CDM

Pólya-Redfield Theory

Klaus Sutner

Carnegie Mellon University

26-poly-redfield 2017/12/15 23:15

Let's push the ideas from Burnside's lemma a little bit further to make it easier to deal with questions involving different types of configurations. First, a slight abstraction.

Let

$$V = \{v_1, \ldots, v_n\}$$

be a set of vertices (arbitrary objects), and

$$C = \{c_1, \ldots, c_m\}$$

a set of colors. A coloring is a map $V \to C$.

We write $X = V \to C$ for the set of all colorings.

Let G be a subgroup of $\mathfrak{S}_n$ and define the natural left action of G on X by

$$\rho \cdot f = \rho \circ f$$

$$[n] \xrightarrow{\rho} [n] \xrightarrow{f} [k]$$

Thus, $\rho \in G$ permutes the vertices, and then we apply the given coloring map.

What are the invariant elements $f = \rho \cdot f$ under this action?

Clearly, if ρ is just a single cycle then f must be constant on the whole cycle.

In the general case, we consider the cycle decomposition of ρ.

Write the cycle decomposition of ρ, including fixed points, as:

$$\rho = (v_{1,1}, \ldots, v_{1,q_1}), (v_{2,1}, \ldots, v_{2,q_2}), \ldots, (v_{p,1}, \ldots, v_{p,q_p})$$

Thus, the $v_{i,j}$ are all distinct and $\sum q_i = n$.

We write

$$\mathsf{cc}_i(\rho) = \text{ number of cycles of length } i \text{ in } \rho$$

$$\#\mathsf{c}(\rho) = \sum \mathsf{cc}_i(\rho)$$

Thus, $\#\mathsf{c}(\rho)$ is the total number of cycles in ρ, $1 \leq \#\mathsf{c}(\rho) \leq n$.

Also note that $\sum_i i \ \mathsf{cc}_i(\rho) = n$.

Lemma

The cardinality of X_ρ is $m^{\#\mathsf{c}(\rho)}$.

Proof.

f is in X_ρ iff f is constant on all the cycles of ρ.

But there are exactly m choices for the value of f on any one of the cycles.

□

Note, though, that this requires knowledge of the cycle number for each group element.

How about the cycle structure of all elements of D_n?

For pure rotations α^k the cycle structure is easy: there are $\gcd(n,k)$ many cycles of length $n/\gcd(n,k)$ each.

But, as we have seen, for reflections things a more complicated; we have to deal with 2-cycles and possibly fixed points.

For example, in an octagon there is a reflection (we write fixed points for clarity)

$$\rho = (1)(2,8)(3,7)(4,6)(5)$$

From Burnside's lemma we immediately have the

Corollary

The number of distinct orbits is $N = \frac{1}{|G|}\sum_{\rho\in G} m^{\#\mathsf{c}(\rho)}$.

For example, when D_4 is acting on the square we have

ρ	$\#\mathsf{c}(\rho)$	ρ	$\#\mathsf{c}(\rho)$
1	4	β	2
α	1	$\alpha\beta$	3
α^2	2	$\alpha^2\beta$	2
α^3	1	$\alpha^3\beta$	3

Hence

$$N = \frac{1}{8}(m^4 + 2m^3 + 3m^2 + 2m).$$

For D_5 acting on the pentagon we get

ρ	$\#\mathsf{c}(\rho)$	ρ	$\#\mathsf{c}(\rho)$
1	5	β	3
α	1	$\alpha\beta$	3
α^2	1	$\alpha^2\beta$	3
α^3	1	$\alpha^3\beta$	3
α^4	1	$\alpha^3\beta$	3

Hence

$$N = \frac{1}{10}(m^5 + 5m^3 + 4m).$$

Let us identify two Boolean functions f and g if

$$f(x_1, x_2, \ldots, x_k) = g(x_1 \oplus a_1, x_2 \oplus a_2, \ldots, x_k \oplus a_k)$$

where the a_i are bits.

So $a_i = 0$ means "leave the bit alone" and $a_i = 1$ means "flip the bit".

We want to count the Boolean functions modulo this equivalence.

As an example, there are 256 3-bit circuits. Flipping bits we could get the number down to $256/8 = 32$ but that would require all variants to be distinct. So we should expect something like 50 (wild guess).

As we will see shortly, the relevant group here is a Boolean group.

Definition

A group is Boolean if every element other than the identity has order 2.

Hence, in a Boolean group, we have

$$x + x = 0$$

for all x .

The additive notation is justified by the following exercise.

Exercise

Show that every Boolean group is commutative.

Example

Let $G = \langle \mathfrak{P}(A), \Delta, \emptyset \rangle$ where Δ denotes symmetric difference: $X \Delta Y = (X - Y) \cup (Y - X)$. Then G is a Boolean group.

Example

Any finite Boolean group arises in the following way:

$$\mathbb{B}_k = \left\langle \mathbf{2}^k, \oplus, \mathbf{0} \right\rangle$$

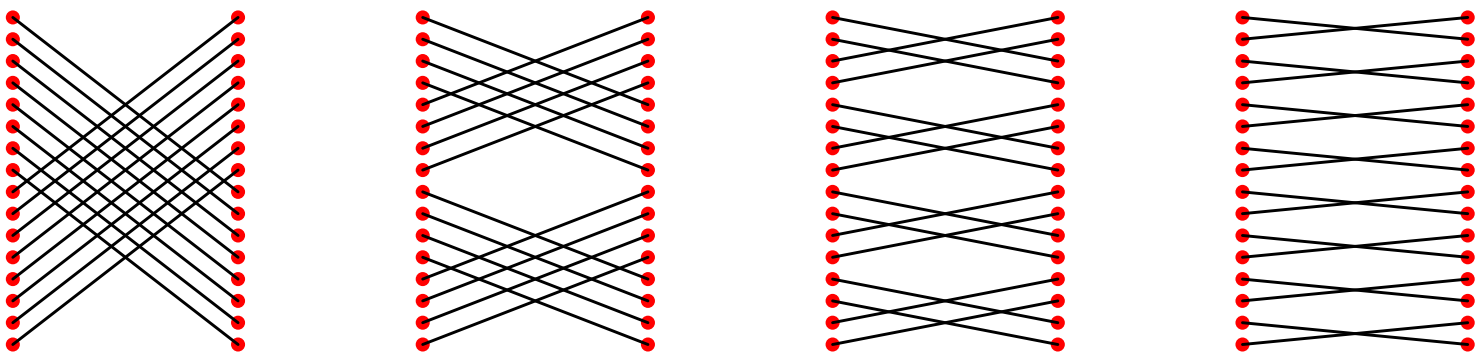
where $\mathbf{2}^k$ means binary lists of length k and $\oplus$ is point-wise exclusive or.

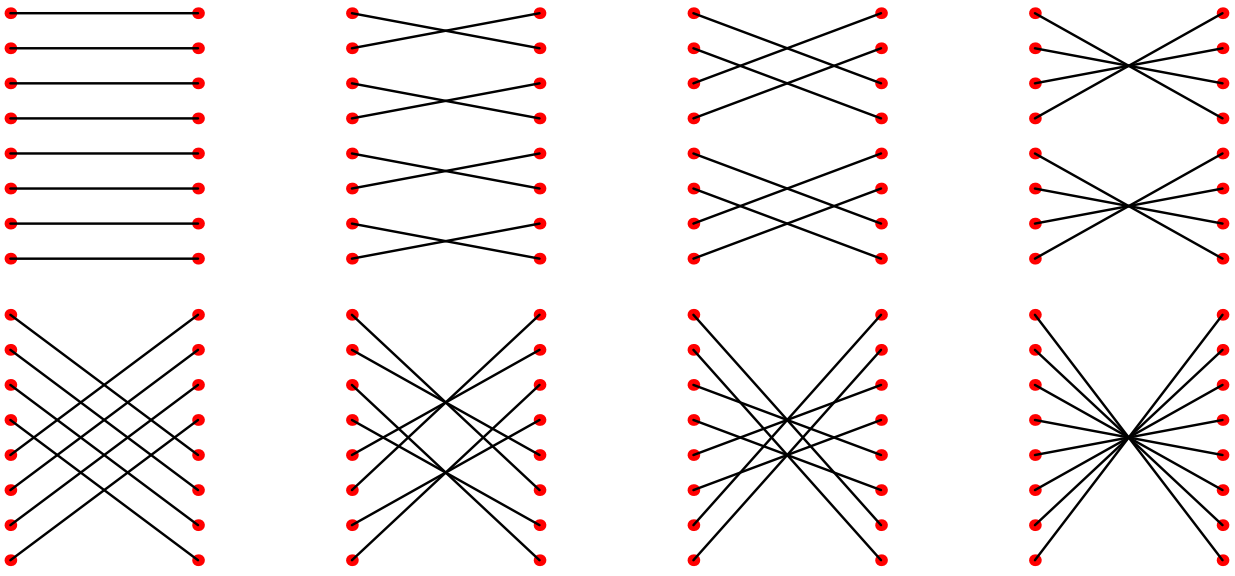
Note that $\mathbb{B}_k$ has k generators $\boldsymbol{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)$.

By Cayley's theorem we can identify $\mathbb{B}_k$ with a subgroup H of $\mathfrak{S}_{2^k}$, the full permutation group on 2^k points.

The permutation $\widehat{\boldsymbol{a}}$ associated with $\boldsymbol{a} \in \mathbb{B}_k$ is the map $\boldsymbol{x} \mapsto \boldsymbol{x} \oplus \boldsymbol{a}$.

For example, the pictures of the permutations $\widehat{\boldsymbol{e}_i}$ for $k = 4$ are shown below.





These are all the 8 permutations of the input values of a 3-bit circuit we can produce by flipping some of the bits.

We can think of a Boolean function $f : \mathbf{2}^k \to \mathbf{2}$ as a coloring of points $V = \mathbf{2}^k$ by just two colors: $C = \mathbf{2}$.

Now the Boolean group $\mathbb{B}_k$ acts on $\mathbf{2}^k$ in the obvious way:

$$\beta \cdot \boldsymbol{x} = \beta \oplus \boldsymbol{x}.$$

But then either β is the identity (in which case $\beta \cdot \boldsymbol{x} = \boldsymbol{x}$) or $\beta^2 \cdot \boldsymbol{x} = \boldsymbol{x}$. So the number of fixed points is either

$$2^{2^k} \qquad \text{or} \qquad 2^{2^{k-1}}$$

For example, here is the case where $\beta = \mathbf{1}$.

$\boldsymbol{x}$	$\mathbf{1} \cdot \boldsymbol{x}$
000	111
001	110
010	101
011	100

For a Boolean function to be invariant, half the values are determined by the other half. The table shows the case $\beta = \mathbf{1}$, but it is easy to see that this actually works for any β.

Hence the total number of orbits is

$$1/2^k \left(2^{2^k} + (2^k - 1)2^{2^{k-1}}\right) = 2^{2^{k-1}-k} \left(2^{2^{k-1}} + 2^k - 1\right)$$

For $k = 1, \ldots, 5$ we obtain the following values:

$$3, 7, 46, 4336, 134281216$$

Thus, flipping input bits reduces the number of ternary Boolean functions from 256 to 46.

The last result deals just with flipping input bits. Of course, there are several other natural operations we could modify a given circuit to obtain others – and even combinations thereof, see below.

- Flip the output bit.
- Reverse inputs.
- Rotate inputs.
- Permute inputs.

Exercise

Count the number of distinct circuits for some of these operations. How hard would it be to deal with combinations thereof?

So far, we have a good tool to count the total number of orbits.

We even get a general formula that depends only on the group G and applies to different kinds of configurations: m is just the number of colors, and the orbit count is a polynomial in m.

But we do not yet know how to determine the size of a specific orbit along the lines of the Tic-Tac-Toe problem: currently, there is somewhat tedious special computation for each given configuration.

To ameliorate this problem we will need to take a closer look at the number of cycles of each possible length:

$$\mathsf{cc}_1(\rho), \mathsf{cc}_2(\rho), \ldots, \mathsf{cc}_n(\rho).$$

Define a monomial in n variables for each $\rho \in G$, cycle index monomial for ρ, by

$$\mathsf{Z}_\rho(x_1, \ldots, x_n) := x_1^{\mathsf{cc}_1(\rho)} x_2^{\mathsf{cc}_2(\rho)} \ldots x_n^{\mathsf{cc}_n(\rho)}$$

and the cycle index polynomial to be the sum of all these:

$$\mathsf{Z}_G(\boldsymbol{x}) := \frac{1}{|G|} \sum_{\rho \in G} \mathsf{Z}_\rho(\boldsymbol{x})$$

Corollary

The number of distinct orbits is $N = \mathsf{Z}_G(m, m, \ldots, m)$.

Why should it be any better to write the polynomial

$$x_1^{\mathsf{cc}_1(\rho)} x_2^{\mathsf{cc}_2(\rho)} \dots x_n^{\mathsf{cc}_n(\rho)}$$

rather than the more pedestrian vector

$$(\mathsf{cc}_1(\rho), \mathsf{cc}_2(\rho), \dots, \mathsf{cc}_n(\rho))$$

The same information is conveyed both ways, we can easily translate back and forth.

True, but polynomials come equipped with a number of operations: addition, multiplication, substitution of integers, substitution of other polynomials.

This is the whole idea behind generating functions. Ours here are finite (polynomials), but they are just as useful as their infinite cousins.

For example, for $G = D_4$ we have

$$\mathsf{Z}_G(\boldsymbol{x}) = \frac{1}{8}(x_1^4 + 2x_1^2x_2 + 3x_2^2 + 2x_4).$$

Hence

$$N = \mathsf{Z}_G(\boldsymbol{m}) = \frac{1}{8}(m^4 + 2m^3 + 3m^2 + 2m).$$

Addition, multiplication and substitution $x_i \mapsto m$ all conspire to produce the right value.

Not bad, but we still can't quite handle the Tic-Tac-Toe problem in style.

For $f \in X$ let

$$\mathsf{dc}_i(f) := \text{ number of vertices } v \text{ such that } f(v) = c_i$$

Note that $\sum_i \mathsf{dc}_i(f) = n$.

Intuitively, the weight of a configuration is given by the vector

$$(\mathsf{dc}_1(f), \mathsf{dc}_2(f), \ldots, \mathsf{dc}_m(f)) \in \mathbb{N}^m$$

and provides full information about the number of vertices of each color.

Weights are an invariant in the sense that all elements in an orbit of G have the same weight: the permutations can move the colors around, but they cannot change the color counts. So we can talk about the weight of an orbit.

Again, it turns out to be more convenient to define the weight of a coloring f by the monomial

$$\mathsf{weight}(f) := c_1^{\mathsf{dc}_1(f)} c_2^{\mathsf{dc}_2(f)} \dots c_m^{\mathsf{dc}_m(f)}$$

Note that $\mathsf{weight}(f)$ is a "purely formal expression", the c_i are just colors.

More precisely, $\mathsf{weight}(f)$ is a an element of the polynomial ring $\mathbb{Z}[c_1, \dots, c_m]$.

Generating functions are your friend.

We will see shortly that this trick makes it easy to determine the number of orbits of a given weight.

Suppose we have $n = 8$, $\rho = (1)(2)(3)(4)(56)(78)$ where we have written the 1-cycles explicitly.

Then $\mathsf{Z}_\rho(\boldsymbol{x}) = x_1^4 x_2^2$.

Which configurations over three colors r, g, b are invariant under ρ?

All the cycles must be mono-chromatic, so the 2-cycles use up two occurrences of a color. Think of $+$ as logical or (choice), and $\cdot$ as logical and. Then we have the possibilities

$$(r+g+b)(r+g+b)(r+g+b)(r+g+b)(r^2+g^2+b^2)(r^2+g^2+b^2) = (r+g+b)^4(r^2+g^2+b^2)^2$$

Major Insight: this is just $\mathsf{Z}_\rho(r+g+b, r^2+g^2+b^2)$.

By expanding this polynomial we get a huge expression.

First assume the variables r, g, b to be non-commuting. Then there are 729 terms.

If we allow commutation, and collect monomials we get 45 terms, each of degree 8.

$$b^8 + 4\,b^7\,g + 8\,b^6\,g^2 + 12\,b^5\,g^3 + 14\,b^4\,g^4 + 12\,b^3\,g^5 + 8\,b^2\,g^6 + 4\,b\,g^7 + g^8 + 4\,b^7\,r +$$
$$12\,b^6\,g\,r + 20\,b^5\,g^2\,r + 28\,b^4\,g^3\,r + 28\,b^3\,g^4\,r + 20\,b^2\,g^5\,r + 12\,b\,g^6\,r + 4\,g^7\,r +$$
$$8\,b^6\,r^2 + 20\,b^5\,g\,r^2 + 32\,b^4\,g^2\,r^2 + 40\,b^3\,g^3\,r^2 + 32\,b^2\,g^4\,r^2 + 20\,b\,g^5\,r^2 + 8\,g^6\,r^2 +$$
$$12\,b^5\,r^3 + 28\,b^4\,g\,r^3 + 40\,b^3\,g^2\,r^3 + 40\,b^2\,g^3\,r^3 + 28\,b\,g^4\,r^3 + 12\,g^5\,r^3 + 14\,b^4\,r^4 +$$
$$28\,b^3\,g\,r^4 + 32\,b^2\,g^2\,r^4 + 28\,b\,g^3\,r^4 + 14\,g^4\,r^4 + 12\,b^3\,r^5 + 20\,b^2\,g\,r^5 + 20\,b\,g^2\,r^5 +$$
$$12\,g^3\,r^5 + 8\,b^2\,r^6 + 12\,b\,g\,r^6 + 8\,g^2\,r^6 + 4\,b\,r^7 + 4\,g\,r^7 + r^8$$

Consider the term $32\, r^2\, g^2\, b^4$.

The 32 choices for an invariant configuration of weight $r^2\, g^2\, b^4$ are

$$rrggbbbb, rrbbggbb, rrbbbbgg, rgrgbbbb, rggrbbbb, rbrbggbb, rbrbbbgg, rbbrggbb,$$
$$rbbrbbgg, grrgbbbb, grgrbbbb, ggrrbbbb, ggbbrrbb, ggbbbbrr, gbgbrrbb, gbgbbbrr,$$
$$gbbgrrbb, gbbgbbrr, brrbggbb, brrbbbgg, brbrggbb, brbrbbgg, bggbrrbb, bggbbbrr,$$
$$bgbgrrbb, bgbgbbrr, bbrrggbb, bbrrbbgg, bbggrrbb, bbggbbrr, bbbbrrgg, bbbbggrr$$

These are precisely the words w over alphabet $\{r, g, b\}$ with Parikh vector $(2, 2, 4)$ subject to the constraint $w_5 = w_6$ and $w_7 = w_8$.

This all hinges on the fact that we can express logic by algebra: we really need to consider choices of colors on the cycles of a given permutation.

But we can translate this counting problem into polynomial arithmetic by introducing formal variables for the colors and then working in the polynomial ring $\mathbb{Z}[r, g, b]$. It is quite surprising how much mileage one can get out of generating functions.

Since we can perform the arithmetic operations easily, this translation allows for easy computation – sort of, a computer algebra system might come in handy.

Here is a more general description of this method.

Let

$$N(a_1, a_2, \ldots, a_m) := \#\text{ orbits with weight } c_1^{a_1} c_2^{a_2} \ldots c_m^{a_m}.$$

and define the pattern inventory of G on X to be

$$\sum_{\boldsymbol{a}} N(a_1, a_2, \ldots, a_m)\, c_1^{a_1} c_2^{a_2} \ldots c_m^{a_m}$$

This is a (typically huge) polynomial in variables $c_1, \ldots, c_m$ whose coefficients contain exactly the counting information we are after.

Nice, but utterly useless unless we can somehow compute the pattern inventory (without resorting to brute force, of course).

Theorem (Pólya-Redfield)

Set $y_i = c_1^i + c_2^i + \ldots + c_m^i$. *Then the pattern inventory is* $\mathsf{Z}_G(y_1, y_2, \ldots, y_n)$.

Likewise, $\mathsf{Z}_\rho(y_1, y_2, \ldots, y_n)$ is the generating function for the number of patterns of the indicated weight, invariant under ρ.

It may still seem rather difficult to compute the pattern inventory, but with a little computer algebra it is not too hard: we need to deal with permutation groups and some polynomial algebra.

First we compute all the 8 monomials.

ρ	Z_ρ	ρ	Z_ρ
1	x_1^4	β	x_2^2
α	x_4	$\alpha\beta$	$x_1^2x_2$
α^2	x_2^2	$\alpha^2\beta$	x_2^2
α^3	x_4	$\alpha^3\beta$	$x_1^2x_2$

Note that this depends solely on the group $G \subseteq \mathfrak{S}_n$ acting on the colorings, not on the colors themselves.

Hence we have to compute this only once.

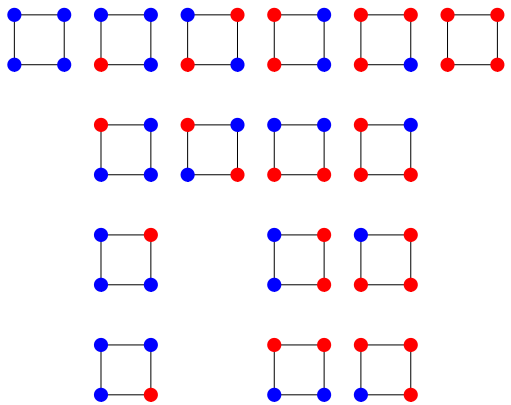
Assume $m = 2$.

$$\mathsf{Z}_{\alpha}(\boldsymbol{y}) = c_1^4 + c_2^4$$
$$\mathsf{Z}_{\alpha^2}(\boldsymbol{y}) = c_1^4 + 2\,c_1^2\,c_2^2 + c_2^4$$
$$\mathsf{Z}_{\alpha^3\beta}(\boldsymbol{y}) = c_1^4 + 2\,c_1^3\,c_2 + 2\,c_1^2\,c_2^2 + 2\,c_1\,c_2^3 + c_2^4$$

Lastly, here is the pattern inventory:

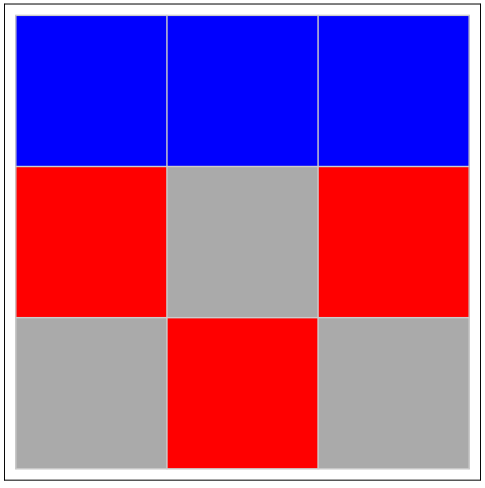
$$\mathsf{Z}_{G}(\boldsymbol{y}) = c_1^4 + c_1^3\,c_2 + 2\,c_1^2\,c_2^2 + c_1\,c_2^3 + c_2^4$$

There are 6 orbits, two of weight $c_1^2\,c_2^2$.



For the Tic-Tac-Toe problem we have $|V| = 9$ and $|C| = 3$.

As we have seen, the group G acting on the board is isomorphic to D_4 but has degree 9.



Thus, the cycle decompositions for the permutations in G are quite different from the previous examples and the CIP is:

$$\frac{1}{8}(x_1^9 + 2x_1x_4^2 + x_1x_2^4 + 4x_1^3x_2^3)$$

- Pólya-Redfield

2. Lamplighters

Most of the groups we have used in Pólya counting so far are fairly straightforward: they correspond directly to geometric symmetries. Of course, things get messier in higher dimensions, but still.

This might create the impression that the groups associated with actions on combinatorial objects are always easy to understand. Not so. Sometimes it requires quite a bit of effort to come up with the right group that describes some (intuitively clear) action.

Here is a notorious example.

Suppose you have a ring of n lamps; each lamp is either on or off.

There is an eponymous lamplighter, who walks around and turns lights on and off.

The lamplighter can perform two atomic actions:

α	move to the next lamp, or
τ	toggle the state of the current lamp.

The actions are clearly reversible, so there must be a group plus action hiding somewhere.

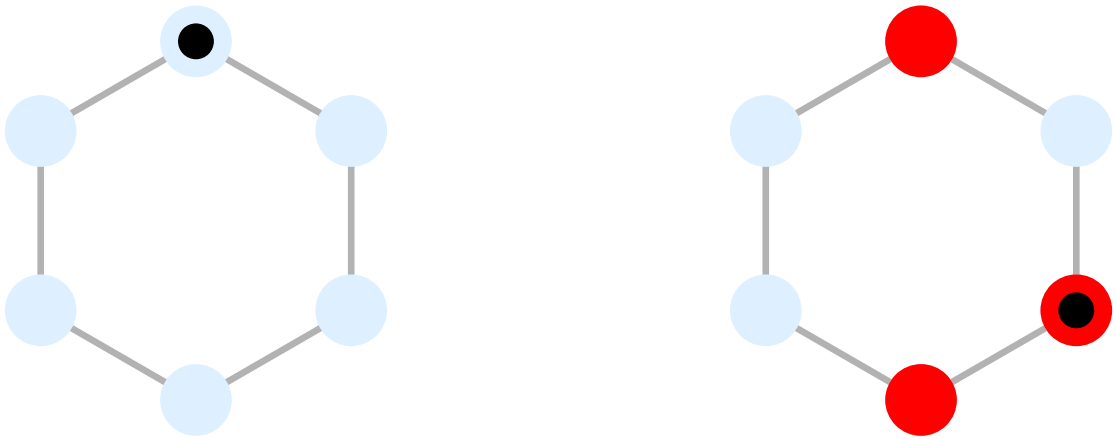
It is obvious that $\alpha^n = 1$ and $\tau^2 = 1$.

The group does not commute, $\alpha\tau \neq \tau\alpha$ (assuming $n > 1$).

But what exactly is the group, and how does it act?

Clearly we can describe the space of configurations as

$$X = \mathbf{2}^n \times (n)$$



Configurations $(\mathbf{0}, 0)$ and $(101100, 2)$.

We need the group G generated by α and τ, something like

$$\langle \alpha, \tau \mid \alpha^n = 1, \tau^2 = 1, \dots \rangle$$

but we don't have all the necessary identities for this kind of description. For example, it is true that

$$(\tau\alpha\tau\alpha^{n-1})^2 = 1$$

It seem like a good idea to try to write the group elements in two parts:

- part 1 describes the changes in lights (toggle pattern), and
- part 2 describes the movement of the lamplighter.

So G should be a product, something like $G = A \times B$.

Traditionally, one considers a bi-infinite row of lamps rather than a finite circle. So the configurations are

$$X = \prod_{\mathbb{Z}} \mathbf{2} \times \mathbb{Z}$$

If only finitely many lamps are ever lit (which is automatically the case if we start with all lights off), we get

$$X = \coprod_{\mathbb{Z}} \mathbf{2} \times \mathbb{Z}$$

So the space of configurations is fairly straightforward in all three cases, but we need to figure out what exactly the group is.

Recall our general strategy: we want to combine the atomic operations and their inverses in arbitrary ways. For example, there has to be a group element for "toggle 0 and 1" (coordinates are relative to the lamplighter's position). and so forth. And the group operation applied to "toggle 0 and 1" and "toggle 1 and 2" should be "toggle 0 and 2".

So one would expect something like

$$G = \mathbf{2}^{\mathbb{Z}} \times \mathbb{Z}$$

The first component keeps track of toggles, the second of the lamplighter's position.

$(\boldsymbol{a}, s)$ should act on $(\boldsymbol{x}, p)$ as follows: flip the lights in $\boldsymbol{x}$ according to $\boldsymbol{a}$, then change p according to s.

Let's go back to the finite case. Let $\boldsymbol{e_i} \in \mathbf{2}^n$ be the unit vector with the 1 in position $i \in (n)$, so e.g. $\boldsymbol{e_3} = 00010000$.

Then we would like to have the following interpretation:

$\tau \rightsquigarrow (\boldsymbol{e}_0, 0)$: toggle the lamp at the current position.

$\alpha \rightsquigarrow (\mathbf{0}, 1)$: the lamplighter moves forward by one.

And $(\boldsymbol{e}_0 \oplus \boldsymbol{e}_1, 3)$ means: toggle the lights in relative positions 0 and 1, then move forward by 3 places, corresponding to $\tau\alpha\tau\alpha^2$.

Does $G = \mathbf{2}^n \times (n)$ properly reflect this intuition?

Close, but no cigar: the componentwise multiplication does not quite work.

Consider the group element $\tau\alpha\tau\alpha$.

According to the intuitive model we should have

$$(\boldsymbol{e}_0, 1)(\boldsymbol{e}_0, 1) = (\boldsymbol{e}_0 \oplus \boldsymbol{e}_1, 2)$$

Instead we get

$$(\boldsymbol{e}_0, 1)(\boldsymbol{e}_0, 1) = (\boldsymbol{e}_0 \oplus \boldsymbol{e}_0, 2) = (\mathbf{0}, 2)$$

where $\oplus$ stands for bit-wise addition modulo 2.

Our model thinks that $\tau\alpha\tau\alpha = \alpha^2$, which is clearly false. The position is right, but not the toggle pattern: simply adding the vectors does not reflect the change in position.

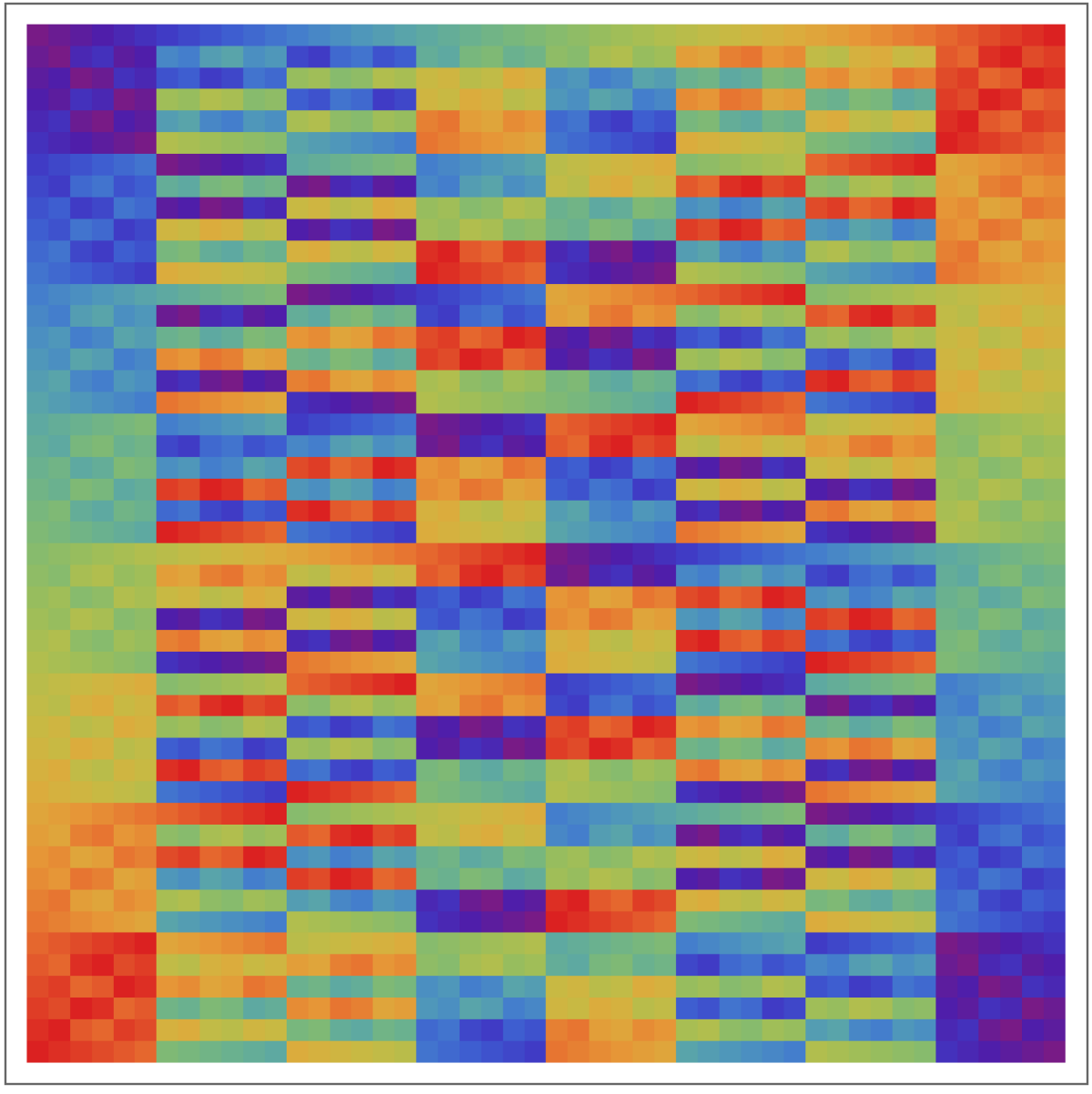
But we already know how to fix this: we need to construct a semidirect or wreath product.

For simplicity write $\mathbf{2}$ for the additive group $\mathbb{Z}_2$, which happens to be the symmetric group on two letters. The lamplighter group on a ring of size n is given by the wreath product

$$G = \mathbf{2} \wr \mathbb{Z}_n = \mathbf{2}^n \rtimes \mathbb{Z}_n$$

A modular number s acts on $\boldsymbol{a}$ by shifting the sequence by s places. Writing σ for the cyclic shift we have

$$(\boldsymbol{a}, r)(\boldsymbol{b}, s) = (\boldsymbol{a} \oplus \sigma^r(\boldsymbol{b}), r + s)$$



The infinite lamplighter group is given by the restricted wreath product

$$G = \mathbf{2} \wr \mathbb{Z} = \mathbf{2}_0^{\mathbb{Z}} \rtimes \mathbb{Z}$$

We are dealing with bi-infinite bitvectors with finite support, the coproduct, here written $\mathbf{2}_0^{\mathbb{Z}}$.

As in the finite case, this group has two generators τ and α corresponding to "toggle current" and "move right by 1" subject to identities

$$(\tau\alpha^i\tau\alpha^{-i})^2 = 1$$

where $i \in \mathbb{Z}$. Note that there are infinitely many identities; one can show that the lamplighter group has no finite presentation.

Exercise

Verify $(\alpha\tau^i\alpha\tau^{-i})^2 = 1$.

Exercise

Interpret $(\alpha\tau^i\alpha\tau^{-i})^2 = 1$ *geometrically.*

Exercise

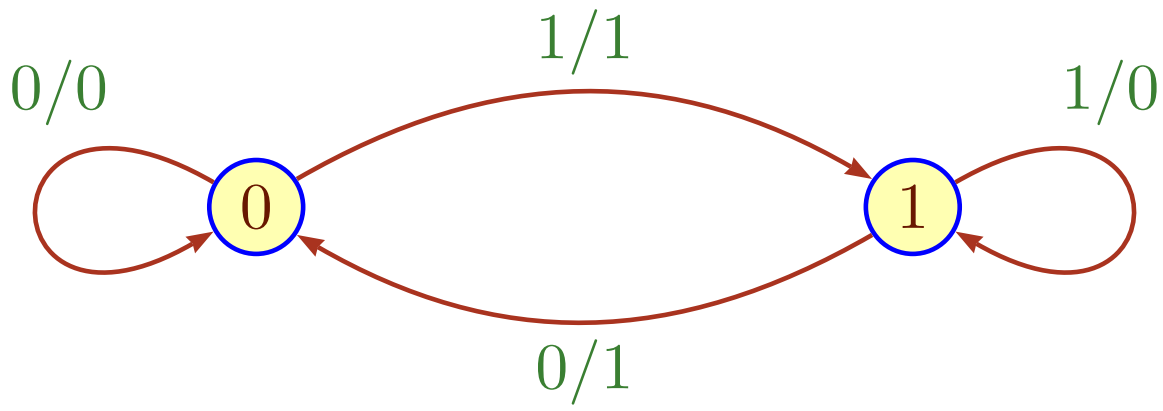
What would happen if we were to use the "move first, then toggle" interpretation?

We could also consider the full product A^B instead of the coproduct to get a larger version of the lamplighter group where infinitely many lights may be on (with the old one naturally forming a subgroup).

At any rate, the Lamplighter group has lots of interesting properties and has been studied extensively; Google Scholar shows $15,900$ hits.

One fun fact: the lamplighter group is closely related to finite state machines.

Consider the following 2-state synchronous transducer:



Note that state 0 copies the next bit while state 1 flips it.

It is easy to see that picking one of the states k as initial state defines a permutation $\underline{k}$ of all binary words. Moreover, the permutation is length and prefix preserving; it can be pictured as an automorphism of the infinite binary tree.

The automaton is just an elegant description of a recursive system of equations:

$$\underline{0}(ax) = a\,\underline{a}(x)$$

$$\underline{1}(ax) = \overline{a}\,\underline{a}(x)$$

But then we can consider the group G generated by the permutations $\underline{0}$ and $\underline{1}$, a subgroup of the group of all automorphism of the binary tree.

Surprise: G is isomorphic to the lamplighter group.

Let us revisit the problem of counting Boolean functions modulo equivalence. We know how to handle two cases:

- Permutation of variables: $\mathfrak{S}_n$ acting on $\mathbf{2}^n$
- Negation of variables: $\mathbb{B}_n$ acting on $\mathbf{2}^n$

It is tempting to ask how to deal with permutation and negation together.

For example

$$x \wedge (y \vee z) \text{ and } z \wedge (\overline{x} \vee y)$$

are in a sense the same Boolean function: naming of variables does not really matter and exchanging x and $\overline{x}$ does not matter either.

We refer to two Boolean functions that can be obtained from each other by permuting and/or negating variables as PN-equivalent.

Counting the number of PN-equivalence classes should be a problem that falls well within the reach of the counting methods we have just seen.

It is even clear that $V = \mathbf{2}^n$ and $C = \mathbf{2}$ is the right framework.

But what is the group G acting on V in this case?

No doubt both $\mathfrak{S}_n$ and $\mathbb{B}_n$ are involved, but exactly how?

Note that we can figure out the cardinality of G: $2^n \cdot n!$

A not unreasonable first guess might be that

$$G = \mathbb{B}_n \times \mathfrak{S}_n$$

is the right group; after all, the pieces make sense and we get the right size.

Alas, that's plain wrong. To see why, consider $g = (e_1, (12)) \in G$. Informally, g means "swap the first two variables and negate the first."

The effect of g^2 ought to be something like $x\, y\, z \ldots \mapsto \overline{x}\, \overline{y}\, z \ldots$

However, in G we have $g^2 = 1$.

Both in the full and restricted wreath product we considered so far, the group B acts on sequences, $\prod_B A$ or $\coprod_B A$.

But we also could have B acting on some set X and then consider sequences in $\prod_X A$ or $\coprod_X A$.

The definitions don't change at all: we can still "multiply" a sequence index with a group element in B.

That's all we need!

One writes $A \wr_X B$ for this version of the wreath product.

The problem is again that the ordinary Cartesian product simply fails here, we need something slightly more complicated: the (generalized) wreath product $G = \mathfrak{S}_2 \wr_{[n]} \mathfrak{S}_n$. Writing the operations additively and multiplicatively, we have in G:

$$(\boldsymbol{x}, f)(\boldsymbol{y}, g) = (\boldsymbol{x} + f * \boldsymbol{y}, fg)$$

where

$$f * \boldsymbol{y} = (y_{f(1)}, \ldots, y_{f(n)}).$$

is the vanilla left action.

If you find this description of G a bit obscure, think of permutation matrices instead: a permutation matrix of order n is 0/1-matrix of size n by n such that every column and every row contains exactly a single 1.

It is not hard to see that these matrices correspond precisely to permutations of $[n]$. Under ordinary matrix multiplication, they produce a group that is isomorphic to $\mathfrak{S}_n$.

To represent $G = \mathfrak{S}_2 \wr_{[n]} \mathfrak{S}_n$ we need to use signed permutation matrices: the entries now can be $+1$ or -1.

Again, with ordinary matrix multiplication, we obtain a group isomorphic to G. Note that the cardinalities match up.

Exercise

Verify the last claim.

Still, for very small n we can compute G directly and determine its cycle index polynomial. For example, for $n = 5$ we get

$$\frac{1}{3840}\left(x_1^{32} + 20x_1^{16}x_2^8 + 60x_1^8x_2^{12} + 231x_2^{16} + 80x_1^8x_3^8 + 240x_1^4x_2^2x_4^6 + 240x_2^4x_4^6 + 520x_4^8 + 384x_1^2x_5^6 + 160x_1^4x_2^2x_3^4x_6^2 + 720x_2^4x_6^4 + 480x_8^4 + 384x_2x_{10}^3 + 320x_4^2x_{12}^2\right)$$

The number of PN-equivalent Boolean functions of 5 variables is therefore

$$1,228,158$$

as opposed to $4,294,967,296$ functions in the set-theoretic sense.

A brute force computation of CIPs as on the last slide is not really the right answer: we should try to compute the CIP for G given the CIPs for $\mathbb{B}_n$ and $\mathfrak{S}_n$.

The good news is: this can be done and produces a computationally satisfactory answer.

The bad new is: the calculation is quite complicated and would lead us too far astray.