INTEGERS: ELECTRONIC JOURNAL OF COMBINATORIAL NUMBER THEORY 6 (2006), #G01

MATHEMATICS OF GHALY'S MACHINE

David Joyner
Department of Mathematics, US Naval Academy, Annapolis, MD, USA
wdj@usna.edu

Received: 5/20/05, Revised: 1/20/06, Accepted: 1/21/06, Published: 2/1/06

Abstract

This paper mathematically models and then analyzes an electronic device (referred to as Ghaly's machine in the title) described in the 1994 Patent 5,286,037.

0. Introduction

This paper sketches some mathematical models of an electronic device (referred to as Ghaly's machine in the title) described in the 1994 Patent 5,286,037. We show, among other things, the following facts.

- The game begins with a "random" initial assignments of codes (as the game is described in the patent in its easiest mode), displayed as an initial color configuration of the 4×4 array of buttons. There are $4!^4 = 331776$ possible initial assignments, though it is possible for different initial assignments to have the same initial display of colored buttons.
- Given a random initial configuration of colors (including off) on the 4×4 array, the probability that it arises from an initial assignment of codes (with no buttons pressed) as in Ghaly's patent is no more than $2.17... \times 10^{-6}$.
- In a sense to be made precise below, "most" button presses cannot correspond to a solution of any initial assignment of codes (as the game is described in the patent in its easiest mode).

1. Some history

1.1. Ghaly's machine

In 1994, the U.S. Patent Office granted Nabil Ghaly patent 5,286,037 for an electrical device using "state of the art" circuitry. In its most basic form, this device is a 4×4 array of buttons which, when pressed, can display any one of 4 colors (or "off", which could be regarded as a fifth color though we shall not do so here). We shall call this device (and its general version, discussed below) **Ghaly's machine**. The goal is to take a "random" starting position, where each of the buttons on a 4×4 array can be one of the four colors or off, and press buttons until all the buttons are the same color[1].

The rule for how the buttons of Ghaly's machine changes colors is left to another section.

1.2. Merlin's machine

What is necessary to point out here, regarding the history of events, is that Ghaly's game was patented at least 12 years *after* Parker Brothers marketed another electronic device called "Merlin's Magic" (the rules book was published copyright 1979). This device was a marketed as a square array of buttons which could turn off or on. We shall call this device (and some of its generalizations [2], discussed below) **Merlin's machine**. Here the rule for how the buttons of change's machine toggles on/off is very simple: pressing any button toggles the certain neighboring buttons (see Example 4 below and the survey in [J]).

2. Finite state machines

This section introduces a precise language which is helpful for comparing two different electrical devices, such as, for example, Ghaly's machine and Merlin's Magic.

A **finite state machine** (or **Mealy machine**) is a 5-tuple (S, I, O, f, g) where S is a finite set of objects called **states**, I is a finite set of symbols called the **input alphabet**, O is a finite set of symbols called the **output alphabet**, $f : S \times I \to S$ is the **transition** or **next state** function, and $g : S \times I \to O$ is the **output** function. (One reference for this is, for example, Grimaldi [Gr].)

Remark 1. *The states of the machine can be roughly regarded as the "memory registers" of the machine. The input alphabet be be roughly regarded as a labeling of the buttons on might press (or moves you make, depending on the constitution of the machine).*

[1]The rules of the patent do not seem to restrict to this case, so we shall allow several buttons to be pressed simultaneously, if desired, in our mathematical models below.

[2]Tiger Electronics (a subsidiary of Hasbro) later marketed several generalizations under the name **Lights Out**.

Remark 2. *More precisely, f and/or g need not be defined on all of $S \times I$. One or both may only be defined on some subset of $S \times I$ (in other words, f and g need only be what is called a* **partial function***). When both f and g are functions (each defined on all of $S \times I$) then the machine is called* **complete***.*

In the theory of finite state machines, one studies morphisms between machines, submachines, components of a state of a machine, and experiments to determine whether or not machines are isomorphic. For more details on these topics, we refer the reader to Conway [C].

2.1. Examples of machines

Example 3. (Rubik's cube) A **Rubik's cube** is a cube, which has been sliced into 27 subcubes of equal sizes, each of which has 6 smaller faces parallel to the 6 faces of the larger cube. In the "solved state" each face of the larger cube is colored a different (solid) color. To be concrete, suppose we have fixed such a cube in space (say on a table top in front of you, the reader). The 9 facets of the front face shall be colored yellow, the 9 facets of the back face shall be colored green, the 9 facets of the up (top) face shall be colored red, the 9 facets of the down (bottom) face shall be colored orange, the 9 facets of the right face shall be colored blue, the 9 facets of the left face shall be colored white. There are $9 \cdot 6 = 54$ colored facets total. Each of the 6 faces has 3 parallel slices of 9 subcubes, each of which may be independently rotated at angles of 90^o, 180^o, and 270^o.

With the yellow-colored center facet facing front, and the red-colored center facet facing up, let R denote the 90^o clock-wise rotation (as you face that side) of the right slice, let L denote the 90^o clock-wise rotation of the left slice, let F denote the 90^o clock-wise rotation of the front slice, let B denote the 90^o clock-wise rotation of the back slice, let D denote the 90^o clock-wise rotation of the down slice, and let U denote the 90^o clock-wise rotation of the up slice. Note that each of these rotations leaves the center facets fixed. There are $9 \cdot 6 - 6 = 48$ colored non-central facets total.

Let S denote the set of all possible permutations of the 48 colored facets on the cube (leaving the center facets fixed). Let $I = \{R, L, F, B, U, D\}$. Let $O = S$. Let the transition function $f : S \times I \to S$ be the function sending (s, i) ($s \in S$ and $i \in I$) which takes the state s (the cube with 48 non-central facets scrambled according to some permutation) and the input i (one of the moves R, L, ..., U) to the new state s' obtained by performing the move i on the scrambled cube s: $f(s, i) = s'$. Let the output function g be the same as f.

This describes the Rubik's cube as a finite state machine.

Define $E_{i,j}$ to be the $N \times N$ matrix all of whose entries are 0 *except* for the (i, j)-th entry which is equal to 1. An **elementary matrix** is an $N \times N$ matrix which is equal to $E_{i,j}$ for some i and j with $1 \leq i \leq N$, $1 \leq j \leq N$. Clearly, $E_{i,j} \in M_N(\mathbf{F}_2)$. (Recall $M_N(\mathbf{F}_2)$ denotes

the set of all $N \times N$ matrices whose entries are either equal to 0 or equal to 1.) We can, if we want, add to elements of $M_N(\mathbf{F}_2)$ (adding each corresponding matrix element mod 2) and get another matrix in $M_N(\mathbf{F}_2)$.

Example 4. (Merlin's machine) In this example, $N > 2$ be any integer. Let

$$S = M_N(\mathbf{F}_2), \qquad I = \{E_{i,j} \mid 1 \le i \le N, 1 \le j \le N\}, \qquad O = M_N(\mathbf{F}_2).$$

We interprete the input $E_{i,j}$ to mean that the $(i,j) - th$ button of the array was pressed. The effect of pressing any button it to toggle the button itself and those which are directly north, south, east, or west of the button pressed. There is no wrap-around (at least not in the most basic version) so there are

- 3 buttons toggled, in case a corner button is pressed,
- 4 buttons toggled, in case a edge button is pressed,
- 5 buttons toggled, in case a central button is pressed.

To model this as a finite state machine, let the transition function $f : S \times I \to S$ be defined by

$$f(s, E) = s + t(E), \qquad s \in S,\ E \in I,$$

where addition is coordinate-wise mod 2 and $t(E)$ is the **toggle matrix**:

$$t(E_{i,j}) = \begin{cases} E_{i,j} + E_{i-1,j} + E_{i,j-1} + E_{i+1,j} + E_{i,j+1}, & 1 < i < N,\ 1 < j < N, \\ E_{i,j} + E_{i-1,j} + E_{i+1,j} + E_{i,j+1}, & 1 < i < N,\ j = 1, \\ E_{i,j} + E_{i+1,j} + E_{i,j+1}, & i = 1,\ j = 1, \\ E_{i,j} + E_{i-1,j} + E_{i,j+1}, & i = N,\ j = 1, \\ E_{i,j} + E_{i,j-1} + E_{i+1,j} + E_{i,j+1}, & i = 1,\ 1 < j < N, \\ E_{i,j} + E_{i-1,j} + E_{i,j-1} + E_{i,j+1}, & i = N,\ 1 < j < N, \\ E_{i,j} + E_{i-1,j} + E_{i,j-1} + E_{i+1,j}, & 1 < i < N,\ j = N, \\ E_{i,j} + E_{i,j-1} + E_{i+1,j}, & i = 1,\ j = N, \\ E_{i,j} + E_{i-1,j} + E_{i,j-1}, & i = N,\ j = N. \end{cases}$$

For example, if $N = 3$ and $E = E_{1,2}$ then

$$t(E) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

and if $E = E_{2,2}$ then

$$t(E) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

Finally, let the output function $g : S \times I \to O$ be the same as the transition function: $g = f$.

This describes Merlin's Magic/Lights Out as a finite state machine.

Let R_{N+1} denote the set $\{0, 1, ..., N\}$ and let $M_N(R_{N+1})$ denote the set of $N \times N$ matrices with coordinates in R_{N+1}. Both R_{N+1} and $M_N(R_{N+1})$ are closed under coordinate-wise addition mod N. If X is a finite set, let S_X denote the symmetric group of permutations of X. As usual, if $X = \{1, ..., n\}$ then write $S_n = S_X$.

We shall fix a correspondence between the set of N colors in the machines multi-color display and the elements of R_{N+1}.

Example 5. (Ghaly's machine) Let $N \geq 4$ be a power of 2. Let $\sigma \in S_I \times S_J \subset S_{2N}$ and $\tau \in S_I \times S_J \subset S_{2N}$ be choosen arbitrarily ("at random" in the terminology of [Gh]) but fixed for the remainder of this example. Different choices of (σ, τ) will (in general) yield different machines.

Let

$$S = M_N(\mathbf{F}_2), \qquad I = \{E_{i,j} \mid 1 \leq i \leq N, 1 \leq j \leq N\}, \qquad O = M_N(R_{N+1}).$$

Let the transition function $f : S \times I \to S$ be defined by

$$f(s, E) = s + E, \qquad s \in S,\ E \in I,$$

where addition is coordinate-wise mod 2. Let the output function $f : S \times I \to O$ be the $N \times N$ color codes matrix $(c_{ij}) \in M_N(R_{N+1})$, as defined in Section 3.6 below, computed using

- σ, τ,
- the wiring diagram for $f(s, E)$, and
- the rules for color assignments.

Since σ, τ are fixed, the present state of the machine, namely the wiring matrix $s \in S$, completely determines the coloring of the buttons, which is represented by an element $o \in O$. The effect of pressing the $(i, j)^{th}$ button is represented by the value of the output function $o' = f(s, E_{i,j})$. We may, if we wish to reformulate this to look a little more similar to the above example of Merlin's machine, also represent this as follows: $o' = o + t(s, E_{i,j})$. In other words, the output function is "additive" as in the case of Merlin's machine, but it required addition by an array which depends on both the value of the input variable and the state variable.

This describes Ghaly's machine as a finite state machine.

3. Rules of the game

"We shall now give a brief summary of the beginnings of the Glass Bead Game." **Magister Ludi**, *Hermann Hesse*

In this section, the operation of Ghaly's machine (at least, on the simplest of the four levels of difficulty) shall be described.

3.1. Notation

Most of the notation is taken from the section "Mathematical description of the logic problem" starting on column 10 of Ghaly's patent [Gh].

N	a power of 2, at least 4 (N = the array dimension = the number of colors)
n	$\log_2(N)+1$, so $2^n = 2N$
$\mathbf{F}_2$	Boolean field $\{0,1\}$
D	$\mathbf{F}_2^n$, the set of all n-tuples of 0's and 1's
d_i, $1 \le i \le 2N$	i^{th} element of D - labeled so d_i corresponds to the binary representation of $i-1$
X_i, $1 \le i \le 2N$	labels for "transmitter codes" (the $X_i \in D$ are pairwise distinct, see (2))
CG_i, $1 \le i \le 2N$	labels for "color generators codes" (the $CG_i \in D$ are pairwise distinct, see (3))
C_i, $1 \le i \le 2N$	"color codes" (the $CG_i \in D$ are not necessarily distinct, see (4))
$\oplus$	*logical exclusive or* (addition mod 2, where 0 is "false" and 1 is "true")
$\odot$	*logical inclusive or*
$B : D \times D \to D$	"Boolean function" defined by $B((x_1,x_2,...,x_n),(y_1,y_2,...,y_n)) = (x_1 \odot y_1, x_2 \oplus y_2, ..., x_n \oplus y_n)$
$\mathbf{Z}_m$	$\{1,2,...,m\}$
S_m	the set of all 1-1, onto functions from $\mathbf{Z}_m$ to itself
$M_N(\mathbf{F}_2)$	set of all $N \times N$ matrices of 0's and 1's

The elements of the set S_m are called **permutations** of $\mathbf{Z}_m$.

The table of values of the binary operations $\oplus$ and $\odot$ are given below (0 for "false", 1 for "true").

x	y	$x \odot y$	$x \oplus y$
1	1	1	0
1	0	1	1
0	1	1	1
0	0	0	0

For the definition of B, $\odot$ and $\oplus$, see also Fig. 20 (flow chart), Fig. 23 (color assignments for 4×4), Fig. 24 (color assignments for 8×8), and column 6 (not all of which are consistent) of [Gh]. (Figures 23 and 24 of [Gh] contain, it turns out, calculations inconsistent with the description in column 6 of [Gh]. We shall assume here *that Figure 20 and column 6, lines 35-45 are correct*, where "inclusive or" refers to the function $\odot$ above.)

Here is the table of values of Ghaly's "Boolean function" B:

B	[0,0,0]	[0,0,1]	[0,1,0]	[0,1,1]	[1,0,0]	[1,0,1]	[1,1,0]	[1,1,1]
[0,0,0]	[0,0,0]	[0,0,1]	[0,1,0]	[0,1,1]	[1,0,0]	[1,0,1]	[1,1,0]	[1,1,1]
[0,0,1]	[0,0,1]	[0,0,0]	[0,1,1]	[0,1,0]	[1,0,1]	[1,0,0]	[1,1,1]	[1,1,0]
[0,1,0]	[0,1,0]	[0,1,1]	[0,0,0]	[0,0,1]	[1,1,0]	[1,1,1]	[1,0,0]	[1,0,1]
[0,1,1]	[0,1,1]	[0,1,0]	[0,0,1]	[0,0,0]	[1,1,1]	[1,1,0]	[1,0,1]	[1,0,0]
[1,0,0]	[1,0,0]	[1,0,1]	[1,1,0]	[1,1,1]	[1,0,0]	[1,0,1]	[1,1,0]	[1,1,1]
[1,0,1]	[1,0,1]	[1,0,0]	[1,1,1]	[1,1,0]	[1,0,1]	[1,0,0]	[1,1,1]	[1,1,0]
[1,1,0]	[1,1,0]	[1,1,1]	[1,0,0]	[1,0,1]	[1,1,0]	[1,1,1]	[1,0,0]	[1,0,1]
[1,1,1]	[1,1,1]	[1,1,0]	[1,0,1]	[1,0,0]	[1,1,1]	[1,1,0]	[1,0,1]	[1,0,0]

In general,

$$B((x_1, x_2, ..., x_n), (y_1, y_2, ..., y_n)) = (x_1 \odot y_1, x_2 \oplus y_2, ..., x_n \oplus y_n),$$

for $(x_1, x_2, ..., x_n), (y_1, y_2, ..., y_n) \in D$. (This is according to Ghaly's patent, column 6 line 36 and column 7 line 48, where both indicate that the $\odot$ is to be applied to the "left" bit. Column 6 calls it the "third" bit and column 7 calls it the "first" bit. *It will be assumed that "left" is correct and one of the others is a typographical error.*)

We make the following color code assignments:

$[1, 0, 0]$	color 1
$[1, 0, 1]$	color 2
$[1, 1, 0]$	color 3
$[1, 1, 1]$	color 4
$[0, *, *]$	off ,

where color 1, ..., color 4 are four distinct colors. For example, one might take color 1 to be red, color 2 to be yellow, color 3 to be green, and color 4 to be blue.

3.2. The graph

Let $N \geq 2$ be an integer. (Later we shall see how Ghaly's patent forces N to be a power of 2.) Subdivide a square in the plane into N^2 equal subsquares whose edges are parallel to the

edges of the larger square. Each of these smaller subsquares represents a button in Ghaly's machine which may display one of N colors or off. We shall label the N^2 buttons on this $N \times N$ array of squares $(Q_{i,j})$ of as though they were entries of a matrix. Thus, for example, Q_{11} is in the upper left-hand corner, Q_{1N} is in the upper right-hand corner, Q_{1N} is in the lower left-hand corner, and Q_{NN} is in the lower right-hand corner.

Remark 6. *There are* $(N+1)^{N^2}$ *possible configurations of this array of colored buttons. For example, if* $N = 4$ *there are*

$$5^{16} = 152587890625 = (1.5...) \times 10^{11},$$

or over 150 *billion, possible configurations* [3].

Let Γ denote the graph whose vertices are the $(N+1)^2$ vertices of these squares and whose $2N(N+1)$ edges are the edges of these subsquares.

3.3. The wiring/color matrix, codes and diagrams

The wiring matrix: When the button associated to the square $Q_{i,j}$ has been pressed, we shall indicate this by putting $W_{i,j} = 1$. By the rules of the game [Gh], if a button is pressed twice in a row that is the same as not pressing it at all. A button which has not been pressed shall be indicated by putting $W_{i,j} = 0$. The array $W = (W_{i,j})_{1 \le i,j \le N}$ will be called the **wiring matrix**. We shall see how the wiring matrix affects the colors of the buttons below.

3.3.1. Transmitter codes and diagrams

The transmitter codes: The left edges of the left-most squares (top-to-bottom) $Q_{1,1}$, $Q_{2,1}$, ..., $Q_{N,1}$ will be labeled with the "weights" X_1, X_2, ..., X_N, resp.. The bottom edges of the bottom-most squares (left-to-right) $Q_{N,1}$, $Q_{N,2}$, ..., $Q_{N,N}$ will be labeled with the "weights" X_{N+1}, X_{N+2}, ..., X_{2N}, resp.. These labels/weights will be called the **transmitter codes**. According to Ghaly's patent [Gh], the weights assigned to these X_i are to be all the distinct elements of $\mathbf{F}_2^n$. *This forces* N *to be a power of* 2. *We shall assume this from this point on.*

There is a constraint on the order to the assignment of the

$$\mathbf{F}_2^n = \{d_1, ..., d_{2N}\}$$

to the $\{X_1, ..., X_{2N}\}$ assumed in [Gh], which we now describe. For convenience of notation, let

$$I = \{1, 2, ..., N\}, \quad J = \{N+1, N+2, ..., 2N\}. \tag{1}$$

[3] We will see in a later section that according to the patent description of the game only 40320 of these possibilities can actually arise from the wiring constraints of the game.

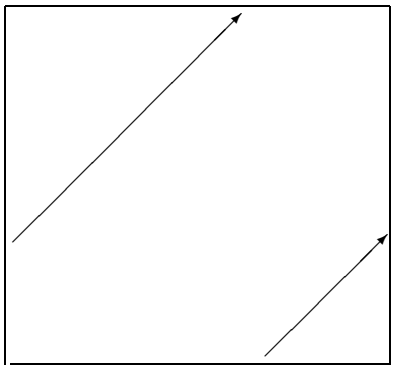
Let $\sigma_1 \in S_I$ be a permutation of I and $\sigma_2 \in S_I$ be a permutation of J. This gives rise to an assignment as follows:

$$d_i = X_{\sigma_1(i)}, \quad 1 \le i \le N, \qquad d_i = X_{\sigma_2(i)}, \quad N+1 \le i \le 2N. \tag{2}$$

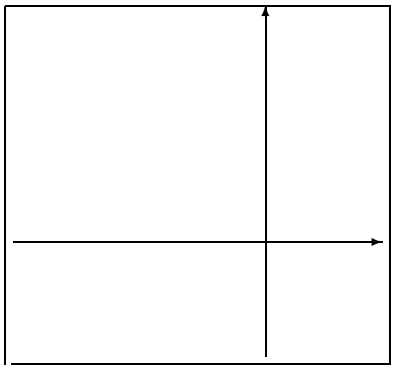
(Here $\sigma(i)$ denotes the effect which the permutation σ has on i.) Conversely, any such assignment of the elements of $\mathbf{F}_2^n$ to the $X_1, ..., X_{2N}$ will yield such a permutation

$$\sigma = (\sigma_1, \sigma_2) \in S_I \times S_J \subset S_{2N}.$$

In the remainder of this section, we shall see how the state of the machine affects the "wiring diagram" (defined below) which routes transmitter codes from the left and bottom edges to the top and right edges.



Transmission wiring diagram on Q_{ij} when $W_{ij} = 0$.



Transmission wiring diagram on Q_{ij} when $W_{ij} = 1$.

Example 7. Examples of transmitter wiring diagrams for Ghaly's machine are given in Figures 1, 2, and 3 using $E_{1,1}$, $E_{1,2}$, and $E_{3,1}$.

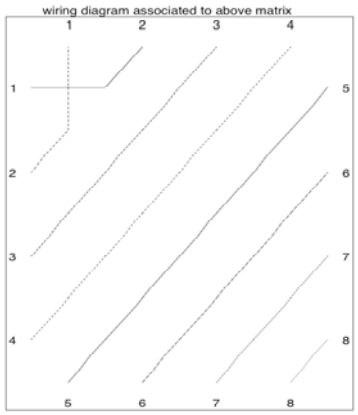


Figure 1: Transmitter wiring diagram for $W = E_{1,1}$.

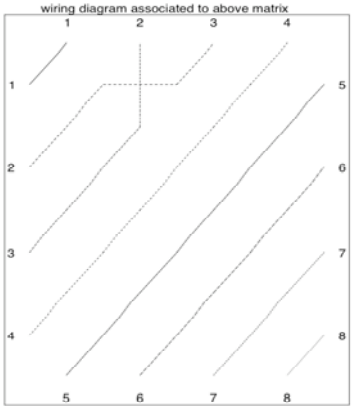


Figure 2: Transmitter wiring diagram for $W = E_{1,2}$.

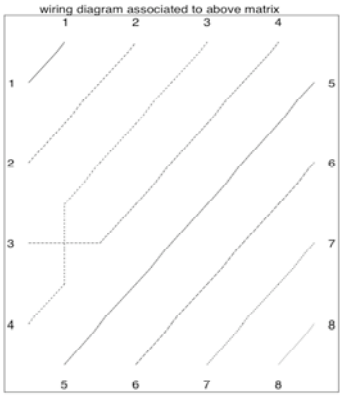


Figure 3: Transmitter wiring diagram for $W = E_{3,1}$.

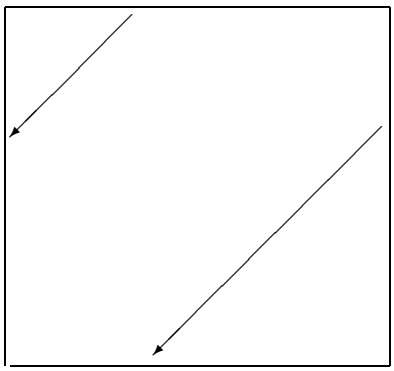
3.3.2. Color generator codes and diagrams

The color generator codes: The top edges of the upper-most squares (left-to-right) $Q_{1,1}$, $Q_{1,2}$, ..., $Q_{1,N}$ will be labeled with the "weights" CG_1, CG_2, ..., CG_N, resp.. The right edges of the right-most squares (top-to-bottom) $Q_{1,N}$, $Q_{2,N}$, ..., $Q_{N,N}$ will be labeled with the "weights" CG_{N+1}, CG_{N+2}, ..., CG_{2N}, resp.. These labels/weights will be called the **color generator codes**. According to Ghaly's patent [Gh], the weights assigned to these CG_i are to be all the distinct elements of $\mathbf{F}_2^n$. Let $\tau = (\tau_1, \tau_2) \in S_I \times S_J$ be a permutation, where I, J are as in (1). This gives rise to an assignment as follows:

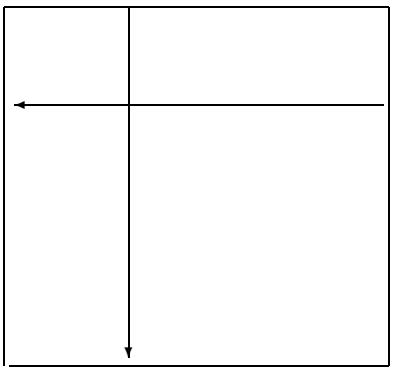
$$d_i = X_{\tau_1(i)}, \quad 1 \le i \le N, \qquad d_i = X_{\tau_2(i)}, \quad N+1 \le i \le 2N. \tag{3}$$

Conversely, any such assignment of the elements of $\mathbf{F}_2^n$ to the $CG_1, ..., CG_{2N}$ will yield such a permutation $\tau = (\tau_1, \tau_2) \in S_I \times S_J \subset S_{2N}$.

In the remainder of this section, we shall see how the state of the machine affects the "coloring diagram" (defined below) which routes color generator codes from the top and right edges to the individual squares.



Color generator wiring diagram on Q_{ij} when $W_{ij} = 1$.



Color generator wiring diagram on Q_{ij} when $W_{ij} = 0$.

Example 8. Examples of color generator wiring diagrams for Ghaly's machine are given in Figures 4, 5, and 6 using $E_{1,1}$, $E_{1,2}$, and $E_{3,1}$.

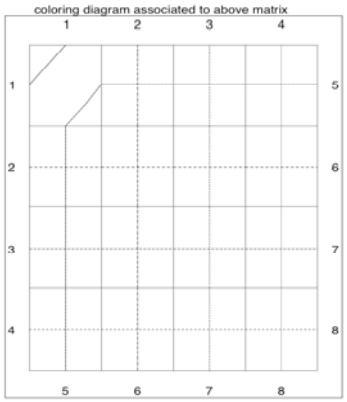


Figure 4: Color generator wiring diagram for $W = E_{1,1}$.

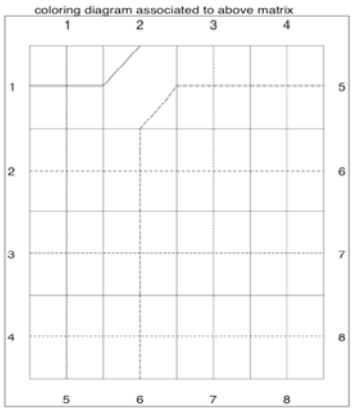


Figure 5: Color generator wiring diagram for $W = E_{1,2}$.

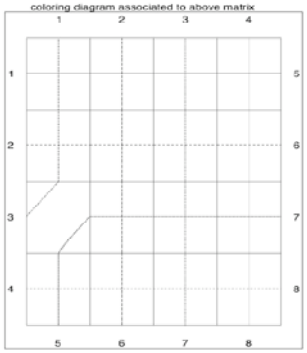
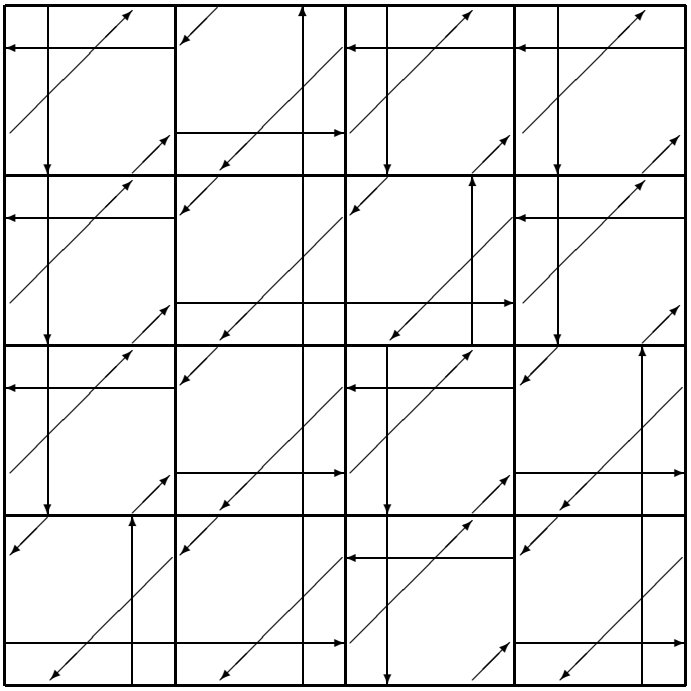


Figure 6: Color generator wiring diagram for $W = E_{3,1}$.

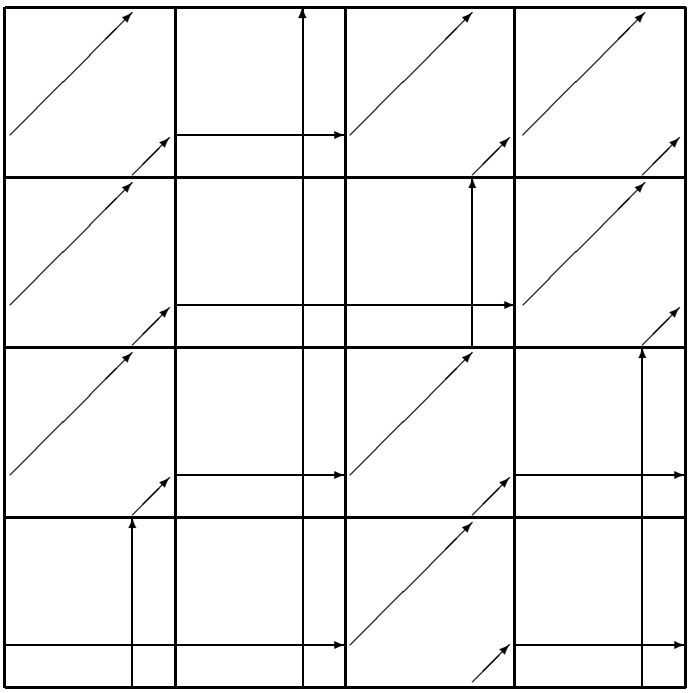
3.4. An example

Below, we present the transmitter/color generator diagrams for

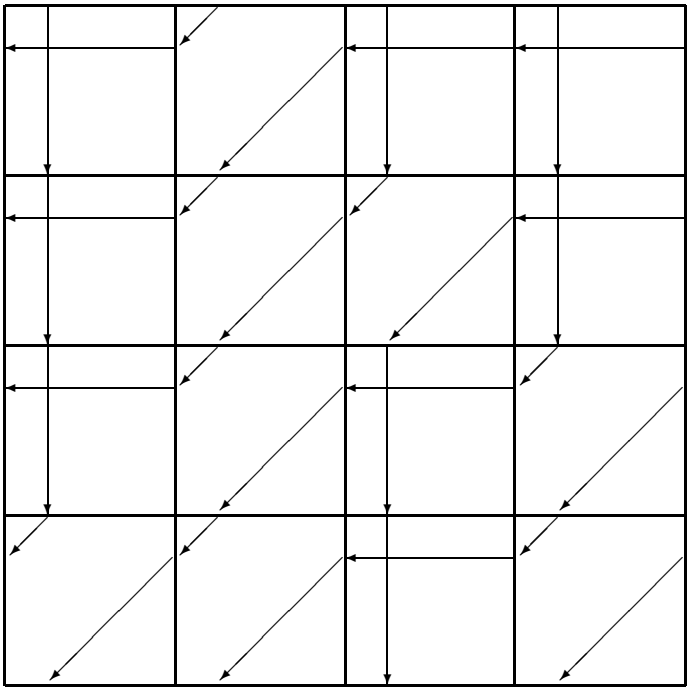
$$W = E_{12} + E_{22} + E_{23} + E_{32} + E_{34} + E_{41} + E_{42} + E_{44}.$$



For simplicity, we shall split this diagram into the transmitter diagram and color generator diagrams.



Transmitter diagram for $W = E_{12} + E_{22} + E_{23} + E_{32} + E_{34} + E_{41} + E_{42} + E_{44}$.



Color generator diagram for $W = E_{12} + E_{22} + E_{23} + E_{32} + E_{34} + E_{41} + E_{42} + E_{44}$.

For color assignments, see the example in Section 3.6.

3.5. The wiring permutation

The wiring matrix W gives rise to an element of S_{2N} as follows.

The transmitter diagrams associated to W give rise to a mapping

$$\{X_1, ..., X_{2N}\} \to \{CG_1, ..., CG_{2N}\}.$$

Since the transmitter wiring paths never merge, the construction of the transmitter wiring diagrams implies that this map is one-to-one. The fact that this map is 1-1 *forces* it to also be onto. Therefore, it represents an element of S_{2N}. We denote this element by $\pi_W \in S_{2N}$ and call it the **wiring permutation**.

Example 9.

- If W is the 4×4 matrix of all $1's$ then $\pi_W = (1,5)(2,6)(3,7)(4,8)$, in disjoint cycle notation.
- If W is the 4×4 matrix of all $0's$ then $\pi_W = 1$ (the identity).
- If $W = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ then $\pi_W = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 1 & 3 & 4 & 5 & 6 & 7 & 8 \end{bmatrix} = (1,2)$.

- If $W = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ then $\pi_W = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 3 & 2 & 4 & 5 & 6 & 7 & 8 \end{bmatrix} = (2,3)$.
- If $W = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ then $\pi_W = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 3 & 2 & 4 & 5 & 6 & 7 & 8 \end{bmatrix} = (2,3)$.
- If $W = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ then $\pi_W = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{bmatrix} = 1$.

3.6. The color assignments

We must perform several computations.

1. For each i, $1 \leq i \leq 2N$, the wiring matrix gives rise to a path from the edge labeled X_i to an edge labeled CG_j. In fact, $j = \pi_W(i)$, where π_W is the wiring permutation. Define the j^{th} **color code** C_j by

$$C_j = B(X_i, CG_j), \qquad 1 \leq j \leq 2N,$$

where $j = \pi_W(i)$. Here $X_i = d_{\sigma^{-1}(i)}$ and $CG_i = d_{\tau^{-1}(i)}$, for $1 \leq i \leq 2N$. This implies

$$C_j = B(d_{\sigma^{-1}(\pi_W^{-1}(j))}, d_{\tau^{-1}(j)}), \tag{4}$$

for $1 \leq j \leq 2N$.

2. For each pair (i,j), $1 \leq i,j \leq N$, the square $Q_{i,j}$ has a color generator wiring path which passes into its bottom edge. That path can be traced back to a edge CG_k, for some $1 \leq k \leq 2N$. Give $Q_{i,j}$ the color $c_{i,j}$ associated to the code C_k. We call the matrix $(c_{i,j})_{1 \leq i,j \leq N}$ the **color codes matrix**. (This description of the color decoding algorithm relies on column 7, lines 30-36 of [Gh]. A different description is given for the same algorithm in column 12, lines 29-31 of [Gh]. In this paper, we shall *assume that the description of the color decoding algorithm in column 7, lines 30-36, is the correct one.*)

Example 10. $W = 0$ and initial assignments $\{d_1, ..., d_8\} \to \{X_1, ..., X_8\}$, $\{d_1, ..., d_8\} \to \{CG_1, ..., CG_8\}$ given by $\sigma = 1$, $\tau = 1$:

$$X_i = d_i, \quad CG_i = d_i, \quad 1 \leq i \leq 8.$$

Then $\pi_W = 1$ and all buttons are off (dark). The matrix of color codes is then

$$\begin{pmatrix} C_1 & C_2 & C_3 & C_4 \\ C_1 & C_2 & C_3 & C_4 \\ C_1 & C_2 & C_3 & C_4 \\ C_1 & C_2 & C_3 & C_4 \end{pmatrix},$$

where

$$\begin{aligned} C_1 &= B(X_1, CG_1) = B(d_1, d_1) = [0,0,0] = \text{off}, \\ C_2 &= B(X_2, CG_2) = B(d_2, d_2) = [0,0,0] = \text{off}, \\ C_3 &= B(X_3, CG_3) = B(d_3, d_3) = [0,0,0] = \text{off}, \\ C_4 &= B(X_4, CG_4) = B(d_4, d_4) = [0,0,0] = \text{off}, \end{aligned}$$

using the table of B-values given in §3.1.

If each button is pressed (so $W_{i,j} = 1$ for all i, j) then $\pi_W = (1,5)(2,6)(3,7)(4,8)$ and all buttons are colored red (color 1). Indeed, the matrix of color codes is then

$$\begin{pmatrix} C_2 & C_3 & C_4 & C_5 \\ C_3 & C_4 & C_5 & C_6 \\ C_4 & C_5 & C_6 & C_7 \\ C_5 & C_6 & C_7 & C_8 \end{pmatrix},$$

where

$$\begin{aligned} C_1 &= B(X_5, CG_1) = B(d_5, d_1) = [1,0,0] = \text{color } 1, \\ C_2 &= B(X_6, CG_2) = B(d_6, d_2) = [1,0,0] = \text{color } 1, \\ C_3 &= B(X_7, CG_3) = B(d_7, d_3) = [1,0,0] = \text{color } 1, \\ C_4 &= B(X_8, CG_4) = B(d_8, d_4) = [1,0,0] = \text{color } 1, \\ C_5 &= B(X_1, CG_5) = B(d_1, d_5) = [1,0,0] = \text{color } 1, \\ C_6 &= B(X_2, CG_6) = B(d_2, d_6) = [1,0,0] = \text{color } 1, \\ C_7 &= B(X_3, CG_7) = B(d_3, d_7) = [1,0,0] = \text{color } 1, \\ C_8 &= B(X_4, CG_8) = B(d_4, d_8) = [1,0,0] = \text{color } 1. \end{aligned}$$

Example 11. $W = 0$ and initial assignments $\{d_1, ..., d_8\} \to \{X_1, ..., X_8\}$, $\{d_1, ..., d_8\} \to \{CG_1, ..., CG_8\}$ given by $\sigma = (6,7)$, $\tau = (2,3)(5,8)$:

$d_1 = [0,0,0]$	X_1	CG_1
$d_2 = [0,0,1]$	X_2	CG_3
$d_3 = [0,1,0]$	X_3	CG_2
$d_4 = [0,1,1]$	X_4	CG_4
$d_5 = [1,0,0]$	X_5	CG_8
$d_6 = [1,0,1]$	X_7	CG_6
$d_7 = [1,1,0]$	X_6	CG_7
$d_8 = [1,1,1]$	X_8	CG_5

The matrix of color codes is then

$$\begin{pmatrix} C_1 & C_2 & C_3 & C_4 \\ C_1 & C_2 & C_3 & C_4 \\ C_1 & C_2 & C_3 & C_4 \\ C_1 & C_2 & C_3 & C_4 \end{pmatrix},$$

where

$$\begin{aligned} C_1 &= B(X_1, CG_1) = B(d_1, d_1) = [0,0,0] = \text{off}, \\ C_2 &= B(X_2, CG_2) = B(d_2, d_3) = [0,1,1] = \text{off}, \\ C_3 &= B(X_3, CG_3) = B(d_3, d_2) = [0,1,1] = \text{off}, \\ C_4 &= B(X_4, CG_4) = B(d_4, d_4) = [0,0,0] = \text{off}, \\ C_5 &= B(X_5, CG_5) = B(d_5, d_8) = [1,1,1] = \text{color 4}, \\ C_6 &= B(X_6, CG_6) = B(d_7, d_6) = [1,1,1] = \text{color 4}, \\ C_7 &= B(X_7, CG_7) = B(d_6, d_7) = [1,1,1] = \text{color 4}, \\ C_8 &= B(X_8, CG_8) = B(d_8, d_5) = [1,1,1] = \text{color 4}. \end{aligned}$$

Here every button in the 4×4 array is colored off (dark).

Example 12.

(a) Take initial assignments $\{d_1, ..., d_8\} \to \{X_1, ..., X_8\}$, $\{d_1, ..., d_8\} \to \{CG_1, ..., CG_8\}$ *given by* $\sigma = \tau = (5,6)(7,8)$. *This can be solved for dark (press no buttons) or yellow (press all buttons).*

(b) Take initial assignments $\{d_1, ..., d_8\} \to \{X_1, ..., X_8\}$, $\{d_1, ..., d_8\} \to \{CG_1, ..., CG_8\}$ *given by* $\sigma = \tau = (1,3)(2,4)$. *This can be solved for off (press no buttons) or green (press all buttons).*

(c) Take initial assignments $\{d_1, ..., d_8\} \to \{X_1, ..., X_8\}$, $\{d_1, ..., d_8\} \to \{CG_1, ..., CG_8\}$ *given by* $\sigma = \tau = (1,4)(2,3)$. *This can be solved for off (press no buttons) or blue (press all buttons).*

Remark 13. *Even identical looking initial color configurations (when* $W = 0$*) can represent different Ghaly machines (which, when* $W \neq 0$*, look different in general). This is in sharp contrast to Merlin's machine, where identical looking initial color configurations correspond to different machines.*

4. Mathematical properties

4.1. Some interesting questions

There are several questions which immediately arise.

1. Let $\phi : M_N(\mathbf{F}_2) \to S_{2N}$ be defined by $\phi(W) = \pi_W$. Is ϕ onto? (No.)
2. Let $\phi : M_N(\mathbf{F}_2) \to S_{2N}$ be defined by $\phi(W) = \pi_W$. Does the image of ϕ contain all of $S_I \times S_J$, where I and J are as in (1)? (Yes.)

   This is closely related to the question of whether or not the puzzle posed by a "random" initial assignment of codes is actually solvable.
3. Is ϕ a homomorphism? (No.)
4. Given W, how do you compute π_W? (There is a formula - see Proposition 15 below.)
5. What is the probability that $W \in M_N(\mathbf{F}_2)$, is the solution of *some* puzzle (provided the initial assignment of codes is as described in Ghaly's patent)?

We shall give detailed answers to these questions later.

Other questions which arise are the following.

1. What is the image of ϕ?

   This question has no nice, compact answer. Using the computer algebra program MAPLE, running for about 1 day on a PC with a 750Mz processor and 1G of RAM, a list of all possible permutations were computed. There are 6902 permutations in S_8 of the form π_W, for some $W \in M_4(\mathbf{F}_2)$. However, only 576 of them belong to the subset $S_I \times S_J \subset S_8$, where I, J are as in (1). Since $|S_I \times S_J| = 4!^2 = 576$, each element of $S_I \times S_J$ may be represented as a π_W.

   Proposition 15 permits us to give the following reinterpretation of this information: these 6902 permutations are all the elements of S_8 which can be written as a product of certain simple transpositions taken in a certain order, corresponding to a matrix in $M_4(\mathbf{F}_2)$.
2. What is the probability that a "random" initial assignment is solvable?

   There are $|S_I \times S_J|^2 = 4!^4$ possible initial assignments. How many of these are solvable? This is unknown.
3. What is the probability that a "random" color configuration of the array of buttons arises from some $\sigma, \tau \in S_I \times S_J$ via an initial assignment of codes?

There are 5^{16} possible color configurations which can arise. There are $(4!^2)^2$ possible choices of ordered pairs (σ, τ), with $\sigma, \tau \in S_I \times S_J$. However, due to the construction of the machine, only $4!^2$ of these are actually different *looking* [4] initial color configurations (when $W = 0$) since such a configuration is determined by $\sigma \cdot \tau^{-1}$ (see the proof of Theorem 32 below). Therefore, the probability desired is

$$576/5^{16} = (3.7...) \times 10^{-9}.$$

(In fact, if Ghaly's machine was somehow reconfigured so that the only possible colors were "off" and "on" then the the probability that a "random" color configuration of the array of buttons arises from some $\sigma, \tau \in S_I \times S_J \cong S_4 \times S_4$ via an initial assignment of codes is less than a 1/100 chance [5].)

4.2. $\phi : M_N(\mathbf{F}_2) \to S_{2N}$

In the remainder of this section, we give some properties of the mapping $\phi : M_N(\mathbf{F}_2) \to S_{2N}$. Here, $N > 2$ can be any integer.

For brevity, let $\pi_{i,j} = \pi_{E_{i,j}}$, $1 \le i \le 2N$, $1 \le i \le 2N$. This is the permutation associated to pressing the $(i,j)^{th}$ button.

A **simple transposition** is a permutation which swaps two consecutive numbers: $(k, k+1)$, for some $k > 0$.

Lemma 14. *Each $\pi_{i,j}$ is order 2. In fact, $\pi_{i,j} = (i+j-1, i+j)$, $1 \le i, j \le N$. In particular, $\pi_{i,j}$ is a simple transposition.*

Proof. This follows immediately from the transmitter wiring diagrams. □

It is known that the simple transpositions generate the group S_{2N} (see for example, [JKT], chapter 4). Thus it is conceivable that ϕ is an onto map. However, this turns out not to be the case. The following proposition computes $\phi(W)$ explicitly.

Proposition 15. *For any $W = (W_{i,j}) \in M_N(\mathbf{F}_2)$, we have*

$$\pi_W = \prod_{(i,j)}{}' \pi_{i,j}^{W_{i,j}} = \prod_{(i,j)}{}' (i+j-1, i+j)^{W_{i,j}},$$

where $\prod'_{(i,j)} = \prod_{i=N}^{1} \prod_{j=N}^{1}$ (note that the product runs top-to-bottom, right-to-left, in that order).

[4] Recall even identical looking initial color configurations (when $W = 0$) can represent different Ghaly machines.

[5] This too is in stark contrast to the case of Merlin's machine, where the corresponding probability is 1/4 [J].

Proof. This follows immediately from the transmitter wiring diagrams and the fact (seen by inspection) that to "undo" the transmitter wiring diagram corresponding to W you may start from the upper right-hand corner and work right-to-left, top-to-bottom. This is the opposite order than what was indicted by $\prod'$, as desired. □

Remark 16. *Note $\pi_{E_{1,1}+E_{1,2}} = \pi_{E_{1,1}} \cdot \pi_{E_{1,2}} \neq \pi_{E_{1,2}} \cdot \pi_{E_{1,1}}$. This does not mean that, pressing button $(1,1)$ then $(1,2)$ yields a different wiring permutation, hence a different output, than if you pressed button $(1,2)$ then $(1,1)$. This is because the output of Ghaly's machine only depends on the wiring matrix W and the initial assignments σ, τ.*

Call two elementary matrices $E_{i,j}$, $E_{i',j'}$ **independent** if $\{i+j-1, i+j\} \cap \{i'+j'-1, i'+j'\}$ is empty. Two sets S, S' of elementary matrices will be said to be **independent** if each element of S is independent from each element of S'. Call $W, W' \in M_N(\mathbf{F}_2)$ **independent** if $W = \sum_{E \in S} E$, $W' = \sum_{E \in S'} E$, where S, S'' are independent sets of elementary matrices. The following result follows from the previous Proposition.

Corollary 17. *If $W, W' \in M_N(\mathbf{F}_2)$ are independent then $\phi(W + W') = \phi(W)\phi(W')$.*

Define the **weight** of the wiring matrix W to be equal to the number of non-zero entries in the array.

Proposition 18. *For any $W = (W_{i,j}) \in M_N(\mathbf{F}_2)$, π_W is an even permutation if and only if the weight of W is even. In other words, π_W is of even order if and only if the total number of buttons which have been pressed is even.*

Proof. Arrange the labeled array

$$\begin{array}{cccccc} & 1 & 2 & \ldots & N & \\ 1 & & & & & N+1 \\ 2 & & & & & N+2 \\ \vdots & & & & & \vdots \\ N & & & & & 2N \\ & N+1 & N+2 & \ldots & 2N & \end{array}$$

as two columns:

$$\begin{array}{cc} 1 & 1 \\ 2 & 2 \\ \vdots & \vdots \\ N & N \\ N+1 & N+1 \\ N+2 & N+2 \\ \vdots & \vdots \\ 2N & 2N \end{array}$$

Any paths from the X_i's to the CG_j's in the former array corresponding to the wiring permutation π_W may be represented by "crossing paths" from the first column to the second column in the latter array. The parity of the number of crossings in either array is the same. The parity of the number of crossings in the transmitter wiring diagram is equal to the weight of W since, by construction, each 1 in W corrsponds to a change in course, resulting in a crossing. The number of crossings in the second diagram has the same parity as the crossing number in [NJ], page 410. Moreover, it is known (from [NJ] or [JKT], §4.2) that the parity of the crossing number of the second diagram is equal to the parity of the permutation π_W. □

Let ${}^tW$ denote the transpose of W.

Lemma 19. *$\pi_{{}^tW} = (\pi_W)^{-1}$. In other words, $\phi(W)^{-1} = \phi({}^tW)$, for all $W \in M_N(\mathbf{F}_2)$.*

Proof. The wiring associated to W yields paths from the X_i's to the CG_j's. These paths are reversed by using the transpose matrix. □

4.3. How do you compute W, given π_W?

Is there a simple algorithm for constructing each π_W as a product of simple transpositions?

Yes (or no, depending on how you define "simple").

First, one can use the *reduce* command in the **coxeter**[6] package of Stembridge [St] to rewrite *any* permutation in S_8 (hence any permutation of the form π_W) as a product of simple transpositions.

Second, we have the identities:

$$(a_1, a_2)(a_1, a_2, a_3, ..., a_k) = (a_1, a_3, ..., a_k),$$

and

$$(a_1, a_2)(a_1, a_3)(a_1, a_2) = (a_2, a_3).$$

For examples of how these can be used to decompose a permutation into a product of simple transpositions, see the example below.

Example 20. Let $\pi = (3, 6, 4)(5, 8, 7)$. This was picked from a set of permutations in the image of ϕ (determined using a computer program written by the author), so we know $\pi = \pi_W$ for some $W \in M_4(\mathbf{F}_2)$.

[6]This **coxeter** package was written by Stembridge for MAPLE version 3 or version 4. It does not run on MAPLE versions 5 or 6. The Ghaly machine simulations I wrote run on MAPLE versions 6. The may work on version 5 but have not been tested on versions 3 or 4.

We shall compute W. By Lemma 19, $(3,4,6)(5,7,8) = \pi^{-1} = \phi({}^tW)$. We shall solve for ${}^tW$ first since $(3,4,6)(5,7,8)$ is easier to decompose. We have

$$(7,8)(3,4)(3,4,6)(5,7,8) = (7,8)(3,6)(5,7,8) = (3,6)(5,7),$$

and

$$(3,4)(3,6)(5,7)(3,4) = (4,6)(5,7),$$

and

$$(5,6)(4,6)(5,7)(5,6) = (5,6)(4,6)(5,6)(5,6)(5,7)(5,6) = (4,5)(6,7).$$

Thus,

$$(5,6)(3,4)(7,8)(3,4)\pi^{-1}(3,4)(5,6) = (4,5)(6,7),$$

so $\pi^{-1} = (3,4)(7,8)(3,4)(5,6)(4,5)(6,7)(5,6)(3,4)$, or

$$\pi = (3,4)(5,6)(6,7)(4,5)(5,6)(3,4)(7,8)(3,4).$$

This forces

$$\pi = \pi_{1,3}\pi_{2,4}\pi_{3,4}\pi_{1,4}\pi_{2,4}\pi_{1,3}\pi_{4,4}\pi_{1,3}.$$

This is sufficient to solve the puzzle.

To solve for W, however, we want to use Proposition 15. For this purpose, we wish to rewrite this product so that terms associated to upper buttons appear first. We can commute some of the terms towards the front as follows:

$$\pi = \pi_{2,4}\pi_{3,4}\pi_{4,4}\pi_{1,3}\pi_{1,4}\pi_{2,4}\pi_{1,3}\pi_{1,3} = \pi_{2,4}\pi_{3,4}\pi_{4,4}\pi_{1,3}\pi_{1,4}\pi_{2,4}.$$

Now since $\pi_{i,j}$ only depends on the value $i+j$ (and not on i,j), we may rewrite this as:

$$\pi = \pi_{4,2}\pi_{4,3}\pi_{4,4}\pi_{2,2}\pi_{2,3}\pi_{2,4},$$

so

$${}^tW = E_{2,2} + E_{2,3} + E_{2,4} + E_{4,2} + E_{4,3} + E_{4,4}.$$

This has been verified by a direct calculation as well, using Lemma 14.

4.4. The support

Which color codes occur in a given state of the machine? We want to know the answer to this because if all the color codes are equal then the puzzle is solved. This motivates the following definition.

Definition 21. *Let $C = C(W,\sigma,\tau) = (c_{ij})_{1\le i,j\le N}$ denote the color array associated to $\sigma,\tau \in S_I \times S_J$ (the permutations associated to the initial assignments $\{X_i\}_{1\le i\le 2N} \to \{d_j\}_{1\le j\le 2N}$, $\{CG_i\}_{1\le i\le 2N} \to \{d_j\}_{1\le j\le 2N}$) and $W \in M_N(\mathbf{F}_2)$. Here each c_{ij} is a color code C_k as defined in §3.6. We call the indices of the color codes which occur*

$$\{k \mid 1 \leq k \leq 2N, C_k = c_{ij}, \text{ some } 1 \leq i, j \leq N\},$$

the **support** *of the state* (W, σ, τ). *Sometimes we abuse terminology and also call* $\{C_k \mid 1 \leq k \leq 2N, C_k = c_{ij}, \text{ some } 1 \leq i, j \leq N\}$, *the* **support** *of the state.*

If $C_k = B(X_j, CG_k)$ *then we call* j *the* **origin** *of* k. *Note that the origin of any* $k \in \{1, 2, ..., 2N\}$ *depends only on* W *and not on* σ *or* τ.

The remainder of this section shall be devoted to the "explicit computation" of $supp(W)$.

Theorem 22.

- *For* $5 \leq j \leq 8$, $i \notin supp(W)$ *if and only if* $W_{i-4,j} = 0$, *for all* $1 \leq j \leq 4$.
- *For* $2 \leq j \leq 4$, $j \notin supp(W)$ *if and only if* $W_{1,j} = 1$ *and* $W_{1,j'} = 0$, *for all* $j' < j$.
- $1 \notin supp(W)$ *if and only if* $W_{1,1} = 1$.

This completely determines the support of any $W \in M_N(\mathbf{F}_2)$.

Corollary 23.

- $5 \notin supp(W)$ *if and only if* $\{1, 2, 3, 4\} \subset supp(W)$.
- *At most one of* $\{1, 2, 3, 4\}$ *does not occur in* $supp(W)$.

Proof of Theorem. This follows from §3.6 and the color generator diagrams in §3.3.3. □

Can pressing a single button (only) *ever* be the solution to Ghaly's game?

Let us first look at some examples.

Example 24. Let $W = E_{4,4}$, which corresponds to pressing the lower right-hand button.

In this case, $\pi_W = (7, 8)$ by Lemma 14, and $supp(W) = \{1, 2, 3, 4, 8\}$ by the previous theorem. By (4), we know

$$C_j = B(d_{\sigma^{-1}(j)}, d_{\tau^{-1}(j)}),$$

for $1 \leq j \leq 4$, where $\sigma, \tau \in S_I \times S_J$ are to be "solved for" (using Theorem 32, say). Since $1 \leq \sigma^{-1}(j), \tau^{-1}(j) \leq 4$, using the table of B-values in §3.1, we find that C_1 is always off, no matter what $\sigma, \tau \in S_I \times S_J$ are choosen. Thus $W = E_{4,4}$ never yields a solved state.

Example 25. Let $W = E_{1,1}$, which corresponds to pressing the upper left-hand button.

In this case, $\pi_W = (1, 2)$ by Lemma 14, and $supp(W) = \{2, 3, 4, 5\}$ by the previous theorem. By (4), we know

$$C_3 = B(d_{\sigma^{-1}(3)}, d_{\tau^{-1}(3)}),$$

where $\sigma, \tau \in S_I \times S_J$. Since $1 \leq \sigma^{-1}(3), \tau^{-1}(3) \leq 4$, using §3.1, we find that C_3 is always off, no matter what $\sigma, \tau \in S_I \times S_J$ are choosen. Thus $W = E_{1,1}$ never yields a solved state.

Proposition 26. *Suppose $W \in M_4(\mathbf{F}_2)$ satisfies: $W_{1,1} = 0$. Then W never yields a solved state.*

Proof. The previous theorem implies $1 \in supp(W)$. Since $W_{1,1} = 0$, we know $\pi_W(1) = 1$, by Proposition 15. By (4), we know

$$C_1 = B(d_{\sigma^{-1}(1)}, d_{\tau^{-1}(1)}).$$

By the same reasoning as in Example 25, this color must be off, so W never yields a solved state. □

Of the $2^{16} = 65536$ matrices in $M_4(\mathbf{F}_2)$, $2^{15} = 32768$ of them satisfy the conditions of the above proposition. This proves the following result.

Theorem 27. *Pick a matrix in $M_4(\mathbf{F}_2)$ at random. The probability that it never yields a solved state is at least* $32768/65536 = .5$.

The following example is due to Roger Heppermann. Notice that the matrix W below fails to meet the conditions of the proposition above.

Example 28. Take

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

and $\sigma = 1$, $\tau = (1, 2, 3, 4)$. Then $supp(W) = \{2, 3, 4, 5, 6, 7\}$ and (W, σ, τ) is a solved state. Every button is colored red.

5. Solution strategies

In this section, we collect some results related to solutions (if any) of the puzzle.

We shall discuss

- how to interprete a solution mathematically,
- when a solution is possible,
- how to find the permutation τ in the case $\sigma = 1$ (or how to find the permutation σ in the case $\tau = 1$).

5.1. What is a solution?

The object of the puzzle is: given σ and τ (i.e., an initial assignment of "Boolean values" to the labels X_i's and the CG_j's, find a sequence of button presses (if it exists) which result in all the buttons having the same color.

Example 29. If $\sigma = \tau = 1$ and all the $W_{i,j} = 0$ then all the squares have color 1 (the color associated to $[1, 0, 0]$).

Are there other assignments σ, τ which yield a solution with all the $W_{i,j} = 0$? Yes, but they must be very special permutations. To see the assignments associated to solving the puzzle where all colors have color i, look at the $2N \times 2N$ matrix coordinates of the table for B in §1 associated to color i. Put a 1 in an $N \times N$ matrix in these coordinates and a 0 elsewhere. Denote this matrix by B_i, $1 \leq i \leq N$. Notice that all these matrices B_i are order 2. Let $b_i \in S_{2N}$ denote the permutation associated to B_i.

Example 30. When $N = 4$ there are 4 such 8×8 "solution matrices":

$$B_1 = \begin{pmatrix} 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 0&0&0&0&0&1&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0 \\ 0&1&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \end{pmatrix},$$

$$B_3 = \begin{pmatrix} 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \end{pmatrix}, \quad B_4 = \begin{pmatrix} 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \end{pmatrix}.$$

Note that these four solution matrices are permutations matrices in S_8 but <u>not</u> in $S_I \times S_J$. Only B_1 is of the form π_W, for some $W \in M_4(\mathbf{F}_2)$. This was checked by computer using an exhausive search [7]. They are associated to the permutations $b_1 = (1,5)(2,6)(3,7)(4,8)$, $b_2 = (1,6)(2,5)(3,8)(4,7)$, $b_3 = (1,7)(2,8)(3,5)(4,6)$, $b_4 = (1,8)(2,7)(3,6)(4,5)$, respectively.

[7] In fact, $B_1 = \pi_J$, where $J = J_4$ is the 4×4 matrix of all 1's. However, each of B_2, B_3, B_4 is conjugate to B_1 by some element of $S_I \times S_J$.

Lemma 31. *Suppose $W = 0$. Permutations $\sigma, \tau \in S_I \times S_J \subset S_{2N}$ associated to the assignments of the elements of $\mathbf{F}_2^n = \{d_1, ..., d_{2N}\}$ to $\{X_1, ..., X_{2N}\}$ and to $\{CG_1, ..., CG_{2N}\}$ yields a solution where all the buttons have color k if $\tau\sigma^{-1} = b_k$, for some k.*

Proof. The support of $W = 0$ is

$$supp(W) = \{N + 1, ..., 2N\}.$$

The i^{th} color code is given by $C_i = B(X_i, CG_i) = B(d_{\sigma^{-1}(i)}, d_{\tau^{-1}(i)})$, $1 \leq i \leq 2N$. Multiplying σ and τ by σ^{-1} will only permute the color codes C_i but will not change there values. Therefore, we may assume without loss of generality that $\sigma = 1$. We may also replace τ by b_k, by hypothesis. By construction, all the buttons in the array have color k. □

Theorem 32. *The puzzle is solved (with color i on each button) if $\tau^{-1}\pi_W\sigma = B_i$, where $\sigma, \tau \in S_I \times S_J$ are the permutations associated to the initial assignments $\{X_i\} \to \{d_j\}$, $\{CG_i\} \to \{d_j\}$, and where $W \in M_N(\mathbf{F}_2)$.*

Moreover, if the state (W, σ, τ) is solved then so is (W, σ', τ'), where

- *$\tau'\tau^{-1}$ only permutes indices not in the support of the state, or*
- *$\sigma'\sigma^{-1}$ only permutes indices of the X_i's which are origins of indices of CG_j's which are not in the support of the state, or*
- *$\sigma'\sigma^{-1}$ only permutes indices of the X_i's which do not change the values of the color codes C_k's (where k is in the support of the state (W, σ, τ)).*

Proof. We show that if $\tau^{-1}\pi_W\sigma = B_i$ then the puzzle is solved.

There are several permutations floating around: the initial assignment permutations σ and τ and the wiring permutation π_W. The permutation σ essentially assigns to each X_i a distinct element of $\mathbf{F}_2^n$: $X_i = d_{\sigma(i)}$, $1 \leq i \leq 2N$. The permutation τ essentially assigns to each CG_i a distinct element of $\mathbf{F}_2^n$: $CG_i = d_{\tau(i)}$, $1 \leq i \leq 2N$. The wiring permutation π_W essentially assigns each X_j to a distinct $CG_{j'}$: $X_j = CG_{\pi_W(j)}$, $1 \leq j \leq 2N$.

Putting all these together, we see that the code d_j on the left or bottom of the machine array is sent, via the transmitter wiring diagram, to the code $d_{\tau^{-1}(\pi_W(\sigma(j)))}$, $1 \leq j \leq 2N$.

Because the color pairing defined by B yields the color code associated to color i on each button, the theorem follows.

This proves the "if" direction.

For the "only if" direction: Assume now that the puzzle is solved. The construction of the color assignments for the buttons implies that the changes indicated by the itemized list of actions will not affect the color configuration of the state (W, σ, τ). □

6. Summary

The patent description [Gh] describes an electronic game played by pressing N^2 buttons, aranged in an $N \times N$ array, where N is *a priori* an integer greater than 1. The buttons can potentially be any one of N colors (or off - also called "dark" - which is not counted as a color). The patent specifications force N to be a power of 2 which is at least 4.

The patent description is fairly technical and several typographical errors, ambiguities and inconsistencies have crept in. In this paper, we assume column 6, lines 35-45 are correct (to compute B) and we assume column 7, lines 30-36 are correct (to compute C_i).

We find several striking differences with the Lights Out family of games [J].

1. Ghaly's game can only be played on an $N \times N$ array of buttons where $N = 4, 8, 16, ...$ Lights Out can be played on any rectangular array $M \times N$, where $M > 1,\ N > 1$ are arbitrary.

2. Press any sequence of buttons at random, as many as you want. It is more likely than not that that sequence is not the solution of *any* starting position of Ghaly's game. However, it is *always* the case that that sequence is the solution of *some* starting position of Lights Out.

3. Given a random configuration of 4×4 colored buttons, the probability is quite small (less than 3 in a million) that the configuration can be solved for Ghaly's game. Given a random configuration of 4×4 buttons which are on or off, the probability is at least $1/2^4$ that the configuration can be solved for the Lights Out game.

4. Identical looking initial color configurations (when $W = 0$) can (and often do) represent different Ghaly machines. In other words, when no buttons are pressed, the colored buttons have the *same* colorings but when buttons are pressed then will look *different*. This is in sharp contrast to Merlin's machine, where identical looking initial color configurations correspond to identical machines.

Ackowledgements This paper describes some patent analysis done around 2000-2001 for patent laywers Anthony Sitko and Roger Heppermann at the Chicago law firm Marshall, O'Toole, Gerstein, Murray, & Borun. I am very grateful for them allowing me to work on this interesting project and for permitting me to publish these results. I think Anthony Sitko and Roger Heppermann for useful discussions and an anonymous referee for very helpful comments and corrections.

References

[C] J. H. Conway, **Regular algebra and finite machines**, Chapman and Hall Ltd, London, 1971.

[Gh] Nabil Ghaly, *Electronic hand held logic game*, patent date: 2-15-1994, date filed: 9-3-1991, patent number: 5,286,037.

[Gr] R. Grimaldi, **Discrete and combinatorial mathematics**, 4^{th} ed., Addison-Wesley-Longman, 1999

[J] D. Joyner, **Adventures with group theory: Rubik's cube, Merlin's machine, and other mathematical toys**, The Johns Hopkins Univer. Press, 2002.

[NJ] —— and G. Nakos , **Linear algebra and its applications**, Brooks-Cole, 1998

[JKT] ——, R. Kreminski, J. Turisco, **Applied abstract algebra**, The Johns Hopkins Univ Press, 2004.

[St] J. Stembridge, MAPLE package **coxeter**, available for download at the URL `http://www.math.lsa.umich.edu/~jrs/coxtut.html`