0,0,6,0]</td>
<td>4</td>
</tr>
<tr>
<td>[0,0,6,0,0]</td>
<td>6</td>
</tr>
<tr>
<td>[0,6,0,0,7]</td>
<td>0</td>
</tr>
<tr>
<td>[0,0,0,0,5]</td>
<td>0</td>
</tr>
<tr>
<td>[0,6,0,5,6]</td>
<td>0</td>
</tr>
<tr>
<td>[6,4,3,6,3]</td>
<td>3</td>
</tr>
<tr>
<td>[5,0,0,0,0]</td>
<td>5</td>
</tr>
<tr>
<td>[4,3,3,3,5]</td>
<td>5</td>
</tr>
<tr>
<td>[4,0,0,0,6]</td>
<td>4</td>
</tr>
<tr>
<td>[4,0,6,0,0]</td>
<td>1</td>
</tr>
<tr>
<td>[0,6,0,0,6]</td>
<td>0</td>
</tr>
<tr>
<td>[6,0,0,6,5]</td>
<td>0</td>
</tr>
<tr>
<td>[0,6,0,0,5]</td>
<td>0</td>
</tr>
<tr>
<td>[6,0,0,5,7]</td>
<td>0</td>
</tr>
<tr>
<td>[7,4,3,3,6]</td>
<td>3</td>
</tr>
<tr>
<td>[6,0,0,6,0]</td>
<td>0</td>
</tr>
<tr>
<td>[5,0,6,0,0]</td>
<td>7</td>
</tr>
<tr>
<td>[5,7,0,0,6]</td>
<td>5</td>
</tr>
<tr>
<td>[6,3,3,7,4]</td>
<td>4</td>
</tr>
<tr>
<td>[4,3,3,6,5]</td>
<td>5</td>
</tr>
</tbody>
</table>
rules[4,6,0,0,0]=4
rules[6,0,0,0,7]=0
rules[0,6,0,0,0]=0
rules[5,6,0,0,0]=0
rules[6,0,0,0,5]=0
rules[6,0,0,0,6]=0
rules[6,0,0,0,0]=0
rules[5,6,0,0,6]=0
rules[6,3,3,6,4]=3

#even rules here
rules[20,20,1,0,20]=2
rules[20,1,0,20,20]=2
rules[1,0,0,0,20]=4
rules[1,3,4,0,20]=7
rules[3,4,0,20,20]=1
rules[4,0,0,0,20]=4
rules[4,3,4,0,20]=5
rules[7,3,5,1,20]=0
rules[3,5,1,20,20]=1
rules[4,3,5,1,20]=7
rules[5,0,7,1,20]=0
rules[5,6,0,1,20]=0
rules[5,7,0,1,20]=7

#Loop through the squad and update with new state

#count the number of firings
#create temporary array to hold update

temp = np.zeros(squad_size+4, dtype=np.int)

count=0
time=0
A=rules

while count==0:
    for i in range(2, squad_size+2):
        if squad[i]==21:
            print("Undefined rule at time ", time-1)

    for i in range(2, squad_size+2):
        farLeft=squad[i-2]
        nearLeft=squad[i-1]
        current=squad[i]
        nearRight=squad[i+1]
        farRight=squad[i+2]
        temp[i]=A[farLeft,nearLeft,current,nearRight,farRight]

    for j in range(2, squad_size+2):
        squad[j]=temp[j]

    for k in range(2, squad_size+2):
        if squad[k] == 0:
            sys.stdout.write('%s\u2610' %(fg('white')))
        elif squad[k] == 1:
            sys.stdout.write('%s\u2610' %(fg('green')))
        elif squad[k] == 2:
            sys.stdout.write('%s\u2610' %(fg('red')))
        elif squad[k] == 3:
            sys.stdout.write('%s\u2610' %(fg('yellow')))

95
elif squad[k] == 4:
    sys.stdout.write('%s\u2610' %(fg('magenta')))
elif squad[k] == 5:
    sys.stdout.write('%s\u2610' %(fg('blue')))
elif squad[k] == 6:
    sys.stdout.write('%s\u2610' %(fg('cyan')))
elif squad[k] == 7:
    sys.stdout.write('%s\u2610' %(fg('orchid')))
else:
    sys.stdout.write('"

for l in range(0, squad_size+4):
    if squad[l]==2:
        count=count+1
        time=time+1

if not(count==squad_size):
    print("Error in firing squad algorithm occurs at time ", time)
else:
    print("Firing squad algorithm complete in ", time, " time-steps.")
REFERENCES


Curriculum Vitae

Kathryn A. Boddie

Education:

Master of Science, Pure Mathematics
Florida State University, Tallahassee, FL, May 2011

Bachelor of Science, Mathematics
University of Wisconsin-Madison, Madison, WI, May 2009

Honors and Awards:

Ernst Schwandt Teaching Award
May 2016

Professional Memberships:

American Mathematical Society
September 2013 – Present

Mathematical Association of America
September 2013 – Present

Association for Women in Mathematics
January 2018 – Present

Research Interests:

Applied Mathematics

- Cellular automata models, Firing Squad Synchronization Problem, computer science, discrete mathematics, optimization, computational mathematics.

Algebra
- Commutative algebra, ring theory, localization, zero-divisor graphs, algebraic number theory.

Graph Theory and Combinatorics
- Zero-divisor graphs, nilradical and non-nilradical graphs, path minimization, coloring problems, game theory.

Publications (some publications under the name Kathryn A. Lokken):


Conferences and Presentations:


**Recent Service:**

AWM student chapter outreach at Oostburg Christian School February 2019

UW-Milwaukee AWM student chapter president January 2018–Present

**Programming/Scripting Languages**

- Java
- C++
- Python
- Julia
- Matlab

**Teaching Experience:**

*Teaching Assistant* September 2013 –Present

University of Wisconsin-Milwaukee, Milwaukee, WI

Sole instructor without a course coordinator for the following courses:

- Math 116 –College Algebra as a pre-requisite for Calculus 1
- Math 205 –Finite Mathematics
- Math 211 –Survey in Calculus and Analytical Geometry
- Math 234 –Linear Algebra and Differential Equations

Sole instructor under a course coordinator for the following courses:
• Math 094 – Foundations of Elementary Mathematics – developmental course using a flipped classroom model

• Math 092/102 – Mathematical Literacy for College Students I and II – developmental and college level math courses using a co-requisite flipped classroom model for non-STEM majors

• Math 098 – Algebraic Literacy I – developmental course

• Math 108 – Algebraic Literacy II – college level algebra course (Math 098/108 together are the equivalent of Math 105 – Introduction to College Algebra)

• Math 105 – Introduction to College Algebra as a pre-requisite for Business Calculus

• Math 231 – Calculus and Analytic Geometry I

• Math 232 – Calculus and Analytic Geometry II

Discussion section instructor for:

• Math 211 – Survey in Calculus and Analytical Geometry

Mathematics Teacher
March 2012 – May 2013

Amos P. Godby High School

• Three classes of Honors Geometry

• Four classes of Algebra 1 (one honors)

• One class of Algebra 2

Teaching Assistant
August 2009 – May 2011

Florida State University

Lab Proctor for the following courses

• MAC 1105 – College Algebra

• MGF 1107 – Practical Finite Math