2016

Symmetric Colorings of the Hypercube and Hyperoctahedron

Bo Phillips
Wright State University

SYMMETRIC COLORINGS OF THE HYPERCUBE AND HYPEROCTAHEDRON

A thesis submitted in partial fulfillment of
the requirements for the degree of
Master of Science

By

BO QUILLEN PHILLIPS
B.S., Wright State University, 2015

2016
Wright State University

WRIGHT STATE UNIVERSITY

GRADUATE SCHOOL

April 30, 2016

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Bo Quillen Phillips ENTITLED Symmetric Colorings of the Hypercube and Hyperoctahedron BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science.

Daniel Slilaty, Ph.D.
Thesis Director

Ayse Sahin, Ph.D.
Chair, Department of
Math and Statistics

Committee on
Final Examination

Yuqing Chen, Ph.D.

Xiangqian Zhou, Ph.D.

Robert E.W. Fyffe, Ph.D.
Vice President for Research and
Dean of the Graduate School

ABSTRACT

Phillips, Bo Quillen. M.S. Department of Math and Statistics, Wright State University, 2016. Symmetric Colorings of the Hypercube and Hyperoctahedron.

A *self-complementary* graph G is a subgraph of the complete graph K_n that is isomorphic to its complement. A self-complementary graph can be thought of as an edge 2-coloring of K_n that admits a color-switching automorphism. An automorphism of K_n that is color-switching for some edge 2-coloring is called a *complementing automorphism*. Complementing automorphisms for K_n have been characterized in the past by such authors as Sachs and Ringel.

We are interested in extending this notion of self-complementary to other highly symmetric families of graphs; namely, the hypercube Q_n and its dual graph, the hyperoctahedron O_n. To that end, we develop a characterization of the automorphism group of these graphs and use it to prove necessary and sufficient conditions for an automorphism to be complementing. Finally, we use these theorems to construct a computer search algorithm which finds all self-complementary graphs in Q_n and O_n up to isomorphism for $n = 2, 3, 4$.

Contents

List of Figures

List of Tables

1 Introduction

A *self-complementary* graph G is a subgraph of K_n that is isomorphic to its complement. So a self-complementary graph can be thought of as an edge 2-coloring of K_n that admits a color-switching automorphism. An automorphism of K_n that is color-switching for some edge 2-coloring is called a *complementing automorphism*.

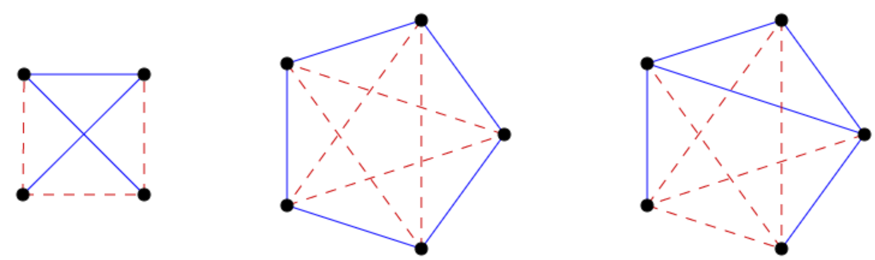

Figure 1: The self-complementary subgraphs of K_4 and K_5

A first fundamental arithmetic result for self-complementary graphs is Proposition 1.1, observed independently by Sachs [4] and Ringel [3].

Proposition 1.1 *If K_n has a complementing automorphism, then $n = 4k$ or $n = 4k + 1$.*

Proof: Note that the number of edges in K_n is $\frac{n(n-1)}{2}$. Thus the result follows from the fact that this number must be even. $\square$

The automorphism group of the complete graph K_n is the symmetric group S_n where $\{1, ..., n\}$ is the set of vertices of K_n. A characterization of complementing automorphisms of K_n is, again, due to Sachs [4] and Ringel [3].

Theorem 1.2 *If $n = 4k$, then σ is a complementary automorphism of some edge 2-coloring of K_n iff σ has cycle structure in which each cycle has length divisible by*

4. If $n = 4k + 1$, then σ is a complementary automorphism of K_n iff σ has cycle structure in which one cycle has length 1 and all other cycles have length divisible by 4.

Theorem 1.2 can be the basis for algorithms to compute all self-complementary graphs for a fixed n. Other algorithms have been used by McNally and Molina [2] to catalog self-complementary of order up to $n = 13$. We are interested in edge 2-colorings of other highly symmetric graphs that admit a color-switching automorphism; again we will call such an automorphism a complementing automorphism. In this thesis, we will achieve an analog of Theorem 1.2 for the hypercube Q_n and its dual the hyperoctahedron O_n. We will then use these characterization to compute all symmetric edge 2-colorings for $n = 2, 3, 4$.

2 Preliminaries

We use Q_n to denote the n-dimensional hypercube, and O_n to denote its dual, the n-dimensional hyperoctahedron.

We use the term *j-face* to refer to a j-dimensional element of Q_n or O_n. For example in Q_4: 0-faces are vertices, 1-faces are edges, 2-faces are square faces, and 3-faces are cubical cells. Note that a j-face in Q_n corresponds to an $(n - j - 1)$-face in O_n and vice-versa.

For simplicity, we use the terms *vertices* and *edges* to refer to 0-faces and 1-faces respectively. Furthermore, we use the terms *facets* and *ridges* to denote respectively $(n - 1)$-faces and $(n - 2)$-faces (which are simply the vertices and edges respectively in the dual graph).

We denote the set of all j-face elements in Q_n by $F_j(Q_n)$ (likewise $F_j(O_n)$ for O_n). For the sake of clarity, we use $V(Q_n)$ and $E(Q_n)$ to denote the set of all vertices and edges respectively of Q_n (and similarly, $V(O_n)$ and $E(O_n)$ for those in O_n).

To represent the vertex elements of Q_n and O_n, we consider them as points embedded in n-dimensional Euclidean space. More precisely, we let $V(Q_n)$ be the set of all $n \times 1$ vectors whose entries are from $\{-1, 1\}$, and we let $V(O_n)$ be the set of all $n \times 1$ vectors with exactly one nonzero entry from $\{-1, 1\}$.

For example, represent the vertices of Q_3 with the following vectors:

$$1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad 2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} \qquad 3 = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \qquad 4 = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$$

$$5 = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix} \qquad 6 = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \qquad 7 = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \qquad 8 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

To represent the edge set $E(Q_3)$, notice that an edge in Q_3 is just a line segment in $\mathbb{R}^3$ connecting two appropriate vertices. This conveniently allows us to represent the elements in $E(Q_3)$ as just the midpoints of the corresponding line segments:

$$12 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \qquad 24 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \qquad 34 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \qquad 13 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$15 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \qquad 26 = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \qquad 48 = \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix} \qquad 37 = \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix}$$

$$56 = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} \qquad 68 = \begin{bmatrix} -1 \\ 0 \\ -1 \end{bmatrix} \qquad 78 = \begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix} \qquad 57 = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

This idea of midpoints can be naturally extended to higher-dimensional elements. Thus, the square faces of Q_3 (which are just the vertices of O_3) are given as follows:

$$1234 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad 5678 = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \qquad 1256 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$3478 = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \qquad 1357 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad 2468 = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$$

In general, the j-face elements of Q_n can be represented as those $n \times 1$ vectors whose entries are from $\{-1, 0, 1\}$ and contain exactly j zero entries. In particular, the vertices of O_n are just the positive and negative vectors of the standard basis of $\mathbb{R}^n$. From this representation, it is easily verified through combinatorial argument that Q_n contains $2^{n-j}\binom{n}{j}$ j-face elements.

n	Q_n	Vertices	Edges	Faces	Cells	4-Faces	5-Faces	6-Faces
0	Point	1						
1	Line Segment	2	1					
2	Square	4	4	1				
3	Cube	8	12	6	1			
4	Tesseract	16	32	24	8	1		
5	Pentaract	32	80	80	40	10	1	
6	Hexaract	64	192	240	160	60	12	1

Table 1: Number of j-face Elements of Q_n

Because of their dual nature, the set of symmetries (i.e. graph automorphisms) of Q_n is the same as the set of symmetries of O_n. These symmetries form a group under composition called the *hyperoctahedral group*, denoted $\mathcal{H}_n$. From the above description, it's obvious that any automorphism of O_n simply permutes coordinate axes of $\mathbb{R}^n$ and possibly reflects along certain axes. This allows us to write the elements $\mathcal{H}_n$ as those $n \times n$ matrices with entries from $\{-1, 0, 1\}$ and exactly one nonzero entry in each row and column; these are called signed permutation matrices. In this context, we define the action of a symmetry in $\mathcal{H}_n$ with matrix representation M on any j-face element v in Q_n or O_n as simply the product Mv.

3 Signed Permutations

It is well-known that a permutation σ on n objects can be represented as an $n \times n$ matrix:

$$\sigma = \begin{pmatrix} 1 & \dots & n \\ \\ \sigma(1) & \dots & \sigma(n) \end{pmatrix} \equiv \begin{bmatrix} s_{ij} \end{bmatrix} = M$$

where

$$s_{ij} = \begin{cases} 1, & \sigma(j) = i \\ 0, & \text{else} \end{cases}$$

Such a matrix is called a *permutation matrix*. These matrices have many properties in common with those in $\mathcal{H}_n$; namely, they are invertible and have exactly one nonzero entry per row and column. In fact, one can easily verify that the set of $n \times n$ permutation matrices is a subset of $\mathcal{H}_n$. This allows us to naturally extend the above isomorphism to any matrix $M = [s_{ij}]$ in $\mathcal{H}_n$:

$$M = \left[s_{ij} \right] \equiv \begin{pmatrix} 1 & ... & n \\ \star_1(\sigma) & ... & \star_n(\sigma) \\ \underline{\sigma}(1) & ... & \underline{\sigma}(n) \end{pmatrix} = \sigma$$

where

$$s_{ij} = \begin{cases} \star_i(\sigma), & \underline{\sigma}(j) = i \\ 0, & \text{otherwise} \end{cases}$$

Such a permutation-like object is called a *signed permutation*, and acts on a set just like an ordinary permutation, except it associates a positive or negative sign $\star_i$ to the image of each object i. Clearly then, a signed permutation σ without the sign association is just an ordinary permutation $\underline{\sigma}$ called the *underlying permutation.* The action of a signed permutation σ on $\vec{x} \in F_j(Q_n)$ (and by extension, $F_{n-j-1}(O_n)$), without referring to its matrix form, is that for any vector $\vec{x} = [x_1, ..., x_n] \in F_j(Q_n)$, we get that $\sigma([x_1, ..., x_n]) = [y_1, ..., y_n]$ where

$$y_i = \star_i(\sigma) x_{\underline{\sigma}^{-1}(i)}$$

Like an ordinary permutation, a signed permutation σ can be decomposed cyclically. The cycles of this decomposition are the cycles of the underlying permutation $\underline{\sigma}$. The sign mark placed above the symbol k is $\star_k(\sigma)$. For example

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ + & - & + & + & + & - \\ 2 & 5 & 4 & 3 & 1 & 6 \end{pmatrix} \equiv (\overset{+}{1}\overset{-}{2}\overset{+}{5})(\overset{+}{3}\overset{+}{4})(\overset{-}{6})$$

We call this the decomposition of σ into *signed cycles*. Because of their sign markings, 1-cycles are not omitted in notation unlike ordinary permutations. A signed permutation whose underlying permutation is a cycle is itself called a *signed cycle*. A signed cycle σ is said to be *sign-switching* if the product of its signs is negative. Otherwise, it is called *sign-preserving*.

For a signed cycle σ of length n, we use the notation n^+ to describe σ if it is sign-preserving; similarly, we use n^- if it is sign-switching. Such a representation is called the *signed-cycle type* or *symmetry type* of σ denoted $\chi(\sigma)$. For a signed permutation $\sigma = \tau_1 ... \tau_k$ that is expressed as a product of multiple disjoint signed cycles, we denote the signed-cycle type of σ by the product of signed-cycle types of $\tau_1, ..., \tau_k$. That is, $\chi(\sigma) = \chi(\tau_1)...\chi(\tau_k)$.

For example, let $\sigma = (\overset{+}{1}\overset{-}{2}\overset{+}{5})(\overset{+}{3}\overset{+}{4})(\overset{-}{6})$ be as above. Then $\chi(\sigma) = 1^- 2^+ 3^-$.

We define the *symmetry class* of σ by $\mathrm{Cl}(\sigma) = \{\tau \in \mathcal{H}_n : \chi(\tau) = \chi(\sigma)\}$. If a given symmetry type χ is clear from context, we may denote the corresponding symmetry class by $\mathrm{Cl}(\chi)$. The following theorem is from Kerber [1].

Theorem 3.1 *The conjugacy classes of $\mathcal{H}_n$ are the symmetry classes of $\mathcal{H}_n$. That is, $\sigma, \tau \in \mathcal{H}_n$ are conjugates if and only if $\chi(\sigma) = \chi(\tau)$.*

Proposition 3.2 *Let $\sigma \in \mathcal{H}_n$ be a cyclic signed permutation, and $\vec{x} = [x_1, ..., x_n]$ be a j-face of Q_n or O_n. Then:*

$$\sigma^n(\vec{x}) = \begin{cases} \vec{x}, & \chi(\sigma) = n^+ \\ -\vec{x}, & \chi(\sigma) = n^- \end{cases}$$

Proof: Since σ is an n-cycle, we have that $\sigma^n(x_i) = Sx_i$, where S is the product of the sign markings of σ. Thus the result follows from the definitions of sign-preserving and sign-switching permutations. $\square$

For the j-faces of Q_n, each $\sigma \in \mathcal{H}_n$ induces an ordinary permutation on these j-faces. If $\chi(\sigma) = \chi_0$, we denote the cycle structure of the ordinary permutation on the j-faces induced by σ by $\kappa_j(\sigma)$ or $\kappa_j(\chi_0)$. In transcribing the cycle structure of σ, we use $\kappa_j(\sigma) = \sum_{i=1}^{k} a_i \times b_i$ to mean a_1 cycles of length b_1, a_2 cycles of length b_2, etc. For example, the table below describes the cycle structure of the vertices and edges of Q_2 for each of the conjugacy classes of its symmetries.

Type $\chi(\sigma)$	$\lvert\mathrm{Cl}(\sigma)\rvert$	Vertex Cycles $\kappa_0(\sigma)$	Edge Cycles $\kappa_1(\sigma)$
1^+1^+	1	4×1	4×1
1^+1^-	2	2×2	$2 \times 1 + 1 \times 2$
1^-1^-	1	2×2	2×2
2^+	2	$2 \times 1 + 1 \times 2$	2×2
2^-	2	1×4	1×4

Table 2: Symmetries and Cycle Structures of Q_2

Type $\chi(\sigma)$	$\lvert\mathrm{Cl}(\sigma)\rvert$	Vertex Cycles $\kappa_0(\sigma)$	Edge Cycles $\kappa_1(\sigma)$	Face Cycles $\kappa_2(\sigma)$
$1^+1^+1^+$	1	8×1	12×1	6×1
$1^+1^+1^-$	3	4×2	$4 \times 1 + 4 \times 2$	$4 \times 1 + 1 \times 2$
$1^+1^-1^-$	3	4×2	6×2	$2 \times 1 + 2 \times 2$
$1^-1^-1^-$	1	4×2	6×2	3×2
1^+2^+	6	$4 \times 1 + 2 \times 2$	$2 \times 1 + 5 \times 2$	$2 \times 1 + 2 \times 2$
1^+2^-	6	2×4	3×4	$2 \times 1 + 1 \times 4$
1^-2^+	6	4×2	$2 \times 1 + 5 \times 2$	3×2
1^-2^-	6	2×4	3×4	$1 \times 2 + 1 \times 4$
3^+	8	$2 \times 1 + 2 \times 3$	4×3	2×3
3^-	8	$1 \times 2 + 1 \times 6$	2×6	1×6

Table 3: Symmetries and Cycle Structures of Q_3

Note that $\kappa_j(\sigma)$ is well-defined only for $0 \leqslant j < n$. For theoretical purposes, we define $\kappa_j(\sigma) = 0 \times 1$ whenever $j < 0$ or $j > n$, and $\kappa_j(\sigma) = 1 \times 1$ whenever $j = n$.

Proposition 3.3 *Every sign-preserving cyclic permutation $\sigma \in \mathcal{H}_n$ fixes exactly two vertices of Q_n. Furthermore, these two vertices are antipodal.*

Proof: Let $\sigma \in \mathcal{H}_n$ be a sign-preserving n-cycle. Construct a vector $\vec{x} = [x_1, ..., x_n]$ as follows:

$$x_1 = 1 \qquad \text{and} \qquad x_{\sigma^i(1)} = \prod_{j=1}^{i-1} \star_{\sigma^j(1)}$$

Then since σ is sign-preserving, we have that σ fixes $\vec{x}$ and $-\vec{x}$. Furthermore, it's clear from their construction that these vectors are unique. $\square$

Proposition 3.4 *Every sign-switching cyclic permutation induces no odd length cycles on the j-faces of Q_n for all $j \in \{0, 1, ..., n-1\}$.*

Proof: Assume to the contrary there exist a j-face $\vec{x} \in Q_n$, a sign-switching cycle $\sigma \in \mathcal{H}_n$, and a nonnegative integer k such that $\sigma^{2k+1}(\vec{x}) = \vec{x}$ is minimal with respect to k. Since σ is sign-switching, we have that $\sigma^n(\vec{x}) = -\vec{x}$ and $\sigma^{2n}(\vec{x}) = \vec{x}$. By the Division Algorithm, we have $2n = (2k+1)q + r$ for some $q, r \in \mathbb{Z}$, $0 \leqslant r < 2k+1$. But since k is minimal and $\sigma^r(\vec{x}) = \sigma^r(\sigma^{(2k+1)q}(\vec{x})) = \sigma^{2n}(\vec{x}) = \vec{x}$, we must have $r = 0$. Hence $(2k+1)|(2n)$ from which it follows that $(2k+1)|n$. But this means that $\sigma^n(\vec{x}) = \vec{x}$ which is a contradiction. $\square$

Proposition 3.5 *Let $\sigma \in \mathcal{H}_n$ be a signed cycle of length n. Then*

$$\kappa_{n-1}(\sigma) = \begin{cases} 2 \times n, & \chi(\sigma) = n^+ \\ 1 \times (2n), & \chi(\sigma) = n^- \end{cases}$$

Proof: Let $\vec{x} \in F_{n-1}(Q_n) = F_0(O_n)$ be a vector with unique nonzero entry x_i. Since σ is an n-cycle, we have that $n \in \mathbb{N}$ is minimal such that $x_{\underline{\sigma}^n(i)}$ is nonzero. If $\chi(\sigma) = n^+$, we have that $\sigma^n(\vec{x}) = \vec{x}$. Likewise, if $\chi(\sigma) = n^-$, we have that $\sigma^n(\vec{x}) = -\vec{x}$ and thus $\sigma^{2n}(\vec{x}) = \vec{x}$. $\square$

Proposition 3.6 *Let $\sigma \in \mathcal{H}_n$ be a signed cycle of length n. Then*

$$\kappa_1(\sigma) = \begin{cases} (2^{n-1}) \times n, & \chi(\sigma) = n^+ \\ (2^{n-2}) \times (2n), & \chi(\sigma) = n^- \end{cases}$$

Proof: Let $\vec{x} \in E(Q_n) = F_1(Q_n) = F_{n-2}(O_n)$ be a vector with unique zero entry x_i. Since σ is an n-cycle, we have that $n \in \mathbb{N}$ is minimal such that $x_{\underline{\sigma}^n(i)}$ is zero. If $\chi(\sigma) = n^+$, we have that $\sigma^n(\vec{x}) = \vec{x}$. Likewise, if $\chi(\sigma) = n^-$, we have that $\sigma^n(\vec{x}) = -\vec{x}$ and thus $\sigma^{2n}(\vec{x}) = \vec{x}$. $\square$

Proposition 3.7 *Let $\sigma \in \mathcal{H}_n$ be a cycle of length n. Then*

$$\kappa_{n-2}(\sigma) = \begin{cases} 2 \times (\frac{n}{2}) + (2n-3) \times n, & \chi(\sigma) = n^+, \quad n \textit{ is even} \\ (2n-2) \times n, & \chi(\sigma) = n^+, \quad n \textit{ is odd} \\ (n-1) \times (2n), & \chi(\sigma) = n^- \end{cases}$$

Proof: We consider the following cases:

Case i: $\chi(\sigma) = n^+$, n even

WLOG let $\sigma = (\overset{+}{1}\overset{+}{2}...\overset{+}{n})$, and let $n = 2k$ for some $k \in \mathbb{Z}$. Consider the ridge $\vec{x} \in F_{n-2}(Q_n)$ with entries given as follows:

$$x_i = \begin{cases} 1, & i = 1, k+1 \\ 0, & \text{else} \end{cases}$$

Then we have $\sigma^k(\vec{x}) = \vec{x}$ and $\sigma^k(-\vec{x}) = -\vec{x}$. Hence, we have two induced k-cycles consisting of vectors with two nonzero entries of the same sign and a distance of k entries from each other.

Now let $\vec{y} \in F_{n-2}(Q_n)$ have entries as follows:

$$y_i = \begin{cases} 1, & i = 1 \\ -1, & i = k+1 \\ 0, & \text{else} \end{cases}$$

Then $\sigma^k(\vec{y}) = -\vec{y}$ and $\sigma^k(-\vec{y}) = \vec{y}$. Hence, we have a single induced n-cycle consisting of vectors with two nonzero entries of opposite signs and a distance of k entries from each other.

Finally, let $\vec{z} \in F_{n-2}(Q_n)$ such that $\vec{z}$ does not lie in any of the above induced cycles. That is, the nonzero entries of $\vec{z}$ are not a distance of k entries from each other. Then it follows that $\sigma^i(\vec{z}) \neq \vec{z}$ for all $i \in \{1, ..., n-1\}$, and $\sigma^n(\vec{z}) = \vec{z}$.

Thus, σ induces exactly two cycles of length k, and the remaining cycles are all of length n. Therefore, the result follows from $|F_{n-2}(Q_n)| = 2^2\binom{n}{n-2} = 2n(n-1)$.

Case ii: $\chi(\sigma) = n^+$, n odd

Let $\vec{x} \in F_{n-2}(Q_n)$. Since n is odd, we have that $\sigma^i(\vec{x}) \neq \vec{x}$ for all $i \in \{1, ..., n-1\}$. Thus $\sigma^n(\vec{x}) = n$ since σ is sign-preserving, and so the result follows from $|F_{n-2}(Q_n)| = 2^2\binom{n}{n-2} = 2n(n-1)$.

Case iii: $\chi(\sigma) = n^-$

Let $\vec{x} \in F_{n-2}(Q_n)$ with unique nonzero entries x_i and x_j, and let $\ell > 1$ be minimal such that $\sigma^\ell(\vec{x}) = \vec{x}$. Since σ is sign-switching, we will have $\sigma^n(\vec{x}) = -\vec{x}$ and $\sigma^{2n}(\vec{x}) = \vec{x}$. Thus, ℓ divides $2n$.

If n is odd, then $\ell \geqslant n$; since $\sigma^n(\vec{x}) = -\vec{x}$, we must have $\ell = 2n$.

If $n = 2k$, then ℓ is at least k and must divide $2n = 4k$. But since $\sigma^{2k}(\vec{x}) = -\vec{x}$, it must be that ℓ does not divide $n = 2k$. Thus, $\ell = 4k = 2n$. $\square$

4 Cycle Products

Our next results deal with the induced cycle structures of signed permutation consisting of two or more disjoint cycles.

Before doing this, we need to develop some preliminary ideas. Let A and B be disjoint sets and $\rho_1 \colon A \to A$ and $\rho_2 \colon B \to B$ be permutations. Let $\rho_1 \times \rho_2$ be the induced permutation on $A \times B$. One can think of $\rho_1 \times \rho_2$ as follows. The automorphism group of the complete bipartite graph $K_{A,B}$ is $S_A \times S_B$. Now given $\rho_1 \in S_A$ and $\rho_2 \in S_B$, $\rho_1 \times \rho_2$ is the permutation on the edges of $K_{A,B}$.

Proposition 4.1 *If the cycle structure on A by τ_1 is $a \times b$ and the cycle structure on B by τ_2 is $c \times d$, then the induced cycle structure on $A \times B$ by $\tau_1 \times \tau_2$ is*

$$((ac)(gcd(b,d))) \times (lcm(b,d))$$

Proof: Certainly, the length of each induced cycle on $A \times B$ is $\mathrm{lcm}(b,d)$. Now the number of cycles is

$$\frac{abcd}{\mathrm{lcm}(b,d)} = (ac)\mathrm{gcd}(b,d)$$

which completes our proof. $\square$

Let A, B be two sets, and τ_1, τ_2 be permutations acting on A and B respectively such that the cycle structure on A by τ_1 is $\sum_i a_i \times b_i$, and the cycle structure on B by τ_2 is $\sum_j c_j \times d_j$. Define

$$\left(\sum_i a_i \times b_i\right)\left(\sum_j c_j \times d_j\right) = \sum_i \sum_j ((a_i c_j)(\gcd(b_i, d_j))) \times (\mathrm{lcm}(b_i, d_j))$$

Proposition 4.2 *The cycle structure on $A \times B$ induced by $\tau_1 \times \tau_2$ is*

$$\left(\sum_i a_i \times b_i\right)\left(\sum_j c \times d\right) = \sum_i \sum_j ((a_i c_j)(gcd(b_i, d_j))) \times (lcm(b_i, d_j))$$

Proof: Since the cycles of τ_1 and τ_2 partition A and B respectively, we can write $A = \bigcup A_i$ and $B = \bigcup B_j$, where the elements of A_i are exactly those elements in the a_i cycles of length b_i induced by τ_1, and the elements of B_j are exactly those elements in the c_j cycles of length d_j induced by τ_2. Thus, we have that $A \times B$ is partitioned by $A \times B = \bigcup (A_i \times B_j)$. Therefore, the result follows from 4.1. □

Proposition 4.3 *Let $\sigma \in \mathcal{H}_n$ be a product of k disjoint signed cycles $\tau_1, .., \tau_k$ where $\tau_i \in \mathcal{H}_{m_i}$ is a signed m_i-cycle with vertex cycle structure $\kappa_0(\tau_i)$. Then*

$$\kappa_0(\sigma) = \prod \kappa_0(\tau_i)$$

Proof: We use the fact that we can write Q_n as the graph Cartesian product $Q_n = \underset{m_i}{\Box} Q_{m_i}$ where $\sum_i m_i = n$ and each τ_i acts precisely on the vertices of Q_{m_i} for each $i \in \{1, ..., k\}$. The action of σ on $V(Q_n)$ is described by the Cartesian product action of $\tau_1 \times ... \times \tau_k$. Then the result follows inductively from 4.2. □

Proposition 4.4 *Let $\sigma \in \mathcal{H}_n$ be a product of k disjoint signed cycles $\tau_1, .., \tau_k$ where $\tau_i \in \mathcal{H}_{m_i}$ is a signed m_i-cycle with vertex cycle structure $\kappa_0(\tau_i)$ and edge cycle structure $\kappa_1(\tau_i)$. Then*

$$\kappa_1(\sigma) = \sum_{j=1}^{k} \kappa_1(\tau_j) \prod_{i \neq j} \kappa_0(\tau_i)$$

Consider a vector $\vec{v}$ with rows indexed by $\{1, ..., n\}$. Let $A_1, ..., A_m$ be a partition of $\{1, ..., n\}$ and $\vec{v}_i$ be the vector obtained from $\vec{v}$ using the rows from A_i. In this case, we write $\vec{v} = \bigoplus_i \vec{v}_i$.

Proof of Proposition 4.4: We use the fact that we can write $Q_n = \underset{m_i}{\square} Q_{m_i}$ where τ_i acts on Q_{m_i} for each $i \in \{1, ..., k\}$. For any edge $\vec{e} \in F_1(Q_n)$, we can write

$$\vec{e} = \left(\bigoplus_{i \neq j} \vec{v_i} \right) \oplus \vec{e_j}$$

where $\vec{e_j}$ is an edge in Q_{m_j} for some $j \in \{1, ..., k\}$, and $\vec{v_i}$ is a vertex in Q_{m_i} for all $i \neq j$. The reason for this is because the single 0 coordinate of $\vec{e}$ lies in exactly one $E(Q_{m_j})$. Since we can partition $E(Q_n)$ by $E(Q_n) = \bigcup_{j=1}^{k} A_j$ where $A_j = \{\vec{e} = (\bigoplus_{i \neq j} \vec{v_i}) \oplus \vec{e_j} : \vec{e_j} \in E(Q_{m_j})\}$ and since the action of σ on $\vec{e} \in A_j$ is given by the Cartesian product $\tau_1 \times ... \times \tau_k$, the result follows from 4.2. $\square$

Proposition 4.5 *Let $\sigma \in \mathcal{H}_n$ be a product of k disjoint cycles $\tau_1, .., \tau_k$ where $\tau_i \in \mathcal{H}_{m_i}$ is a signed m_i-cycle with facet cycle structure $\kappa_{n-1}(\tau_i)$. Then*

$$\kappa_{n-1}(\sigma) = \sum_{j=1}^{k} \kappa_{m_j - 1}(\tau_j)$$

Proof: We use the fact that we can write $Q_n = \underset{m_i}{\square} Q_{m_i}$ where τ_i acts on Q_{m_i} for each $i \in \{1, ..., k\}$. Since any $\vec{x} \in F_{n-1}(Q_n)$ has exactly one nonzero entry, $\vec{x}$ is equivalent to some $\vec{y} \in F_{m_i-1}(Q_{m_i})$ for exactly one $i \in \{1, ..., k\}$. Thus, the result follows from considering each case for $i \in \{1, ..., k\}$. $\square$

Proposition 4.6 *Let $\sigma \in \mathcal{H}_n$ be a product of k disjoint cycles $\tau_1, .., \tau_k$ where $\tau_i \in \mathcal{H}_{m_i}$ is a signed m_i-cycle with facet cycle structure $\kappa_{n-1}(\tau_i)$ and ridge cycle structure $\kappa_{n-2}(\tau_i)$. Then the ridge cycle structure of σ is given by:*

$$\kappa_{n-2}(\sigma) = \sum_{j=1}^{k} \kappa_{m_j-2}(\tau_j) + \sum_{j=1}^{k} \sum_{i \neq j} \kappa_{m_j-1}(\tau_j)\kappa_{m_i-1}(\tau_i)$$

Proof: We use the fact that we can write $Q_n = \underset{m_i}{\square} Q_{m_i}$ where τ_i acts on Q_{m_i} for each $i \in \{1, ..., k\}$. Since any $\vec{x} \in F_{n-2}(Q_n)$ has exactly two nonzero entries, we will have one of two cases:

If both nonzero entries lie in exactly one Q_{m_j}, then $\vec{x}$ is equivalent to some $\vec{y} \in F_{m_j-2}(Q_{m_j})$ (since all other entries are 0, and therefore invariant under σ). Thus, we consider $\kappa_{m_j-2}(\tau_j)$ for each choice of $j \in \{1, ..., k\}$.

If the nonzero entries lie in two distinct Q_{m_i} and Q_{m_j}, then $\vec{x}$ can is equivalent to the direct sum of two facets $\vec{x_1} \in F_{m_i-1}(Q_{m_i})$ and $\vec{x_2} \in F_{m_j-1}(Q_{m_j})$ (since all other entries are 0, and therefore invariant under σ). Then by 4.2, we need only consider each $\kappa_{m_j-1}(\tau_j)\kappa_{m_i-1}(\tau_i)$ for distinct $i, j \in \{1, ..., k\}$. $\square$

Given $\sigma \in \mathcal{H}_n$ and $\vec{x} \in F_j(Q_n) = F_{n-j-1}(O_n)$, the *index* of $\vec{x}$ with respect to σ, denoted by $|\vec{x}|_\sigma$, is the minimum positive integer t such that $\sigma^t(\vec{x}) = \vec{x}$, i.e., the length of the cycle induced by σ on $\vec{x}$ in $F_j(Q_n)$. The proof of Proposition 4.7 is immediate.

Proposition 4.7 *Let $\sigma \in \mathcal{H}_n$ be a product of disjoint signed cycles $\tau_1, ..., \tau_k$ where $\tau_i \in \mathcal{H}_{m_i}$. Let $\vec{x} \in F_j(Q_n)$ be a j-face such that $\vec{x} = \bigoplus \vec{x}_i$, where $\vec{x}_i \in F_{t_i}(Q_{m_i})$ is a t_i-face acted on by τ_i for each $i \in \{1, ..., k\}$. Then*

$$|\vec{x}|_\sigma = \underset{1 \leqslant i \leqslant k}{lcm}(|\vec{x}_i|_{\tau_i})$$

5 Main Results

Theorem 5.1 *Let $\sigma \in \mathcal{H}_n$. The j-faces of Q_n can be 2-colored with complementing automorphism σ if and only if $\kappa_j(\sigma) = \sum a_i \times b_i$ where b_i is even for all i.*

Proof: Let σ be a color-switching automorphism on $F_j(Q_n)$. Then clearly $\kappa_j(\sigma)$ must consist entirely of even-length cycles.

Now let $\sigma \in \mathcal{H}_n$ such that $\kappa_j(\sigma) = \sum a_i \times b_i$ where each b_i is even. Then the elements of each cycle can be colored red and blue alternatingly. Thus, we have a symmetric 2-coloring. □

Theorem 5.2 *Let $\sigma \in \mathcal{H}_n$, where $\sigma = \tau_1 ... \tau_k$ is a product of disjoint signed cycles. Then σ is a complementing automorphism of Q_n iff it satisfies one of the following:*

i: *Each of $\tau_1, ..., \tau_k$ is even-length.*

ii: *σ contains odd-length cycles but no 1-cycles, and at least one τ_i is sign-switching.*

iii: *At least one τ_i is a 1-cycle, and either σ contains a sign-switching ℓ-cycle for $\ell > 1$ or two sign-switching 1-cycles.*

Proof: We first show that if an element $\sigma \in \mathcal{H}_n$ does not satisfy any of the above three conditions, we can find some odd-length cycle in $\kappa_1(\sigma)$. Given $\sigma = \tau_1 ... \tau_k$, let $m_i = |\tau_i|$ for all $i \in \{1, ..., k\}$ such that at least one m_i is odd, say m_j. Then we have the following cases:

Case i: $m_i > 1$ for all $i \in \{1, ..., k\}$

Assume that each τ_i is sign-preserving. Using Proposition 3.3, we can construct an edge $\vec{x} \in E(Q_n)$ such that $\vec{x} = \bigoplus \vec{x}_i$, where $\vec{x}_i \in V(Q_{m_i})$ is a vertex fixed by τ_i for each $i \neq j$, and $\vec{x}_j$ is an arbitrary edge in Q_{m_j}. Because τ_j is sign-preserving, we have by Proposition 3.6 and Proposition 4.7 that:

$$|\vec{x}|_\sigma = \underset{1 \leqslant i \leqslant k}{\mathrm{lcm}} (|\vec{x}_i|_{\tau_i}) = \mathrm{lcm}(1, ..., 1, m_j) = m_j$$

which is odd.

Case ii: $m_\ell = 1$ for some $\ell \in \{1, ..., k\}$

Assume that σ has at most a single sign-switching 1-cycle (WLOG let τ_ℓ be this 1-cycle) and the remainder of $\tau_1, ..., \tau_k$ are sign-preserving. Now define the edge $\vec{x} \in Q_n$ by $\vec{x} = \bigoplus \vec{x}_i$, where each $\vec{x}_i \in Q_{m_i}$ is a vertex fixed by τ_i for $i \neq \ell$, and $\vec{x}_\ell = [0]$. Then regardless of whether τ_ℓ is sign-preserving or sign-switching, we have $\tau_\ell(\vec{x}_\ell) = \vec{x}_\ell$. Therefore:

$$|\vec{x}|_\sigma = \underset{1 \leqslant i \leqslant k}{\mathrm{lcm}} (|\vec{x}_i|_{\tau_i}) = \mathrm{lcm}(1, ..., 1) = 1$$

Next, we show that any of conditions (i)-(iii) are sufficient to yield no odd-length cycles. We consider the following cases:

Case i: m_i is even for all $i \in \{1, ..., k\}$

By 3.6, we have each $\kappa_1(\tau_i)$ consists entirely of even-length cycles. Therefore, by 4.4, $\kappa_1(\sigma)$ consists wholly of even-length cycles as well.

Case ii: m_j is odd for some j, τ_ℓ is sign-switching for some ℓ, and $m_i > 1$ for all i

Since $m_\ell > 1$, we have by 4.2 that $\kappa_1(\tau_\ell)$ and $\kappa_0(\tau_\ell)$ have no cycles of odd length. Thus, by 4.4, σ induces no odd-length cycles as well.

Case iii: $m_j = 1$ for some j

Subcase iiia: τ_ℓ is sign-switching and $m_\ell > 1$ for some ℓ

By 3.4, we have $\kappa_0(\tau_\ell)$ and $\kappa_1(\tau_\ell)$ consist entirely of even-length cycles. Thus $\kappa_1(\sigma)$ has no odd-length cycles by 4.4.

Subcase iiib: τ_r and τ_s are sign-switching 1-cycles for some r, s

From 3.4, we have $\kappa_0(\tau_r)$ and $\kappa_0(\tau_2)$ have no odd-length cycles. Then the result follows from 4.4. $\square$

Theorem 5.3 *Let $\sigma \in \mathcal{H}_n$, where $\sigma = \tau_1 ... \tau_k$ is a product of disjoint signed cycles. Then σ is a complementing automorphism of O_n iff it satisfies each of the following:*

i: *If $\chi(\tau_i) = \ell_i^+$ for $\ell_i > 1$, then $\ell_i \equiv_4 0$*

ii: *There is at most one τ_i such that $\chi(\tau_i) = 1^+$*

Proof: We first show that if an element $\sigma \in \mathcal{H}_n$ fails to satisfy one of the above conditions, we can find some odd-length cycle induced by σ. Let σ be a product of disjoint cycles $\tau_1, ..., \tau_k$ such that $|\tau_i| = m_i$ for all $i \in \{1, ..., k\}$. We consider two cases:

Case i: There is some τ_i such that $\chi(\tau_i) = m_i^+$ for $m_i > 1$ and $m_i \not\equiv_4 0$

By 3.7, $\kappa_{m_i-2}(\tau_i) = (2m_i - 2) \times m_i$ if m_i is odd. Then we obtain $2m_i - 2$ odd length cycles by 4.6.

If $m_i = 2\ell$ for some odd ℓ, then by 3.7 $\kappa_{m_i-2}(\tau_i) = 2 \times \ell + (4\ell - 3) \times (2\ell)$. Thus, by 4.6, we obtain at least two odd length cycles.

Case ii: There exist two τ_i, τ_j such that $\chi(\tau_i) = \chi(\tau_j) = 1^+$

By 4.2, we have $\kappa_{m_i-1}(\tau_i)\kappa_{m_j-1}(\tau_j) = \kappa_0(\tau_i)\kappa_0(\tau_j) = (2 \times 1)(2 \times 1) = 4 \times 1$. Thus, by 4.6, σ fixes at least 4 ridges.

Next we show that if σ satisfies both conditions, $\kappa_{n-2}(\sigma)$ will consist entirely of even length cycles. By 4.6, it suffices to show that each $\kappa_{m_i-2}(\tau_i)$ and $\kappa_{m_j-1}(\tau_j)\kappa_{m_\ell-1}(\tau_\ell)$ consists of even-length cycles. We consider the following cases for some τ_i:

Case i: $\chi(\tau_i) = m_i^-$

By 3.5, $\kappa_{m_i-1}(\tau_i)$ has all even-length cycles. Similarly, 3.5 guarantees that $\kappa_{m_i-1}(\tau_i)$ has all even-length cycles. Therefore, by 4.2, $\kappa_{m_i-2}(\tau_i)\kappa_{m_j-1}(\tau_j)$ has no odd cycles for all $j \neq i$.

Case ii: $\chi(\tau_i) = m_i^+$ for $m_i > 1$

By hypothesis, $m_i = 4\ell$ for some $\ell \in \mathbb{Z}$. Then by 3.5 and 3.7, $\kappa_{m_i-1}(\tau_i) = 2 \times (4\ell)$ and $\kappa_{m_i-2}(\tau_i) = 2 \times (2\ell) + (8\ell - 3) \times (4\ell)$ have no odd cycles. Furthermore, by 4.2, $\kappa_{m_i-1}(\tau_i)\kappa_{m_j-1}(\tau_j)$ has no odd cycles for all $j \neq i$.

Case iii: $\chi(\tau_i) = 1^+$

Then $\kappa_{m_i-2}(\tau_i) = 0 \times 1$. By hypothesis, τ_i is unique, and every other τ_j falls into one of the above two cases. Thus, we will have that $\kappa_{m_i-1}(\tau_i)\kappa_{m_j-1}(\tau_j)$ yields no odd cycles. $\square$

Computationally, we want to construct all symmetric 2-colorings of O_n and Q_n for $n = 1, 2, 3, 4$. These theorems allow such a computation to be tractable, especially with the the following theorem which says we only need to computationally check one element of $\mathrm{Cl}(\chi)$ rather than all.

Theorem 5.4 *Let σ be a complementing automorphism, and let g be a 2-coloring of σ. Then for any $\tau \in \mathcal{H}_n$, $g\tau^{-1}$ is a 2-coloring with complementing automorphism $\tau\sigma\tau^{-1}$.*

Proof: Let E denote the edge set of Q_n or O_n (for the purpose of the proof, there is no difference). Then we can write $E = R \cup B$ where $g(R) =$ red and $g(B) =$ blue. By definition of g, we know that $\sigma(R) = B$ and $\sigma(B) = R$. Now for $\tau \in \mathcal{H}_n$, let $X = \tau(R)$ and $Y = \tau(B)$. Since τ is an automorphism, X and Y are disjoint and $X \cup Y = E$. Furthermore, we will have $(\tau\sigma\tau^{-1})(X) = (\tau\sigma)(R) = \tau(B) = Y$ and $(g\tau^{-1})(X) = g(R) =$ red. Likewise, $(\tau\sigma\tau^{-1})(Y) = (\tau\sigma)(B) = \tau(R) = X$ and $(g\tau^{-1})(Y) = g(B) =$ blue. Thus $g\tau^{-1}$ is a 2-coloring with complementing automorphism $\tau\sigma\tau^{-1}$ as desired. $\square$

6 Algorithm and Output

Based on the above theorems 5.2, 5.3, and 5.4, the following code was written in Haskell to find all self-complementary graphs in Q_n and O_n for $n = 2, 3, 4$. It should be noted that the following algorithm, while refined by the theory developed in the previous sections, is nowhere near optimal. It is merely a proof of concept of the theory previously discussed.

The algorithm can be summarized as follows. For each conjugacy class χ of $\mathcal{H}_n$, if $\kappa_1(\chi)$ consists of even length cycles only, then 2-color the edges according to these cycles in $\kappa_1(\sigma)$ for any arbitrary $\sigma \in \mathrm{Cl}(\chi)$. The total list of 2-colorings is then checked for isomorphism within the entire group $\mathcal{H}_n$. Of course, this second step is vastly more computationally intensive.

```
import System.IO


--------------------------------

--------MISCELLANEOUS--------

--------------------------------


index1 y [] n = -1

index1 y (x:xs) n =

          if y == x

          then n

          else index1 y xs (n+1)

index y xs = index1 y xs 1

--finds the index of an element y in a list xs, returns -1 if
     y is not an element of xs
```

```
odds xs = [(xs !! n) | n <- [0..(length xs)-1], mod n 2 == 1]
--gets every element of odd index from a list xs


evens xs = [(xs !! n) | n <- [0..(length xs)-1], mod n 2 ==
    0]
--gets every element of even index from a list xs


insertRow r [] n = r:[]
insertRow r xs 1 = r:xs
insertRow r (x:xs) n = x:(insertRow r xs (n-1))
--inserts a row r in a matrix xs at the n-th position


dot x y = sum [(x !! i)*(y !! i)| i <-[0..((length x)-1)]]
--returns the dot product of two vectors x and y


mTrans x u = [(dot a u) | a <- x]
--multiplies (on the left) a vector u by a matrix x


count :: Eq a => a -> [a] -> Int
count x =  length . filter (==x)
--counts the number of instances of x in a list
```

———————————————

———J–FACE SETS———

———————————————

```
vSet :: Integer -> [[Integer]]
vSet 0 = [[]]
vSet n = [1:x | x <- (vSet (n-1))]++[(-1):x | x <- (vSet (n
    -1))]
--lists the vertex elements of the n-cube

eSet :: Integer -> [[Integer]]
eSet 1 = [[0]]
eSet n = [1:x | x <- (eSet (n-1))]++[(-1):x | x <- (eSet (n
    -1))]++[0:x | x <- (vSet (n-1))]
--lists the edge elements of the n-cube

fSet :: Int -> [[Integer]]
fSet 0 = []
fSet n = [1:(take (n-1) (repeat 0)),(-1):(take (n-1) (repeat
    0))]++(map (0:) (fSet (n-1)))
--lists the facet elements of the n-cube
```

```
rSet :: Int -> [[Integer]]
rSet 1 = []
rSet n = (map (1:) (fSet (n-1)))++(map ((-1):) (fSet (n-1)))
    ++(map (0:) (rSet (n-1)))
--lists the ridge elements of the n-cube


-----------------------------------------
-------H_N MATRICES-------
-----------------------------------------


hyperOct 1 = [[[1]],[[-1]]]
hyperOct n = [insertRow (1:(take (n-1) (repeat 0))) y i | y
    <- [map (0:) x | x <- hyperOct (n-1)], i <- [1..n]]++[
    insertRow ((-1):(take (n-1) (repeat 0))) y i | y <- [map
    (0:) x | x <- hyperOct (n-1)], i <- [1..n]]
--lists the matrix elements of the hyperoctohedral group of
    order n


m1m1 = [[-1,0],[0,-1]]
p2 = [[0,1],[1,0]]
m2 = [[0,-1],[1,0]]
```

```
e2Matrices = [m1m1, p2 ,m2]
--lists the "easiest" symmetry matrices that induce 2-
   colorings on the edges of the square


p1m1m1 = [[1 ,0 ,0] ,[0 , -1 ,0] ,[0 ,0 , -1]]
m1m1m1 = [[ -1 ,0 ,0] ,[0 , -1 ,0] ,[0 ,0 , -1]]
p1m2 = [[1 ,0 ,0] ,[0 ,0 , -1] ,[0 ,1 ,0]]
m1m2 = [[ -1 ,0 ,0] ,[0 ,0 , -1] ,[0 ,1 ,0]]
m3 = [[0 ,0 , -1] ,[1 ,0 ,0] ,[0 ,1 ,0]]
e3Matrices = [p1m1m1,m1m1m1, p1m2 ,m1m2,m3]
--lists the "easiest" symmetry matrices that induce 2-
   colorings on the edges of the cube


p1p1m1m1 = [[1 ,0 ,0 ,0] ,[0 ,1 ,0 ,0] ,[0 ,0 , -1 ,0] ,[0 ,0 ,0 , -1]]
p1m1m1m1 = [[1 ,0 ,0 ,0] ,[0 , -1 ,0 ,0] ,[0 ,0 , -1 ,0] ,[0 ,0 ,0 , -1]]
m1m1m1m1 = [[ -1 ,0 ,0 ,0] ,[0 , -1 ,0 ,0] ,[0 ,0 , -1 ,0] ,[0 ,0 ,0 , -1]]
p1p1m2 = [[1 ,0 ,0 ,0] ,[0 ,1 ,0 ,0] ,[0 ,0 ,0 , -1] ,[0 ,0 ,1 ,0]]
p1m1m2 = [[1 ,0 ,0 ,0] ,[0 , -1 ,0 ,0] ,[0 ,0 ,0 , -1] ,[0 ,0 ,1 ,0]]
m1m1p2 = [[ -1 ,0 ,0 ,0] ,[0 , -1 ,0 ,0] ,[0 ,0 ,0 ,1] ,[0 ,0 ,1 ,0]]
m1m1m2 = [[ -1 ,0 ,0 ,0] ,[0 , -1 ,0 ,0] ,[0 ,0 ,0 , -1] ,[0 ,0 ,1 ,0]]
p2p2 = [[0 ,1 ,0 ,0] ,[1 ,0 ,0 ,0] ,[0 ,0 ,0 ,1] ,[0 ,0 ,1 ,0]]
p2m2 = [[0 ,1 ,0 ,0] ,[1 ,0 ,0 ,0] ,[0 ,0 ,0 , -1] ,[0 ,0 ,1 ,0]]
```

```
m2m2 = [[0,-1,0,0],[1,0,0,0],[0,0,0,-1],[0,0,1,0]]

p1m3 = [[1,0,0,0],[0,0,0,-1],[0,1,0,0],[0,0,1,0]]

m1m3 = [[-1,0,0,0],[0,0,0,-1],[0,1,0,0],[0,0,1,0]]

p4 = [[0,0,0,1],[1,0,0,0],[0,1,0,0],[0,0,1,0]]

m4 = [[0,0,0,-1],[1,0,0,0],[0,1,0,0],[0,0,1,0]]

e4Matrices = [p1p1m1m1,p1m1m1m1,m1m1m1m1,p1p1m2,p1m1m2,m1m1p2
    ,m1m1m2,p2p2,p2m2,m2m2,p1m3,m1m3,p4,m4]
--lists the "easiest" symmetry matrices that induce 2-
   colorings on the edges of the tesseract


r4Matrices = [p1m1m1m1,m1m1m1m1,p1m1m2,m1m1m2,m2m2,p1m3,m1m3,
    p4,m4]
--lists the "easiest" symmetry matrices that induce 2-
   colorings on the edges of the hexadecachoron


------------------------------
------2-COLORING SETS---------
------------------------------


ePows1 m e xs =
        if elem (mTrans m e) xs
        then reverse xs
```

```
        else ePows1 m (mTrans m e) ((mTrans m e):xs)
ePows m e = ePows1 m e []
--gives the induced edge cycle of an edge e under a symmetry
   matrix m


eCycles1 m [] xs = xs
eCycles1 m (e:es) xs =
        if elem True [elem e x | x <- xs]
        then eCycles1 m es xs
        else eCycles1 m es ((ePows m e):xs)
eCycles m n = eCycles1 m (eSet n) []
--lists all edge cycles in the n-cube induced by a symmetry
   matrix m


rCycles m n = eCycles1 m (rSet n) []
--lists all ridge cycles in the n-cube induced by a symmetry
   matrix m


twoColor [] = [[]]
twoColor (es:ess) = (map ((evens es)++) (twoColor ess))++(map
    ((odds es)++) (twoColor ess))
--takes a list of cycles and returns the set of all possible
```

```
  2-colorings of those cycles


tessEdge2Color = foldl1 (++) [map ((evens (head (eCycles m 4)
  ))++) (twoColor (tail (eCycles m 4))) | m <- e4Matrices]
--lists all symmetric 2-colorings of the edges of the
  tesseract


cubEdge2Color = foldl1 (++) [map ((evens (head (eCycles m 3))
  )++) (twoColor (tail (eCycles m 3))) | m <- e3Matrices]
--lists all symmetric 2-colorings of the edges of the cube


sqrEdge2Color = foldl1 (++) [map ((evens (head (eCycles m 2))
  )++) (twoColor (tail (eCycles m 2))) | m <- e2Matrices]
--lists all symmetric 2-colorings of the edges of the square


hexEdge2Color = foldl1 (++) [map ((evens (head (rCycles m 4))
  )++) (twoColor (tail (rCycles m 4))) | m <- r4Matrices]
--lists all symmetric 2-colorings of the edges of the
  hexadecachoron
```

———ISOMORPHISM CHECKING———

```
graphSyms gs n = [map (mTrans m) gs | m <- (hyperOct n)]
--lists all symmetries of a given subgraph g

graphEquiv gs hs =
        if elem False [elem g hs | g <- gs]
        then False
        else True
--checks if a graph gs is a subgraph of hs (and thus,
   equivalent in the case of cube subgraphs)

graphIso gs hs n = elem True [graphEquiv gs x | x <- (
   graphSyms hs n)]
--checks if two graphs are isomorphic in the n-cube

removeIso [] xs n = xs
removeIso (g:gs) xs n =
        if elem True [graphIso g x n | x <- xs]
        then removeIso gs xs n
        else removeIso gs (g:xs) n
--removes isomorphic copies from a list of graphs
```

```
sqrEdgeGraphs = removeIso (sqrEdge2Color) [] 2
--lists all symmetric edge 2-colorings in the square up to
   isomorphism

cubeEdgeGraphs = removeIso (cubEdge2Color) [] 3
--lists all symmetric edge 2-colorings in the cube up to
   isomorphism

tessEdgeGraphs = removeIso (tessEdge2Color) [] 4
--lists all symmetric edge 2-colorings in the tesseract up to
    isomorphism

hexEdgeGraphs = removeIso (hexEdge2Color) [] 4
--lists all symmetric edge 2-colorings in the hexadecachoron
   up to isomorphism

--------------------
--------OUTPUT--------
--------------------

q3Output = do
```

```
handle <- openFile "cubeGraphs.txt" AppendMode
mapM_ (hPutStrLn handle) (map show cubeEdgeGraphs)
hClose handle
--outputs graphs found by cubeEdgeGraphs to a text file


o4Output = do
handle <- openFile "hexGraphs.txt" AppendMode
mapM_ (hPutStrLn handle) (map show hexEdgeGraphs)
hClose handle
--outputs graphs found by hexEdgeGraphs to a text file


q4Output = do
handle <- openFile "tessGraphs.txt" AppendMode
mapM_ (hPutStrLn handle) (map show tessEdgeGraphs)
hClose handle
--outputs graphs found by tessEdgeGraphs to a text file
```

The above code was run on an HP 15 Notebook with 4 GB of RAM and a 64-bit operating system. The output functions for $n = 2, 3$ were virtually instantaneous. The output for the $Q_2 = O_2$ was just two graphs [[0,-1],[0,1]] and [[0,-1],[-1,0]]. For Q_3, the algorithm yielded the following output of 8 graphs.

[[0,-1,1],[0,1,-1],[-1,0,1],[-1,0,-1],[1,0,1],[1,0,-1]]

[[0,-1,1],[0,-1,-1],[-1,0,1],[1,0,1],[-1,1,0],[-1,-1,0]] [[0,-1,1],[0,-1,-1],[-1,0,-1],[-1,1,0],[1,0,1],[1,1,0]]

[[0,-1,1],[0,-1,-1],[-1,0,-1],[-1,1,0],[1,0,-1],[1,1,0]]

[[0,-1,1],[0,-1,-1],[-1,0,-1],[-1,-1,0],[1,0,1],[1,1,0]]

[[0,-1,1],[0,-1,-1],[-1,0,-1],[-1,-1,0],[1,0,1],[1,-1,0]]

[[0,-1,1],[0,-1,-1],[-1,0,-1],[-1,-1,0],[1,0,-1],[1,1,0]]

[[0,-1,1],[0,-1,-1],[-1,0,-1],[-1,-1,0],[1,0,-1],[1,-1,0]]

Visually, the graphs above look like the following.

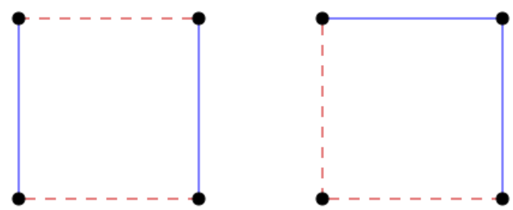

Figure 2: Self-complementary graphs in Q_2

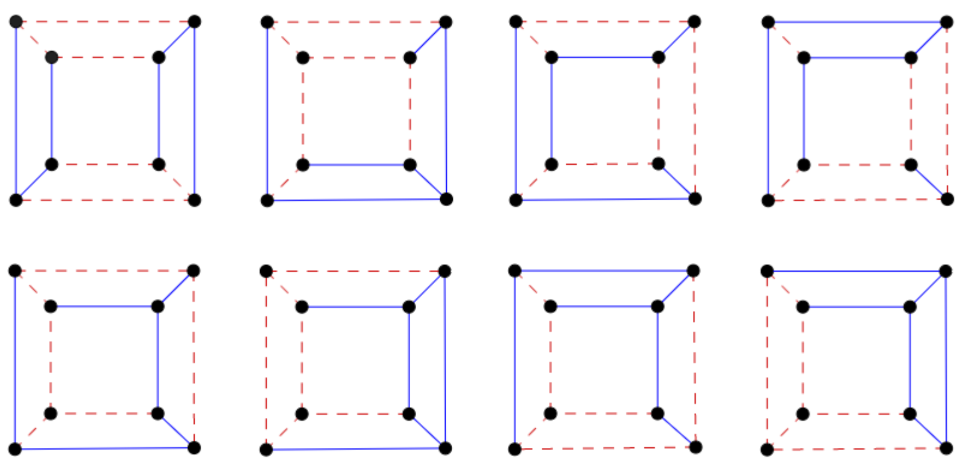

Figure 3: Self-complementary graphs in Q_3

For $n = 4$, the computation was considerably longer. After running for three days, the algorithm found 6074 non-isomorphic self-complementary subgraphs in Q_4

from a total search space of 164680 self-complementary subgraphs. In comparison, the output for O_4 was much smaller. After running for three hours, the algorithm yielded the following 87 graphs.

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[0,-1,1,0],[1,0,0,-1],[0,1,0,-1], [0,-1,0,1],[1,0,1,0],[0,0,1,-1],[1,-1,0,0],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[0,-1,1,0],[1,0,0,-1],[0,1,0,-1], [0,-1,0,1],[1,0,1,0],[0,0,1,-1],[1,-1,0,0],[0,1,1,0],[1,0,0,1]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[0,-1,1,0],[1,0,0,-1],[0,1,0,-1], [0,-1,0,1],[1,0,1,0],[0,1,-1,0],[-1,0,0,1],[0,1,1,0],[1,0,0,1]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[0,-1,1,0],[1,0,0,-1],[0,1,0,-1], [0,-1,0,1],[0,1,0,1],[0,1,-1,0],[-1,0,0,1],[0,1,1,0],[1,0,0,1]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[-1,0,1,0], [1,0,-1,0],[1,0,1,0],[0,0,1,-1],[1,-1,0,0],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[-1,0,1,0], [1,0,-1,0],[0,1,0,1],[0,0,1,-1],[1,-1,0,0],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[-1,0,1,0], [1,0,-1,0],[0,1,0,1],[0,1,-1,0],[-1,0,0,1],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[-1,0,1,0], [1,0,-1,0],[0,1,0,1],[0,1,-1,0],[-1,0,0,1],[0,1,1,0],[1,0,0,1]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[0,1,0,-1], [0,-1,0,1],[1,0,1,0],[0,0,1,-1],[1,-1,0,0],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[0,1,0,-1], [0,-1,0,1],[1,0,1,0],[0,0,1,-1],[1,-1,0,0],[0,1,1,0],[1,0,0,1]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[0,1,0,-1],
[0,-1,0,1],[1,0,1,0],[0,1,-1,0],[-1,0,0,1],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[0,1,0,-1],
[0,-1,0,1],[1,0,1,0],[0,1,-1,0],[-1,0,0,1],[0,1,1,0],[1,0,0,1]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[0,1,0,-1],
[0,-1,0,1],[0,1,0,1],[0,0,1,-1],[1,-1,0,0],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[0,1,0,-1],
[0,-1,0,1],[0,1,0,1],[0,1,-1,0],[-1,0,0,1],[0,0,1,1],[1,1,0,0]]

[[0,-1,0,-1],[0,-1,-1,0],[-1,0,0,-1],[-1,1,0,0],[0,0,-1,1],[0,1,0,-1],
[0,-1,0,1],[0,1,0,1],[0,1,-1,0],[-1,0,0,1],[0,1,1,0],[1,0,0,1]]

[[0,0,-1,1],[0,0,1,-1],[0,1,1,0],[0,-1,-1,0],[-1,0,0,-1],[1,0,0,1],
[0,1,0,-1],[0,-1,0,1],[-1,0,-1,0],[1,0,1,0],[-1,1,0,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,1,1,0],[0,-1,-1,0],[0,1,-1,0],[0,-1,1,0],
[0,1,0,-1],[0,-1,0,1],[-1,0,-1,0],[1,0,1,0],[-1,-1,0,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,1,1,0],[0,-1,-1,0],[0,1,-1,0],[0,-1,1,0],
[0,1,0,-1],[0,-1,0,1],[-1,0,-1,0],[1,0,1,0],[-1,1,0,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,-1,0,1],[0,-1,0,-1],
[-1,0,-1,0],[-1,0,1,0],[1,0,-1,0],[1,0,1,0],[-1,1,0,0],[-1,-1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,1,-1,0],[0,1,1,0],
[-1,0,-1,0],[-1,0,1,0],[-1,-1,0,0],[1,0,-1,0],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,1,-1,0],[0,1,1,0],
[-1,0,-1,0],[-1,0,1,0],[-1,-1,0,0],[1,0,-1,0],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,1,-1,0],[0,1,1,0],

[-1,0,0,1],[-1,0,0,-1],[-1,-1,0,0],[1,0,-1,0],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,1,-1,0],[0,1,1,0],
[-1,0,0,1],[-1,0,0,-1],[-1,-1,0,0],[1,0,-1,0],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,1,-1,0],[0,1,1,0],
[-1,0,0,1],[-1,0,0,-1],[-1,-1,0,0],[1,0,0,1],[1,0,0,-1],[1,1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,1,-1,0],[0,1,1,0],
[-1,0,0,1],[-1,0,0,-1],[-1,-1,0,0],[1,0,0,1],[1,0,0,-1],[1,-1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,-1,0,1],[0,-1,0,-1],
[-1,0,0,1],[-1,0,0,-1],[-1,1,0,0],[1,0,0,1],[1,0,0,-1],[1,1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,-1,0,1],[0,-1,0,-1],
[-1,0,0,1],[-1,0,0,-1],[-1,-1,0,0],[1,0,-1,0],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,-1,0,1],[0,-1,0,-1],
[-1,0,0,1],[-1,0,0,-1],[-1,-1,0,0],[1,0,0,1],[1,0,0,-1],[1,1,0,0]]

[[0,0,-1,1],[0,0,1,-1],[0,-1,-1,0],[0,-1,1,0],[0,-1,0,1],[0,-1,0,-1],
[-1,0,0,1],[-1,0,0,-1],[-1,-1,0,0],[1,0,0,1],[1,0,0,-1],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,1],[1,0,0,1],[-1,0,1,0],[-1,0,-1,0],[-1,1,0,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,1],[1,0,0,1],[-1,0,1,0],[1,0,1,0],[-1,1,0,0],[-1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,1],[1,0,0,1],[-1,0,1,0],[-1,0,-1,0],[1,-1,0,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,1],[1,0,0,1],[-1,0,1,0],[-1,0,-1,0],[1,-1,0,0],[-1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,1],[1,0,0,1],[-1,0,1,0],[-1,0,-1,0],[-1,1,0,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,1],[1,0,0,1],[-1,0,1,0],[-1,0,-1,0],[-1,1,0,0],[-1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,1,0],[-1,1,0,0],[1,0,0,1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,1,0,1],[0,-1,1,0],[0,-1,-1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],

[-1,0,0,-1],[-1,0,1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],

[-1,0,0,-1],[-1,0,1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],

[-1,0,0,-1],[-1,0,1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],

[-1,0,0,-1],[-1,0,1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],

[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,1,1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],

[-1,0,0,-1],[-1,0,1,0],[-1,1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,1,0,0],[1,0,0,-1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,1],[1,0,-1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,1,0],[1,-1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,-1,0],[1,1,0,0]]

[[0,0,-1,1],[0,0,-1,-1],[0,-1,0,1],[0,-1,0,-1],[0,-1,1,0],[0,-1,-1,0],
[-1,0,0,-1],[-1,0,-1,0],[-1,-1,0,0],[1,0,0,-1],[1,0,-1,0],[1,-1,0,0]]

7 References

[1] Adalbert Kerber, *Representations of permutation groups. I*, Lecture Notes in Mathematics, Vol. 240, Springer-Verlag, Berlin-New York, 1971. MR 0325752

[2] Myles F. McNally and Robert R. Molina, *Cataloging self-complementary graphs of order thirteen*, Proceedings of the Twenty-sixth Southeastern International Conference on Combinatorics, Graph Theory and Computing (Boca Raton, FL, 1995), vol. 108, 1995, pp. 11–15. MR 1369272

[3] Gerhard Ringel, *Selbstkomplementäre Graphen*, Arch. Math. (Basel) **14** (1963), 354–358. MR 0154273

[4] H. Sachs, *Über selbstkomplementäre Graphen.*, Publ. Math. **9** (1962), 269–288 (German).