Edge Local Complementation and Equivalence of Binary Linear Codes

Lars Eirik Danielsen and Matthew G. Parker

Department of Informatics, University of Bergen, PB 7803, N-5020 Bergen, Norway
{larsed,matthew}@ii.uib.no
http://www.ii.uib.no/~{larsed,matthew}

Abstract. Orbits of graphs under the operation *edge local complementation* (ELC) are defined. We show that the ELC orbit of a *bipartite* graph corresponds to the equivalence class of a *binary linear code*. The *information sets* and the *minimum distance* of a code can be derived from the corresponding ELC orbit. By extending earlier results on *local complementation* (LC) orbits, we classify the ELC orbits of all graphs on up to 12 vertices. We also give a new method for classifying binary linear codes, with running time comparable to the best known algorithm.

1 Introduction

In this section we first give some definitions from graph theory, in particular we describe the two graph operations *local complementation* (LC) and *edge local complementation* (ELC), the latter also known as the *pivot* operation. We then give some definitions related to *binary linear codes*. Of particular interest is the concept of *code equivalence*. Östergård [1] represented codes as graphs, and devised an algorithm for classifying codes up to equivalence. In Section 2, we show a different way of representing a binary linear code as a *bipartite* graph. We prove that ELC on this graph provides a simple way of jumping between equivalent codes, and that the orbit of a bipartite graph under ELC corresponds to the complete equivalence class of the corresponding code. We also show how ELC on a bipartite graph generates all *information sets* of the corresponding code. Finally, we show that the *minimum distance* of a code is related to the minimum vertex degree over the corresponding ELC orbit. In Section 3 we describe our algorithm for classifying ELC orbits, which we have used to generate all ELC orbits of graphs on up to 12 vertices. Although ELC orbits of non-bipartite graphs do not have any obvious applications to classical coding theory, they are of interest in other contexts, such as *interlace polynomials* [2,3] and *quantum graph states* [4], which are related to quantum error correcting codes. From the ELC orbits of bipartite graphs a classification of binary linear codes can be derived. All binary linear codes have previously been classified up to length 14 [1,5]. We have generated the bipartite ELC orbits of graphs on up to 14 vertices, and this classification can be extended to at least 15 vertices [6], showing that our method is comparable to the best known algorithm. However, the main result of this paper is not a classification of codes, but a new way of representing equivalence classes of codes, and a classification of all ELC orbits of length up to 12.

1.1 Graph Theory

A *graph* is a pair $G = (V, E)$ where V is a set of *vertices*, and $E \subseteq V \times V$ is a set of *edges*. A graph with n vertices can be represented by an $n \times n$ *adjacency matrix* Γ, where $\gamma_{ij} = 1$ if $\{i, j\} \in E$, and $\gamma_{ij} = 0$ otherwise. We will only consider *simple undirected* graphs whose adjacency matrices are symmetric with all diagonal elements being 0, i.e., all edges are bidirectional and no vertex can be adjacent to itself. The *neighbourhood* of $v \in V$, denoted $N_v \subset V$, is the set of vertices connected to v by an edge. The number of vertices adjacent to v is called the *degree* of v. The *induced subgraph* of G on $W \subseteq V$ contains vertices W and all edges from E whose endpoints are both in W. The *complement* of G is found by replacing E with $V \times V - E$, i.e., the edges in E are changed to non-edges, and the non-edges to edges. Two graphs $G = (V, E)$ and $G' = (V, E')$ are *isomorphic* if and only if there exists a permutation π on V such that $\{u, v\} \in E$ if and only if $\{\pi(u), \pi(v)\} \in E'$. A *path* is a sequence of vertices, $(v_1, v_2, \ldots, v_i)$, such that $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{i-1}, v_i\} \in E$. A graph is *connected* if there is a path from any vertex to any other vertex in the graph. A graph is *bipartite* if its set of vertices can be decomposed into two disjoint sets such that no two vertices within the same set are adjacent. We call a graph (a, b)-*bipartite* if its vertices can be decomposed into sets of size a and b.

Definition 1 ([7,8]). *Given a graph $G = (V, E)$ and a vertex $v \in V$, let $N_v \subset V$ be the neighbourhood of v.* Local complementation *(LC) on v transforms G into $G * v$ by replacing the induced subgraph of G on N_v by its complement.*

Definition 2 ([8]). *Given a graph $G = (V, E)$ and an edge $\{u, v\} \in E$,* edge local complementation *(ELC) on $\{u, v\}$ transforms G into $G^{(uv)} = G*u*v*u = G*v*u*v$.*

Definition 3 ([8]). *ELC on $\{u, v\}$ can equivalently be defined as follows. Decompose $V \setminus \{u, v\}$ into the following four disjoint sets, as visualized in Fig. 1.*

- A *Vertices adjacent to u, but not to v.*
- B *Vertices adjacent to v, but not to u.*
- C *Vertices adjacent to both u and v.*
- D *Vertices adjacent to neither u nor v.*

To obtain $G^{(uv)}$, perform the following procedure. For any pair of vertices $\{x, y\}$, where x belongs to class A, B, or C, and y belongs to a different class A, B, or C, "toggle" the pair $\{x, y\}$, i.e., if $\{x, y\} \in E$, delete the edge, and if $\{x, y\} \notin E$, add the edge $\{x, y\}$ to E. Finally, swap the labels of vertices u and v.

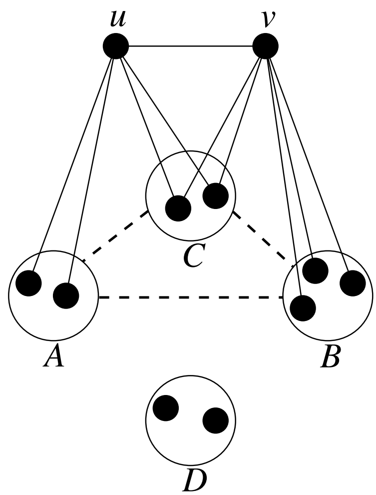

Fig. 1: Visualization of the ELC Operation

Definition 4. *The* LC orbit *of a graph G is the set of all graphs that can be obtained by performing any sequence of LC operations on G. Similarly, the* ELC orbit *of G comprises all graphs that can be obtained by performing any sequence of ELC operations on G. (Usually we only consider LC and ELC orbits of unlabeled graphs. In the cases where we consider orbits of labeled graphs, this will be noted.)*

The LC operation was first defined by Fon-der-Flaas [7], and later studied by Bouchet [8] in the context of isotropic systems. Bouchet defined ELC as "complementation along an edge" [8], but this operation is also known as *pivoting* on a graph [2, 9]. LC orbits of graphs have been used to study *quantum graph states* [10,11,12], which are equivalent to *self-dual additive codes over* GF(4) [13]. We have previously used LC orbits to classify such codes [14, 15]. ELC orbits have also been studied in the context of quantum graph states [4, 9]. *Interlace polynomials* of graphs have been defined with respect to both ELC [2] and LC [3]. These polynomials encode properties of the graph orbits, and were originally used to study a problem related to DNA sequencing [16].

Proposition 1. *If $G = (V, E)$ is a connected graph, then, for any vertex $v \in V$, $G * v$ must also be connected. Likewise, for any edge $\{u, v\} \in E$, $G^{(uv)}$ must be connected.*

Proof. If the edge $\{x, y\}$ is deleted as part of an LC operation on v, both x and y must be, and will remain, connected to v. Similarly, if by performing ELC on the edge $\{u, v\}$, the edge $\{x, y\}$ is deleted, both x and y will remain connected to either u, v, or both, and u and v will remain connected. □

Proposition 2 ([9]). *If G is an (a, b)-bipartite graph, then, for any edge $\{u, v\} \in E$, $G^{(uv)}$ must also be (a, b)-bipartite.*

Proof. A bipartite graph with an edge $\{u, v\}$ can not contain any vertex that is connected to both u and v. Using the terminology of Definition 3, the set C will always be empty when we perform ELC on a bipartite graph. Moreover, all vertices in the set A must belong to the same bipartition as u, and all vertices in B must belong to the same bipartition as v. All edges that are added or deleted have one endpoint in A and one in B, and it follows that bipartiteness is preserved. □

Proposition 3. *Let G be a bipartite graph, and let $\{u, v\} \in E$. Then $G^{(uv)}$ can be obtained by "toggling" all edges between the sets $N_u \backslash \{v\}$ and $N_v \backslash \{u\}$, followed by a swapping of vertices u and v.*

1.2 Coding Theory

A binary linear code, $\mathcal{C}$, is a linear subspace of $\mathrm{GF}(2)^n$ of dimension k, where $0 \leq k \leq n$. $\mathcal{C}$ is called an $[n, k]$ code, and the 2^k elements of $\mathcal{C}$ are called *codewords*. The *Hamming weight* of $\boldsymbol{u} \in \mathrm{GF}(2)^n$, denoted $\mathrm{wt}(\boldsymbol{u})$, is the number of nonzero components of $\boldsymbol{u}$. The *Hamming distance* between $\boldsymbol{u}, \boldsymbol{v} \in \mathrm{GF}(2)^n$ is $\mathrm{wt}(\boldsymbol{u} - \boldsymbol{v})$. The *minimum distance* of the code $\mathcal{C}$ is the minimal Hamming distance between any two codewords of $\mathcal{C}$. Since $\mathcal{C}$ is a linear code, the minimum distance is also given by the smallest weight of any codeword in $\mathcal{C}$. A code with minimum distance d is called an $[n, k, d]$ code. A code is *decomposable* if it can be written as the *direct sum* of two smaller codes. For example, let $\mathcal{C}$ be an $[n, k, d]$ code and $\mathcal{C}'$ an $[n', k', d']$ code. The direct sum, $\mathcal{C} \oplus \mathcal{C}' = \{u||v \mid u \in \mathcal{C}, v \in \mathcal{C}'\}$, where $||$ means concatenation, is an $[n + n', k + k', \min\{d, d'\}]$ code. Two codes, $\mathcal{C}$ and $\mathcal{C}'$, are considered to be *equivalent* if one can be obtained from the other by some permutation of the coordinates, or equivalently, a permutation of the columns of a generator matrix. We define the *dual* of the code $\mathcal{C}$ with respect to the standard inner product, $\mathcal{C}^\perp = \{\boldsymbol{u} \in \mathrm{GF}(2)^n \mid \boldsymbol{u} \cdot \boldsymbol{c} = 0 \text{ for all } \boldsymbol{c} \in \mathcal{C}\}$. $\mathcal{C}$ is called *self-dual* if $\mathcal{C} = \mathcal{C}^\perp$, and *isodual* if $\mathcal{C}$ is equivalent to $\mathcal{C}^\perp$. Self-dual and isodual codes must have even length n, and dimension $k = \frac{n}{2}$. The code $\mathcal{C}$ can be defined by a $k \times n$ *generator matrix*, C, whose rows span $\mathcal{C}$. A set of k linearly independent columns of C is called an *information set* of $\mathcal{C}$. We can permute the columns of C such that an information set make up the first k columns. By elementary row operations, this matrix can then be transformed into a matrix of the form $C' = (I \mid P)$, where I is a $k \times k$ identity matrix, and P is some $k \times (n - k)$ matrix. The matrix C', which is said to be of *standard form*, generates a code $\mathcal{C}'$ which is equivalent to $\mathcal{C}$. Every code is equivalent to a code with a generator matrix of standard form. The matrix $H' = (P^{\mathrm{T}} \mid I)$, where I is an $(n - k) \times (n - k)$ identity matrix is called the *parity check matrix* of $\mathcal{C}'$. Observe that $G' {H'}^{\mathrm{T}} = \mathbf{0}$, where $\mathbf{0}$ is the all-zero vector. It follows that H' must be the generator matrix of $\mathcal{C}'^\perp$.

2 ELC and Code Equivalence

As mentioned earlier, LC orbits of graphs correspond to equivalence classes of self-dual quantum codes. We have previously classified all such codes of length up to 12 [15], by classifying LC orbits of simple undirected graphs. In this paper, we show that ELC orbits of bipartite graphs correspond to the equivalence classes of binary linear codes. First we explain how a binary linear code can be represented by a graph.

Definition 5 ([17, 18]). *Let $\mathcal{C}$ be a binary linear $[n,k]$ code with generator matrix $C = (I \mid P)$. Then the code $\mathcal{C}$ corresponds to the $(k, n-k)$-bipartite graph on n vertices with adjacency matrix*

$$\Gamma = \begin{pmatrix} \mathbf{0}_{k\times k} & P \\ P^T & \mathbf{0}_{(n-k)\times(n-k)} \end{pmatrix},$$

where $\mathbf{0}$ denote all-zero matrices of the specified dimensions.

Theorem 1. *Let $G = (V, E)$ be the $(k, n-k)$-bipartite graph derived from a standard form generator matrix $C = (I \mid P)$ of the $[n,k]$ code $\mathcal{C}$. Let G' be the graph obtained by performing ELC on the edge $\{u,v\} \in E$, followed by a swapping of vertices u and v. Then the code $\mathcal{C}'$ generated by $C' = (I \mid P')$ corresponding to G' is equivalent to $\mathcal{C}$, and can be obtained by interchanging coordinates u and v of $\mathcal{C}$.*

Proof. Assume, without loss of generality, that $u \le k$ and $v > k$. C' can be obtained from C by adding row u to all columns in $N_v \setminus \{u\}$ and then swapping columns u and v, where N_v denotes the neighbourhood of v in G. These operations preserve the equivalence of linear codes. As described in Proposition 3, the bipartite graph G is transformed into G' by "toggling" all pairs of vertices $\{x,y\}$, where $x \in N_u \setminus \{v\}$ and $y \in N_v \setminus \{u\}$. This action on the submatrix P is implemented by the row additions on C described above. However, this also "toggles" the pairs $\{v,y\}$, where $y \in N_v \setminus \{u\}$, transforming column v of C into a vector with 0 in all coordinates except u. But column u of C now contains the original column v, and thus swapping columns u and v restores the neighbourhood of v, giving the desired submatrix P. □

Corollary 1. *Applying any sequence of ELC operations to a graph G corresponding to a code $\mathcal{C}$ will produce a graph corresponding to an equivalent code.*

Instead of mapping the generator matrix $C = (I \mid P)$ to the adjacency matrix of a bipartite graph in order to perform ELC on the edge $\{u,v\}$, we can work directly with the submatrix P. Let the rows of P be labeled $1, 2, \ldots, k$ and the columns of P be labeled $k+1, k+2, \ldots, n$. Assume that u indicates a row of P and that v indicates a column of P. The element P_{ij} is then replaced by $1 - P_{ij}$ if $i \ne u$, $j \ne v$, and $P_{uj} = P_{iv} = 1$.

Consider a code $\mathcal{C}$. As described in Section 1.2, it is possible to go from a generator matrix of standard form, $C = (I \mid P)$, to another generator matrix of standard form, C', of a code equivalent to $\mathcal{C}$ by one of the $n!$ possible permutations of the columns of C, followed by elementray row operations. More precisely, we can get from C to C' via a combination of the following operations.

1. Permuting the columns of P.
2. Permuting the columns of I, followed by the same permutation on the rows of C, to restore standard form.
3. Swapping columns from I with columns from P, such that the first k columns still is an information set, followed by some elementary row operations to restore standard form.

Theorem 2. *Let $\mathcal{C}$ and $\mathcal{C}'$ be equivalent codes. Let C and C' be matrices of standard form generating $\mathcal{C}$ and $\mathcal{C}'$. Let G and G' be the bipartite graphs corresponding to C and C'. G' is isomorphic to a graph obtained by performing some sequence of ELC operations on G.*

Proof. $\mathcal{C}$ and $\mathcal{C}'$ must be related by a combination of the operations 1, 2, and 3 listed above. It is easy to see that operations 1 and 2 applied to G produce an isomorphic graph. It remains to prove that operation 3 always corresponds to some sequence of ELC operations. We know from Theorem 1 that swapping columns u and v of C, where u is part of I and v is part of P, corresponds to ELC on the edge $\{u, v\}$ of G, followed by a swapping of the vertices u and v. When $\{u, v\}$ is not an edge of G, we can not swap columns u and v of C via ELC. In this case, coordinate v of column u is 0, and column u has 1 in coordinate u and 0 elsewhere. Swapping these columns would result in a generator matrix where the first k columns all have 0 at coordinate u. These columns can not correspond to an information set. It follows that if $\{u, v\}$ is not an edge of G, swapping columns u and v is not a valid operation of type 3 in the above list. Thus ELC and graph isomorphsim cover all possible operations that map standard form generator matrices of equivalent codes to each other. □

Let us for a moment consider ELC orbits of *labeled* graphs, i.e., where we do not take isomorphism into consideration. Let $G = (V, E)$ be the bipartite graph representing the code $\mathcal{C}$, and G' be the graph obtained by ELC on the edge $\{u, v\} \in E$. Since we perform ELC on $\{u, v\}$ without swapping u and v afterwards, the adjacency matrix of G' will not be of the type we saw in Definition 5. We can think of G as a graph corresponding to the information set $\{1, 2, \dots, k\}$ of $\mathcal{C}$. Assume that $u \leq k$ and $v > k$. G' will then represent another information set of $\mathcal{C}$, namely $\{1, 2, \dots, k\} \setminus \{u\} \cup \{v\}$. With this interpretation, the next corollary follows from Theorem 2.

Corollary 2. *Let G be the bipartite graph representing the code $\mathcal{C}$. Each labeled graph in the ELC orbit of G corresponds to an information set of $\mathcal{C}$. Moreover, the number of information sets of $\mathcal{C}$ equals the number of labeled graphs in the ELC orbit of G.*

Note that the distinction between ELC with or without a final swapping of vertices is only significant when we want to find information sets. For other applications, where we consider graphs up to isomorphism, this distinction is not of importance.

Theorem 3. *The minimum distance, d, of a binary linear $[n,k,d]$ code $\mathcal{C}$, is equal to $\delta+1$, where δ is the smallest vertex degree over all graphs in the associated ELC orbit.*

Proof. A vertex with degree $d-1$ in the ELC orbit corresponds to a codeword of weight d. We need to show that such a vertex always exists. Let C be a generator matrix of standard form that generates a code equivalent to $\mathcal{C}$. Find a codeword c of weight d, generated by C, and let the i-th row of C be one of the rows that c is linearly dependent on. Permute the columns of C to obtain C' where c is mapped to c' with 1 in coordinate i, and with the rest of the k first coordinates being 0. Replace the i-th row of C' by c' to get C''. Note that the first k columns of C'' will be an information set, so we can transform C'' into a matrix of the form $(I \mid P)$ by elementary row operations. Row i of this final matrix has weight d, and thus the corresponding bipartite graph has a vertex with degree $d-1$. $\square$

3 Classification of ELC Orbits

We have previously classified all self-dual additive codes over GF(4) of length up to 12 [19, 15], by classifying orbits of simple undirected graphs with respect to local complementation and graph isomorphism. In Table 1, the sequence (i_n^{LC}) gives the number of LC orbits of connected graphs on n vertices, while (t_n^{LC}) gives the total number of LC orbits of graphs on n vertices.

Table 1: Numbers of LC Orbits

n	1	2	3	4	5	6	7	8	9	10	11	12
i_n^{LC}	1	1	1	2	4	11	26	101	440	3,132	40,457	1,274,068
t_n^{LC}	1	2	3	6	11	26	59	182	675	3,990	45,144	1,323,363

By recursively applying ELC operations to all edges of a graph, whilst checking for graph isomorphism using the program *nauty* [20], we can find all members of the ELC orbit. Let $\boldsymbol{G}_n$ be the set of all unlabeled simple undirected connected graphs on n vertices. Let the set of all distinct ELC orbits of connected graphs on n vertices be a partitioning of $\boldsymbol{G}_n$ into i_n^{ELC} disjoint sets. Our previous classification of the LC orbits of all graphs of up to 12 vertices helps us to classify ELC orbits, since it follows from Definition 2 that each LC orbit can be partitioned into a set of disjoint ELC orbits. We have used this fact to classify all ELC orbits of graphs on up to 12 vertices, a computation that required approximately one month of running time on a parallel cluster computer. In Table 2, the sequence (i_n^{ELC}) gives the number of ELC orbits of connected graphs on n vertices, while (t_n^{ELC}) gives the total number of ELC orbits of graphs on n

vertices. A database containing one representative from each ELC orbit can be found at `http://www.ii.uib.no/~larsed/pivot/`.

Table 2: Numbers of ELC Orbits and Binary Linear Codes

n	i_n^{ELC}	t_n^{ELC}	$i_n^{ELC,B}$	$t_n^{ELC,B}$	i_n^{C}	$i_n^{C_{iso}}$
1	1	1	1	1	1	-
2	1	2	1	2	1	1
3	2	4	1	3	2	-
4	4	9	2	6	3	1
5	10	21	3	10	6	-
6	35	64	8	22	13	3
7	134	218	15	43	30	-
8	777	1,068	43	104	76	10
9	6,702	8,038	110	250	220	-
10	104,825	114,188	370	720	700	40
11	3,370,317	3,493,965	1,260	2,229	2,520	-
12	231,557,290	235,176,097	5,366	8,361	10,503	229
13	*?*	*?*	25,684	36,441	51,368	-
14			154,104	199,610	306,328	1,880
15			1,156,716	1,395,326	2,313,432	-
16			*?*	*?*	*23,069,977*	*?*
17			*157,302,628*	*?*	*314,605,256*	-

We are particularly interested in bipartite graphs, because of their connection to binary linear codes. For the classification of the orbits of bipartite graphs with respect to ELC and graph isomorphism, the following technique is helpful. If G is an (a, b)-bipartite graph, it has $2^a + 2^b - 2$ possible *extensions*. Each extension is formed by adding a new vertex and joining it to all possible combinations of at least one of the old vertices. Let $\boldsymbol{P}_n$ be a set containing one representative from each ELC orbit of all connected bipartite graphs on n vertices. The set $\boldsymbol{E}_n$ is formed by making all possible extensions of all graphs in $\boldsymbol{P}_{n-1}$. It can then be shown that $\boldsymbol{P}_n \subset \boldsymbol{E}_n$, i.e., that the set $\boldsymbol{E}_n$ will contain at least one representative from each ELC orbit of connected bipartite graphs on n vertices. The set $\boldsymbol{E}_n$ will be much smaller than $\boldsymbol{G}_n$, so it will be more efficient to search for a set of ELC orbit representatives within $\boldsymbol{E}_n$. A similar technique was used by Glynn, et al. [10] to classify LC orbits.

In Table 2, the sequence $(i_n^{ELC,B})$ gives the number of ELC orbits of connected bipartite graphs on n vertices, and $(t_n^{ELC,B})$ gives the total number of ELC orbits of bipartite graphs on n vertices.

Theorem 4. *Let $k \neq \frac{n}{2}$. Then the number of inequivalent binary linear $[n, k]$ codes, which is also the number of inequivalent $[n, n-k]$ codes, is equal to the number of ELC orbits of $(n-k, k)$-bipartite graphs.*

When n is even and $k = \frac{n}{2}$, the number of inequivalent binary linear $[n, k]$ codes is equal to twice the number of ELC orbits of (k, k)-bipartite graphs minus the number of isodual codes of length n.

Proof. We recall that if a code $\mathcal{C}$ is generated by $(I \mid P)$, then its dual, $\mathcal{C}^\perp$, is generated by $(P^{\mathrm{T}} \mid I)$. Also note that $\mathcal{C}^\perp$ is equivalent to the code generated by $(I \mid P^{\mathrm{T}})$. The bipartite graphs corresponding to the codes generated by $(I \mid P)$ and $(I \mid P^{\mathrm{T}})$ are isomorphic. It follows that the ELC orbit associated with an $[n, k]$ code $\mathcal{C}$ is simultaneously the orbit associated with the dual $[n, n-k]$ code $\mathcal{C}^\perp$. In the case where $k = \frac{n}{2}$, each ELC orbit corresponds to two non-equivalent $[n, k]$ codes, except in the case where $\mathcal{C}$ is isodual. □

Corollary 3. *The total number of binary linear codes of length n is equal to twice the number of ELC orbits of graphs on n vertices, minus the number of isodual codes of length n.*

Note that if we only consider connected graphs on n vertices, we get the number of indecomposable codes of length n, i_n^C. The total number of codes can easily be derived from the values of (i_n^C). Table 2 gives the number of ELC orbits of connected bipartite graphs on n vertices, $i_n^{ELC,B}$, the number of indecomposable binary linear codes of length n, i_n^C, and the number of indecomposable isodual codes of length n, $i_n^{C_{iso}}$. A method for counting the number of binary linear codes by using computer algebra tools was devised by Fripertinger and Adalbert [21]. A table enumerating binary linear codes of length up to 25 is available online at `http://www.mathe2.uni-bayreuth.de/frib/codes/tables_2.html`. The numbers in italics in Table 2 are taken from this webpage. Note however that this approach only gives the number of inequivalent codes, and does not produce the codes themselves. Complete classification of binary linear codes has been carried out up to length 14 by Östergård [1]. He also used a graph-based algorithm, but one quite different from the method described in this paper. In a recent book by Kaski and Östergård [5], it is proposed as a research problem to extend this classification to lengths higher than 14. Sang-il Oum [6] demonstrated that the 1,395,326 ELC orbits of bipartite graphs on 15 vertices can be generated in about 58 hours. This indicates that classification of codes by ELC orbits is comparable to the currently best known algorithm. It may also be possible that our method will be more efficient than existing algorithms for classifying special types of codes. For instance, matrices of the form $(I \mid P)$, where P is symmetric, generate a subset of the isodual codes. The bipartite graphs corresponding to these codes, which were also studied by Curtis [17], should be well suited to our method, since any graph of this type must arise as an extension of a graph of the same type.

References

1. Östergård, P.R.J.: Classifying subspaces of Hamming spaces. Des. Codes Cryptogr. **27** (2002) 297–305

2. Arratia, R., Bollobás, B., Sorkin, G.B.: The interlace polynomial of a graph. J. Combin. Theory Ser. B **92** (2004) 199–233 `arXiv:math.CO/0209045`.
3. Aigner, M., van der Holst, H.: Interlace polynomials. Linear Algebra Appl. **377** (2004) 11–30
4. Van den Nest, M., De Moor, B.: Edge-local equivalence of graphs. Preprint, `arXiv:math.CO/0510246` (2005)
5. Kaski, P., Östergård, P.R.J.: Classification algorithms for codes and designs. Volume 15 of Algorithms and Computation in Mathematics. Springer-Verlag, Berlin (2006)
6. Oum, S. Personal communication (2006)
7. Fon-der Flaas, D.G.: On local complementations of graphs. In: Combinatorics (Eger, 1987). Volume 52 of Colloq. Math. Soc. János Bolyai. North-Holland, Amsterdam (1988) 257–266
8. Bouchet, A.: Graphic presentations of isotropic systems. J. Combin. Theory Ser. B **45** (1988) 58–76
9. Riera, C., Parker, M.G.: On pivot orbits of Boolean functions. In: Fourth International Workshop on Optimal Codes and Related Topics, Sofia, Institute of Mathematics and Informatics, Bulgarian Academy of Sciences (2005) 248–253
10. Glynn, D.G., Gulliver, T.A., Maks, J.G., Gupta, M.K.: The geometry of additive quantum codes. Submitted to Springer-Verlag (2004)
11. Hein, M., Eisert, J., Briegel, H.J.: Multi-party entanglement in graph states. Phys. Rev. A **69** (2004) 062311 `arXiv:quant-ph/0307130`.
12. Van den Nest, M., Dehaene, J., De Moor, B.: Graphical description of the action of local Clifford transformations on graph states. Phys. Rev. A **69** (2004) 022316 `arXiv:quant-ph/0308151`.
13. Calderbank, A.R., Rains, E.M., Shor, P.M., Sloane, N.J.A.: Quantum error correction via codes over GF(4). IEEE Trans. Inform. Theory **44** (1998) 1369–1387 `arXiv:quant-ph/9608006`.
14. Danielsen, L.E., Parker, M.G.: Spectral orbits and peak-to-average power ratio of Boolean functions with respect to the $\{I,H,N\}^n$ transform. In: Sequences and Their Applications – SETA 2004. Volume 3486 of Lecture Notes in Comput. Sci., Berlin, Springer-Verlag (2005) 373–388 `arXiv:cs.IT/0504102`.
15. Danielsen, L.E., Parker, M.G.: On the classification of all self-dual additive codes over GF(4) of length up to 12. J. Combin. Theory Ser. A **113** (2006) 1351–1367 `arXiv:math.CO/0504522`.
16. Arratia, R., Bollobás, B., Coppersmith, D., Sorkin, G.B.: Euler circuits and DNA sequencing by hybridization. Discrete Appl. Math. **104** (2000) 63–96
17. Curtis, R.T.: On graphs and codes. Geom. Dedicata **41** (1992) 127–134
18. Parker, M.G., Rijmen, V.: The quantum entanglement of binary and bipolar sequences. In: Sequences and Their Applications – SETA '01. Discrete Math. Theor. Comput. Sci., London, Springer-Verlag (2002) 296–309 `arXiv:quant-ph/0107106`.
19. Danielsen, L.E.: On self-dual quantum codes, graphs, and Boolean functions. Master's thesis, Department of Informatics, University of Bergen, Norway (2005) `arXiv:quant-ph/0503236`.
20. McKay, B.D.: nauty User's Guide. (2003) `http://cs.anu.edu.au/~bdm/nauty/`.
21. Fripertinger, H., Kerber, A.: Isometry classes of indecomposable linear codes. In: Applied algebra, algebraic algorithms and error-correcting codes. Volume 948 of Lecture Notes in Comput. Sci., Berlin, Springer-Verlag (1995) 194–204