Serdica J. Computing **8** (2014), No 4, 327–354

Serdica
Journal of Computing

Bulgarian Academy of Sciences
Institute of Mathematics and Informatics

REPRESENTING EQUIVALENCE PROBLEMS FOR COMBINATORIAL OBJECTS*

Iliya Bouyukliev, Mariya Dzhumalieva-Stoeva

Abstract. Methods for representing equivalence problems of various combinatorial objects as graphs or binary matrices are considered. Such representations can be used for isomorphism testing in classification or generation algorithms. Often it is easier to consider a graph or a binary matrix isomorphism problem than to implement heavy algorithms depending especially on particular combinatorial objects. Moreover, there already exist well tested algorithms for the graph isomorphism problem (nauty) and the binary matrix isomorphism problem as well (Q-Extension).

1. Introduction and preliminaries. Isomorphism computations take place in every classification algorithm and also in algorithms for generating objects of a certain type. In general the combinatorial classification is concerned with a given finite set of combinatorial objects A and an equivalence relation $(A, \cong)$ in it. The classification problem is to find exactly one representative in each equivalence class.

ACM Computing Classification System (1998): F.2.1, G.4.

Key words: isomorphisms, graphs, binary matrices, combinatorial objects.

[*] This work was partially supported by the Bulgarian National Science Fund under Contract I01/0003/2012.

In terms of algebra the equivalence relation is defined as an action of a finite group G on the set of objects and the equivalence classes are defined as orbits of the action of the group on it. In particular two given objects are equivalent if they belong to one and the same equivalence class or one and the same orbit of G on A.

There are two general types of isomorphism problem algorithms. Let X, Y are objects of the finite set A. The first approach to check whether $X \cong Y$ is to use an adapted algorithm specific for certain objects, where X and Y are compared via invariants. If the invariants differ, the objects are not equivalent. But if they are the same, additional computations are required to determine whether Y belongs to the equivalence class of X. The performance of the algorithms depends at most on the order of the group G acting on the set A. In many cases G is too large and the process of equivalence search becomes a very hard task. Such algorithms have been designed for linear codes [13, 20, 28], designs [23], Hadamard matrices [21]. The second type of algorithms consists of obtaining canonical forms for both X and Y using canonical representative map. To test whether $X \cong Y$ is to test their canonical forms for equality. This approach is implemented also for linear codes [3], designs and graphs [17, 25, 18]. In most cases such algorithms are more effective than the specific algorithms of the first type.

In the paper we investigate another type of algorithms, operating on a category on which most types of objects can be represented. In other words we represent the isomorphism problem of some combinatorial objects as the isomorphism problem of two basic objects—graphs and $\{0, 1\}$-matrices (binary matrices). Historically, most combinatorial objects are presented in terms of graph theory. Examples for representing of combinatorial objects as graphs are given by Kaski and Östergård [16] (Ch. 3). There already exist algorithms for the graph isomorphism problem [10, 12, 14, 15]. The best known is the McKay's NAUTY algorithm [25, 26]. On the other hand some of the objects have a more natural computer representation as binary matrices (designs, projective planes, etc). Such an approach is already applied for the classification of linear codes [6], self-dual codes [1, 2], Hadamard matrices [5]. An algorithm for binary matrices isomorphism is included in the package Q-EXTENSION [4], developed by the first author.

It is not difficult to switch from the graph isomorphism problem to the binary matrix isomorphism problem as these two objects have natural representations into each other. That's why each combinatorial object, represented as a graph, could be represented as a binary matrix too. In some cases the second representation is more convenient. A recent elegant example of using binary matrices in terms of bipartite graphs is considered in an algorithm for isomorph-free generation of regular directed graphs [7].

In this paper, we give representations of directed graphs, linear and nonlinear codes and Hadamard matrices directly as binary matrices and colored binary matrices in a different and more efficient way. We want to emphasize that, to the best of our knowledge, a representation of the equivalence problem of linear codes in the general case (for prime and composite finite fields) is not known as of now. Representing objects in this way often leads to a reduction of memory and running time of the machine. But applying such an approach to a classification or generation problem requires a good knowledge of the objects under consideration and the definition of isomorphism. Furthermore, the isomorphism itself may be represented in the terms of a group action. All codes in this paper have full length, i.e., they do not have a coordinate which is identically zero in all words.

The paper is organized as follows: Section 2 describes the isomorphism of binary matrices. Section 3 is devoted to the isomorphism of graphs. In Section 4 we investigate the connection between codes and binary matrices. Hadamard matrices are considered in Section 5.

2. Isomorphism of binary matrices. In this section we present the main definitions related to isomorphisms of binary matrices and colored binary matrices.

A *binary matrix* M, also known as a $\{0,1\}$-matrix, is a $m \times n$ matrix with entries from the alphabet $\mathbb{F}_2 = \{0,1\}$.

Let us denote by Ω the set of all binary $m \times n$ matrices.

Definition 2.1. *Two binary matrices A and B of the same size are* ***isomorphic*** *($A \cong B$) if the rows of A can be obtained from the rows of B after a permutation of the columns of B. All isomorphisms form the set $Iso(A,B)$.*

This definition is based on the natural action of the symmetric group S_n on the set of columns for all matrices in Ω. Any permutation of the columns of A which maps the rows of A into the rows of the same matrix, is called an automorphism of A. The set of all automorphisms of A is a subgroup of the symmetric group S_n and we denote it by $\mathrm{Aut}(A)$.

We also give another definition which is equivalent to Definition 2.1 but it is more useful in some cases.

Definition 2.2. *Two matrices of the same size are* ***isomorphic*** *if the second one can be obtained from the first one by permutations of the columns and the rows.*

For some applications, we need to define a coloring of binary matrices, especially in algorithms, which connect the matrix structure with certain combinatorial objects.

Definition 2.3. *A coloring of a matrix $A \in \Omega$ is a function $\pi_A : A_c \to \mathbb{Z}$, where A_c is the set of the columns of A. The integer $\pi_A(v)$ for $v \in A_c$ is the color of the column v.*

If $c_1 < c_2 < \cdots < c_s$ are the different colors assigned to the columns of A, $s \leq n$, we call $c = (c_1, c_2, \ldots, c_s) \in \mathbb{Z}^s$ the vector of colors of A. The coloring of a matrix A defines an ordered partition of its columns. A partition $\{V_1, V_2, \ldots, V_s\}$ of a set L is a collection of pairwise disjoint, nonempty subsets $V_1, V_2, \ldots, V_s$ of L, called cells, such that $V_1 \cup V_2 \cup \cdots \cup V_s = L$. An ordered partition is a tuple $\pi = (V_1, V_2, \ldots, V_s)$ where $\{V_1, V_2, \ldots, V_s\}$ is a partition of L. If $c = (c_1, c_2, \ldots, c_s)$ is the vector of colors of A and $V_i = \{v \in A_c : \pi_A(v) = c_i\}$, $i = 1, 2, \ldots, s$, then $\pi = (V_1, V_2, \ldots, V_s)$ is an ordered partition of the set A_c of the columns of the matrix A. We denote a colored matrix as a triple (A, π, c).

Definition 2.4. *Two colored matrices (A, π, c) and (B, σ, d) of the same size are* ***isomorphic*** *if there exists a permutation $p \in Iso(A, B)$, which maps columns of one color onto columns of the same color.*

In other words, two colored matrices are isomorphic if one can be obtained from the other by permutations of columns only within one and the same color. If two colored matrices are isomorphic, their vectors of colors coincide. The group G acting on a colored matrix is the direct product of symmetric groups S_i, where $i = |V_i|$ and $V_i \in \pi$.

The case of coloring rows is similar. The case of coloring columns and rows at the same time is more specific. It means that we should have two vectors of colors assigned to the matrix. Thus to each entry of the matrix there corresponds a pair of colors. Another approach is based on a reduction of coloring rows and columns to coloring only columns with an appropriate expansion of the matrix.

Suppose we have an $m \times n$ binary colored matrix, which is colored by columns and rows. Let $c = (c_1, c_2, \ldots, c_p)$ and $r = (r_1, r_2, \ldots, r_s)$ be the vectors of colors for columns and rows, respectively. We reduce the colorings of both columns and rows only to coloring of columns in the following way. For all $j = 1, \ldots, s$, we consider the binary representation of r_j as a binary vector denoted by b_j. All vectors b_j, $j = 1, \ldots, s$, have the same length, equal to the number μ of the binary digits of the largest integer r_i, $1 \leq i \leq s$. Then we expand the given matrix with μ columns as given in the following example:

Example 1. The 5×4 matrix A has two colors by columns and three colors by rows.

$$A = \begin{array}{cccccc} & c_1 & c_1 & c_2 & c_1 & \\ \left(\right. & 1 & 0 & 0 & 1 & \left.\right) \; r_1 \\ & 0 & 1 & 1 & 1 & r_1 \\ & 1 & 1 & 0 & 1 & r_2 \\ & 1 & 0 & 1 & 0 & r_3 \\ & 0 & 0 & 1 & 0 & r_3 \end{array}$$

$$r = (r_1, r_2, r_3) = (1, 2, 3) \quad \Rightarrow \; b_1 = 01, \; b_2 = 10, \; b_3 = 11, \quad \mu = 2.$$

The expanded 5×6 matrix A_e is colored only by columns.

$$A_e = \left(\begin{array}{cccc|cc} c_1 & c_1 & c_2 & c_1 & c_3 & c_3 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right)$$

3. Isomorphism of graphs.

Graphs are well studied combinatorial objects. A simple undirected graph G is an ordered pair $(V, E) = (V(G), E(G))$ where V is a finite set of vertices and E is a set of two-element subsets of V, called edges. The number of vertices in a graph is called its *order* and the number of edges its *size*. An edge $\{u, v\}$ *joins* the vertices u and v. Two vertices are *adjacent* if they are joined by an edge. An edge e and a vertex u are *incident* if $u \in e$.

For further information on other types of graphs and properties we refer to [16]. For the purpose of this paper we need the following definition.

Definition 3.1. *A bipartite graph is a graph G with vertex set $V(G) = V_1 \bigcup V_2$ where $V_1 \bigcap V_2 = \emptyset$. Furthermore, if $\{v_i, v_j\} \in E(G)$ then $v_i \in V_1, v_j \in V_2$. We use the notation $(\{V_1, V_2\}, E)$ for a bipartite graph with no ordering between V_1, V_2 and (V_1, V_2, E) otherwise.*

Graphs are represented either via the set of vertices and the set of edges, or via the adjacency matrix.

Definition 3.2. *Let $G = (V, E)$ be a graph with $|V| = n$ vertices labeled by $\{v_1, v_2, \ldots, v_n\}$. Subject to this labeling, the adjacency matrix of G is the $n \times n$ matrix $A = (a_{ij})$ where $a_{ij} = 1$ if $\{v_i, v_j\} \in E$ and $a_{ij} = 0$ otherwise, for $i \neq j$. The sign ∞ is put in the diagonal of the matrix.*

Since we set $a_{ii} = \infty$ for $1 \leq i \leq n$, the adjacency matrix is not exactly a binary matrix, but this setting is commonly used for practical reasons. The representation of a graph as a binary matrix and its use in isomorphism algorithms will be considered below.

Definition 3.3. *A graph G is* **isomorphic** *to a graph H if there exists a bijection $f : V(G) \to V(H)$ such that, for all $u, v \in V(G)$, we have $\{u, v\} \in E(G)$ if and only if $\{f(u), f(v)\} \in E(H)$. Such a bijection is called an isomorphism from G onto H and the set of all isomorphisms is denoted by $Iso(G, H)$. An isomorphism of G onto itself is called an* **automorphism**.

In terms of group actions two graphs G and H labeled with the same vertex set are isomorphic if and only if there exists a permutation $p \in S_{n=|V(G)|}$ such that $p(G) = H$ and $\{pv_i, pv_j\} \in E(H)$ if and only if $\{v_i, v_j\} \in E(G)$. A permutation p is said to be an automorphism of G, if $p(G) = G$. All automorphisms $p \in S_n$ of G form the group $Aut(G)$.

In terms of matrices two graphs G and H labeled with the same vertex set are isomorphic if and only if there exists a permutation $n \times n$ matrix P such that $A_H = PA_GP$ where A_G and A_H are the adjacency matrices of G and H, respectively.

An important task of computational graph theory is the storage of a graph $G = (V, E)$ in computers. One of the methods is *sequential representation*, i.e., an $|V| \times |V|$ array which represents the adjacency matrix. Another approach is *adjacency list representation*, which uses lists of neighbors. In the worst case $n(n-1)$ memory units are needed. Finally, we may store a graph via its formal definition, i.e., as a collection of vertices and edges, which needs $n + n(n-1)/2$ memory units. NAUTY's manual contains detailed information for representing graphs in the McKay algorithm [27]. For the purpose of computing isomorphisms colored graphs are one of the most used structures.

3.1. Colored graphs. Many combinatorial objects can not be successfully represented just as simple undirected graphs. For isomorphism testing we also need a coloring.

Definition 3.4. *A coloring of the graph G is a function $\pi_G : V(G) \to \mathbb{Z}$. If $c_1 < c_2 < \cdots < c_s$ are the different colors assigned to G, the vector $c = (c_1, c_2, \ldots, c_s) \in \mathbb{Z}^s$ is called vector of the colors of G.*

The coloring of a graph G defines an ordered partition of its vertex set. At the same time, a coloring can be assigned with a given ordered partition of $V(G)$. If $\pi = (V_1, V_2, \ldots, V_s)$ is an ordered partition of the given vertex set then the function $f : V(G) \to \mathbb{Z}$ defined by $f(u) = i$ if $u \in V_i$, $i = 1, \dots, s$, presents a coloring of G. Therefore we can use an equivalent definition.

Definition 3.5. *A colored graph is a triple* $(G, \pi, c) = ((V, E), \pi, c)$ *where* π *is an ordered partition of the vertex set and* c *is the corresponding vector of the colors.*

To define an isomorphism between colored graphs (G, π, c) and (H, σ, d), we use the set of isomorphisms $Iso(G, H)$ from G to H without coloring.

Definition 3.6. *Two colored graphs* (G, π, c) *and* (H, σ, d) *are isomorphic if there exists* $p \in Iso(G, H)$ *such that* $\pi_G(u) = \sigma_H(pu)$ *for all* $u \in V(G)$.

Thus an isomorphism maps vertices of one color onto vertices of the same color. In order to check whether two colored graphs are isomorphic we first test whether their vectors of colors coincide.

3.2. Graphs and binary matrices. It is not difficult to switch from a graph isomorphism problem to a binary matrix isomorphism problem since these two objects can be transformed naturally to each other.

Any binary matrix may be seen as a colored bipartite graph. For that, suppose we have an $m \times n$ binary matrix A. The rows and the columns of the matrix are denoted by $a_1, a_2, \dots, a_m$ and $b_1, b_2, \dots, b_n$, respectively. The corresponding bipartite graph $G = (V_1, V_2, E)$ has vertex set $V = V_1 \bigcup V_2$, where $V_1 = \{a_1, a_2, \dots, a_m\}$, $V_2 = \{b_1, b_2, \dots, b_n\}$, and E consists of all pairs $\{a_i, b_j\}$ for which $A_{ij} = 1$. The coloring here is the function $\pi_G : V_1 \cup V_2 \to \mathbb{Z}$ defined by $\pi_G(a_i) = c_1$ for $i = 1, 2, \dots, m$, and $\pi_G(b_j) = c_2$ for $j = 1, 2, \dots, n$, $c_1 < c_2$. Conversely, to a given bipartite graph $G = (V_1, V_2, E)$, $V_1 = \{a_1, a_2, \dots, a_m\}$, $V_2 = \{b_1, b_2, \dots, b_n\}$, we can correlate a $|V_1| \times |V_2|$ binary matrix A^G with entries $A^G_{ij} = 1$ if $\{a_i, b_j\} \in E$, and $A^G_{ij} = 0$ otherwise. This observation shows that solving the isomorphism problems for bipartite graphs and binary matrices is the same.

On the other hand, any graph can be made bipartite by replacing each edge by two edges connected with a new vertex, and then represented as a binary matrix. Any two graphs are isomorphic if and only if the transformed bipartite graphs are, and any two graphs are isomorphic if and only if the corresponding binary matrices are.

The storage of an $m \times n$ binary matrix A needs $n.m$ computer memory units whereas the corresponding graph $G_A = (V, E), |V| = n + m$ needs $(n + m)^2$. A graph $G = (V, E)$ with $|V| = n$ and $|E| = m$ can be transformed into a bipartite graph $G_b = (V, E, E')$, where $E = \{e_1, e_2, \dots, e_m\}$ and

$E' = \{e'_1, e''_1, e'_2, e''_2, \dots, e'_m, e''_m\}$. This graph also needs $(n+m)^2$ memory cells. This representation is very natural but not so efficient. That's why we give a representation of graphs as colored binary matrices.

Let $G = (V, E)$ be a simple undirected graph with vertex set $V = \{v_1, v_2, \dots, v_n\}$, edge set $E = \{\{v_{i_1}, v_{j_1}\}, \{v_{i_2}, v_{j_2}\}, \dots\}$, and adjacency matrix A_G. Our aim is to define a map which associates a binary matrix to a given graph such that two graphs G and H are isomorphic if and only if their corresponding matrices are isomorphic.

Using the adjacency matrix A_G we define an $2n \times 2n$ binary matrix A_{Gb} by replacing every entry by two digits in the following way: $0 \to 00$, $1 \to 01$, $\infty \to 11$. In addition we put n more rows as it is shown in (1). The reason to have these rows is to keep the pairs corresponding to one entry of A_G together. After that the rows of A_{Gb} are colored with two colors r_1 and r_2 so that the color of the first n rows is r_1 and the color for the new added rows is $r_2 > r_1$.

$$A_{Gb} = \left(\begin{array}{cccc} 11 & 0a_{12} & \dots & 0a_{1n} \\ 0a_{21} & 11 & \dots & 0a_{2n} \\ & & \dots & \\ 0a_{n1} & 0a_{n2} & \dots & 11 \\ \hline 11 & 00 & \dots & 00 \\ 00 & 11 & \dots & 00 \\ & & \dots & \\ 00 & 00 & \dots & 11 \end{array}\right) \quad \begin{array}{c} r_1 \\ r_1 \\ \dots \\ r_1 \\ \hline r_2 \\ r_2 \\ \dots \\ r_2 \end{array} \tag{1}$$

Theorem 3.1. *Two simple graphs G and H are isomorphic, if and only if the colored binary matrices A_{Gb} and A_{Hb} are.*

Proof. Let A_G and A_H be the adjacency matrices of both graphs.

$\Rightarrow$) Suppose that $G \cong H$. Then there exists a permutation matrix P such that $A_H = PA_GP$. It follows that $A_{Hb} = \overline{P} A_{Gb} \widehat{P}$ where $\widehat{P}$ is the $2n \times 2n$ permutation matrix, obtained from P by replacing every 1 by the identity matrix I_2 and every 0 by the 2×2 zero matrix, and $\overline{P} = \left(\begin{array}{c|c} P & O \\ \hline O & P \end{array}\right)$. It turns out that the binary matrices A_{Gb} and A_{Hb} are isomorphic.

$\Leftarrow$) Suppose that A_{Gb} and A_{Hb} are isomorphic. Hence there are $2n \times 2n$ permutation matrices M and Q such that $A_{Hb} = MA_{Gb}Q$. Since the matrices are colored, M has the form $M = \left(\begin{array}{c|c} P & O \\ \hline O & P' \end{array}\right)$, where P and P' are $n \times n$ permutation matrices. So

$$(2)\quad A_{Hb} = \left(\begin{array}{cccc} 11 & 0b_{12} & \dots & 0b_{1n} \\ 0b_{21} & 11 & \dots & 0b_{2n} \\ & & \dots & \\ 0b_{n1} & 0b_{n2} & \dots & 11 \\ \hline 11 & 00 & \dots & 00 \\ 00 & 11 & \dots & 00 \\ & & \dots & \\ 00 & 00 & \dots & 11 \end{array}\right) = MA_{Gb}Q$$

$$= M \left(\begin{array}{cccc} 11 & 0a_{12} & \dots & 0a_{1n} \\ 0a_{21} & 11 & \dots & 0a_{2n} \\ & & \dots & \\ 0a_{n1} & 0a_{n2} & \dots & 11 \\ \hline 11 & 00 & \dots & 00 \\ 00 & 11 & \dots & 00 \\ & & \dots & \\ 00 & 00 & \dots & 11 \end{array}\right) Q,$$

Obviously, a pair of columns with numbers $(2i-1, 2i)$ goes to another pair $(2j-1, 2j)$, $1 \leq i, j \leq n$. Therefore, the matrix Q is a permutation matrix of the same form as described above (as the matrix $\widehat{P}$). If σ is the permutation corresponding to Q acting on the set of pairs of columns, then the matrix P has to act in the same way as σ to the first n rows because the diagonals of both matrices A_G and A_H are the same. Then P is the permutation matrix corresponding to σ. Moreover $P' = P$ and therefore we have $Q = \widehat{P}$ and $M = \left(\begin{array}{c|c} P & O \\ \hline O & P \end{array}\right)$. This gives us that $A_H = PA_GP$ and the graphs G and H are isomorphic. □

For directed graphs the same approach works similarly.

3.3. Directed graphs and binary matrices.

Definition 3.7. *A directed graph G, also called a digraph, is an ordered pair (V, A), where V is a finite set of vertices or nodes and A is a set of ordered pairs of nodes, called directed edges or arcs.*

An arc $a = (x, y)$ is considered to be directed from x to y, where y is said to be a direct successor of x, and x is said to be a direct predecessor of y. The number of all direct successors of a node x is called *outdegree* and denoted by $d_{out}(x)$. The number of all direct predecessors of a node y is called *indegree* and denoted by $d_{in}(y)$. The arc (y, x) is called the inverted arc (x, y). A directed graph D is *symmetric* if, for every arc $(x, y) \in A$, the corresponding inverted arc $(y, x) \in A$. An arc $a = (x, x)$ is called a loop. A symmetric loopless directed graph $D = (V, A)$ is equivalent to a simple undirected graph $G = (V, E)$, where the pairs of inverse arcs in A correspond 1-to-1 to the edges in E.

A digraph can also be represented by its adjacency matrix A_D, which, unlike simple graphs, is not symmetric in general. If $(v_i, v_j) \in A$, then $a_{ij} = 1$, while $a_{ji} = 0$, unless the inverted arc is also in A. The number of ones in the i^{th} row is the outdegree of v_i and the number of ones in the j^{th} column is the indegree of v_j.

Obtaining a binary matrix A_{Db} from a digraph D is analogous to obtaining a binary matrix from a simple graph.

Theorem 3.2. *Two directed graphs D and Q are isomorphic, if and only if the binary matrices A_{Db} and A_{Qb} are.*

Proof. Analogical to the proof in the case of isomorphic simple graphs. □

4. Codes and binary matrices. Classifying codes up to equivalence is one of the main subjects in coding theory. In many cases the equivalence problem for codes is reduced to the graph isomorphism problem. Some methods and implementations are given in [16]. In our work, we consider the equivalence problem for codes via binary matrices.

4.1. Nonlinear codes representation. Let A be a finite alphabet of cardinality q. An (n, M) nonlinear q-ary code C is a set of M words of length n over A. A nonlinear code C is often given by its codewords which can be put into an $M \times n$ matrix. Without loss of generality, we can fix $A = \mathbb{Z}_q = \{0, 1, \dots, q-1\}$.

Definition 4.1. [16] *Two nonlinear codes are said to be equivalent if one can be transformed into the other by a permutation of the coordinates in the codewords followed by permutations of the coordinate values, independently for each coordinate.*

Some authors represent the nonlinear codes as graphs in order to test them for equivalence (see for example [16]). Let C be a q-ary (n, M) code. We correlate to C a colored graph $G_C = (V_C, E_C)$ in the following way. We consider the codewords as vertices colored in one color, say r_1, and add nq more vertices, colored in another color $r_2 \neq r_1$. The vertices colored in r_2 are partitioned into n subsets $V_1, \dots, V_n$, each of them with q elements corresponding to the letters of the alphabet A. For any i, $1 \leq i \leq n$, the subgraph of G_C with vertex set V_i is complete. Furthermore, the vertex corresponding to the codeword $v = (a_1, a_2, \ldots, a_n)$ is connected to the vertex a_i from V_i for all $i = 1, 2, \ldots, n$. Hence the graph G_C has $M + nq$ vertices and $nq(q-1)/2 + Mn$ edges.

We use here a different approach. Our aim is to reduce the equivalence problem for codes to test for isomorphism of binary matrices. That's why we need a representation of codes as binary matrices.

We correlate a binary vector of length q to any element of $A = \mathbb{Z}_q$, such that $0 \mapsto (10\ldots0)$, $1 \mapsto (010\ldots0)$, $\ldots$, $q-1 \mapsto (0\ldots01)$. Let C be a q-ary (n, M) code, given by an $M \times n$ matrix of its codewords. As C is a q-ary code we transform each entry a_{ij} of the matrix to its corresponding binary vector. Then we expand the matrix with n more rows in order to mark the columns, which represent the different coordinates. This means that the $M + i$ th row has ones in positions $q(i-1)+1, q(i-1)+2, \ldots, qi$, $i = 1, 2, \ldots, n$. To distinguish the rows corresponding to codewords from the additional ones we color all rows in two different colors r_1 and r_2.

Example 2. Let C be a $(4, 2, 3)$ ternary code.

$$C = \begin{pmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 2 & 2 \end{pmatrix}$$

We obtain the binary matrix $C^{'}$ by replacing every coordinate with three digits.

$$C^{'} = \begin{pmatrix} 100 & 010 & 010 & 001 \\ 010 & 100 & 001 & 001 \end{pmatrix}$$

After that four extra rows are added. So we obtain the binary matrix C_b.

$$C_b = \left(\begin{array}{cccc} 100 & 010 & 010 & 001 \\ 010 & 100 & 001 & 001 \\ \hline 111 & 000 & 000 & 000 \\ 000 & 111 & 000 & 000 \\ 000 & 000 & 111 & 000 \\ 000 & 000 & 000 & 111 \end{array}\right) \begin{array}{c} r_1 \\ r_1 \\ r_2 \\ r_2 \\ r_2 \\ r_2 \end{array}$$

Theorem 4.1. *Two nonlinear (n, M) codes C_1 and C_2 are equivalent if and only if the corresponding binary matrices C_{1b} and C_{2b} are isomorphic.*

Proof.

$\Rightarrow$) If $C_1 \cong C_2$, then there is a sequence of the following transformations, which maps the code C_1 onto the code C_2: a permutation $p \in S_n$ of the coordinates and permutations $p_1, p_2, \ldots, p_n \in S_q$ of the values of the corresponding coordinates. Consider now the matrices C_{1b} and C_{2b}. The permutation p gives a permutation $\hat{p} \in S_{nq}$ which acts as p on the family consisting of the sets of columns labeled by $\{i, q+i, \dots, (n-1)q+i\}$ of the matrix C_{1b}, $i = 1, 2, \dots, q$. For any $j = 1, 2, \dots, n$, the permutation $p_j \in S_q$ of the letters of the alphabet A can be considered as a permutation of the set of columns $\{(j-1)q+1, (j-1)q+2, \dots, jq\}$. After applying these permutations on the matrix C_{1b}, we obtain the matrix PC_{2b} where P is a permutation $(M+n) \times (M+n)$-matrix which permutes the first M rows of C_{2b}. Hence the matrices C_{1b} and C_{2b} are isomorphic.

$\Leftarrow$) Suppose that the matrices C_{1b} and C_{2b} are isomorphic which means that there are permutation matrices P and Q of the appropriate size such that $C_{2b} = PC_{1b}Q$. Since both matrices are colored, P induces a permutation which permutes the sets of the first M rows and the last n rows of C_{1b} independently. Having in mind the structure of the last n rows, the matrix Q gives a permutation which can be considered as a product of a permutation of the set consisting of the supports of the last n rows and the permutations of the set of columns within any of these supports. This means that the corresponding codes C_1 and C_2 are equivalent. □

4.2. Isomorphisms of linear codes. The equivalence problem of linear codes has been considered in many papers. We distinguish the works of Leon [20], Sendrier [28], Petrank and Roth [30], Sendrier and Simos [29].

Let $\mathbb{F}_q^n$ be the n-dimensional vector space over the field $\mathbb{F}_q$ of q elements. The (Hamming) weight $\mathrm{wt}(v)$ of $v \in \mathbb{F}_q^n$ is the number of its nonzero coordinates. The Hamming distance $d(u, v)$ between two words u and v is the number of coordinates in which they differ. A *q-ary linear $[n, k, d]_q$ code* is a k-dimensional linear subspace of $\mathbb{F}_q^n$ with minimum distance d.

Definition 4.2. *We say that two linear $[n, k]_q$ codes C_1 and C_2 are equivalent, if the codewords of C_2 can be obtained from the codewords of C_1 via a finite sequence of transformations of the following types:*

(1) *Permutation of coordinate positions.*

(2) *Multiplication of the elements in a given coordinate position by a nonzero element of $\mathbb{F}_q$.*

(3) *Application of a field automorphism to the elements in all coordinate positions.*

This definition is well motivated as the transformations (1–3) preserve the Hamming distance. The first two transformations are linear, the third is semilinear. Moreover, a sequence of operations (1) and (2) is equivalent to the right multiplication of the codewords of C with an appropriate monomial matrix. Such a matrix contains exactly one nonzero element of $\mathbb{F}_q$ in each row and column. So, two $[n,k]_q$ linear codes C_1 and C_2 are equivalent if there exists a matrix $M \in Mon(n,q)$, the monomial group, and an automorphism γ of the field $\mathbb{F}_q$, for which $C_1 M\gamma = C_2$, or $c^{(M,\gamma)} = cM\gamma \in C_2$ for each $c \in C_1$.

An *automorphism* of a linear code C is a finite sequence of transformations of type (1), (2), (3) which maps each codeword of C onto a codeword of C.

A *generator matrix* G of a linear $[n,k]$ code C is a $k \times n$ matrix whose row vectors form a basis of the code. In particular G has rank k. The connection between generator matrices of two equivalent codes is given in the following theorem.

Theorem 4.2. *[22] Let p be a prime and C_1 and C_2 be two linear $[n,k]$ codes over $\mathbb{F}_p$ with generator matrices G_1 and G_2. Then $C_1 \cong C_2$ iff there exists a nonsingular $k \times k$ matrix B and a monomial $n \times n$ matrix M, both over $\mathbb{F}_p$, such that $BG_1M = G_2$. An automorphism of a linear code C with generator matrix G is an ordered pair (B,M) such that $BGM = G$.*

The left multiplication of G with a nonsingular matrix produces k other linearly independent codewords. Hence the product BG is just another generator matrix of the same linear code C. The right multiplication with the matrix M gives a sequence of transformations of types (1) and (2) of the classic definition.

4.3. Representation of linear code equivalence as isomorphism of binary matrices. Representing linear codes over prime fields as graphs is described in [16] and the methods are close to the idea of the nonlinear code representation. We know of no-such type of representation for codes over composite fields.

Representing a linear code as a binary matrix has already been introduced by the first author in [3]. We present here a different approach which is more general and more suitable for codes over fields with $q \geq 5$ elements. As we also use integer matrices, we need a definition of the equivalence of such matrices.

Definition 4.3. *Two integer matrices A and B of the same size are **isomorphic** ($A \cong B$) if the rows of the second one can be obtained from the rows of the first one by a permutation of the columns. All isomorphisms form the set $Iso(A,B)$.*

The next definition which is equivalent to the first one is more convenient for us in some cases.

Definition 4.4. *Two integer $m \times n$ matrices A and B are* ***isomorphic*** *($A \cong B$) if there is a permutation $m \times m$ matrix P and another permutation $n \times n$ matrix Q such that $AQ = PB$ or $B = P^{-1}AQ$.*

We use in our representation integer matrices with only three different elements, namely 0, 1 and 2, that's why we call them ternary matrices.

Let C be a linear code over a field $\mathbb{F}_q$ with $q = p^m$ where p is the characteristic of the field, and let α be a primitive element of $\mathbb{F}_q$. We map any nonzero element α^j of the field, $0 \leq j \leq q-2$, to a $2(q-1) \times 2(q-1)$ binary matrix A_j in the following way:

1. We define the map $\pi : \mathbb{F}_q^* \to \mathbb{F}_3$, where $\pi(1) = 1$, $\pi(\alpha) = \pi(\alpha^p) = \cdots = \pi(\alpha^{p^{m-1}}) = 2$, and $\pi(\beta) = 0$ if $\beta \in \mathbb{F}_q^*$, $\beta \neq \alpha^{p^s}$.

2. We correlate to the unity of the field the ternary circulant matrix A_0^* with first row $(1, \pi(\alpha), \pi(\alpha^2), \ldots, \pi(\alpha^{q-2}))$. Then we map the element α^j to the ternary circulant A_j^* with first row $(\pi(\alpha^{q-1-j}), \ldots, \pi(\alpha^{q-2}), 1, \pi(\alpha), \ldots, \pi(\alpha^{q-2-j}))$ which is obtained from the first row of A_0^* by cyclic shift with j positions. This means that $A_j^* = A_0^* P_j^* = A_0^*(P_1^*)^j$ where P_j^* is the permutation circulant matrix with first row with 1 in the $j+1$-th position, $j = 0, 1, \ldots, q-2$.

3. Let us define another map $\rho : \mathbb{F}_3 \to \mathbb{F}_2^2$ where $\rho(0) = (00)$, $\rho(1) = (10)$ and $\rho(2) = (11)$. Denote by A_j' the $(q-1) \times 2(q-1)$ binary matrix $\rho(A_j^*)$ which is obtained by replacing any element of A_j^* by its image under the map ρ. Now $A_j' = A_0' P_j' = A_0'(P_1')^j$ where P_j' is obtained from P_j^* as the 1s are replaced by the 2×2 identity matrix and the 0s by the 2×2 zero matrix.

4. To have the matrix A_j we add $q-1$ more rows to A_j', namely

$$(11000\ldots00), (00110\ldots00), \ldots, (0000\ldots, 011).$$

Let us illustrate this representation by two examples.

Example 3. Let $q = 5$. We can take $\alpha = 2$ or 3 and consider $\mathbb{F}_5 =$

$\{0, 1, \alpha, \alpha^2, \alpha^3\}$. Then

$$1 \mapsto \begin{pmatrix} 10110000 \\ 00101100 \\ 00001011 \\ 11000010 \\ 11000000 \\ 00110000 \\ 00001100 \\ 00000011 \end{pmatrix}, \quad \alpha \mapsto \begin{pmatrix} 00101100 \\ 00001011 \\ 11000010 \\ 10110000 \\ 11000000 \\ 00110000 \\ 00001100 \\ 00000011 \end{pmatrix}, \alpha^2 \mapsto \begin{pmatrix} 00001011 \\ 11000010 \\ 10110000 \\ 00101100 \\ 11000000 \\ 00110000 \\ 00001100 \\ 00000011 \end{pmatrix}, \quad \alpha^3 \mapsto \begin{pmatrix} 11000010 \\ 10110000 \\ 00101100 \\ 00001011 \\ 11000000 \\ 00110000 \\ 00001100 \\ 00000011 \end{pmatrix}.$$

Example 4. Let $q = 8$ and $\mathbb{F}_q = \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}$. Then

$$1 = \alpha^0 \mapsto A_0^* = \begin{pmatrix} 1220200 \\ 0122020 \\ 0012202 \\ 2001220 \\ 0200122 \\ 2020012 \\ 2202001 \end{pmatrix} \mapsto A_0' = \begin{pmatrix} 10111100110000 \\ 00101111001100 \\ 00001011110011 \\ 11000010111100 \\ 00110000101111 \\ 11001100001011 \\ 11110011000010 \end{pmatrix} \mapsto A_0 = \begin{pmatrix} 10111100110000 \\ 00101111001100 \\ 00001011110011 \\ 11000010111100 \\ 00110000101111 \\ 11001100001011 \\ 11110011000010 \\ 11000000000000 \\ 00110000000000 \\ 00001100000000 \\ 00000011000000 \\ 00000000110000 \\ 00000000001100 \\ 00000000000011 \end{pmatrix}$$

Firstly, we would like to find the automorphism group of the matrix A_0. Obviously, $\mathrm{Aut}(A_j) = \mathrm{Aut}(A_0)$ for $j = 1, 2, \ldots, q-2$. Moreover, since the last $q-1$ rows in A_j have weight 2, and the weights of the rows in A_j' are ≥ 3 for $q \geq 3$ (and 1 for $q = 2$), any automorphism of A_j will permute separately the last $q-1$ rows of the matrix.

Lemma 4.3. $\mathrm{Aut}(A_0) \cong \mathrm{Aut}(A_0^*)$.

Proof. Since A_0^* and A_0 are square matrices of size $q-1$ and $2(q-1)$, respectively, then $\mathrm{Aut}(A_0^*) \leq S_{q-1}$ and $\mathrm{Aut}(A_0) \leq S_{2(q-1)}$. Let $\varphi : S_{q-1} \to S_{2(q-1)}$ be the monomorphism defined in the following way: if $\sigma = \sigma_1\sigma_2\cdots\sigma_s$ where $\sigma_1, \sigma_2, \ldots, \sigma_s$ are independent cycles, then $\varphi(\sigma) = \sigma_1'\sigma_1''\sigma_2'\sigma_2''\cdots\sigma_s'\sigma_s''$ where if $\sigma_j = (i_1, i_2, \ldots, i_t)$ then $\sigma_j' = (2i_1 - 1, 2i_2 - 1, \ldots, 2i_t - 1)$ and $\sigma_j'' = (2i_1, 2i_2, \ldots, 2i_t)$. We will prove that $\mathrm{Aut}(A_0) = \varphi(\mathrm{Aut}(A_0^*))$.

Obviously, if $\sigma \in \mathrm{Aut}(A_0^*)$ then $\varphi(\sigma) \in \mathrm{Aut}(A_0)$ and thus $\varphi(\mathrm{Aut}(A_0^*)) \leq \mathrm{Aut}(A_0)$. Now let $\tau \in \mathrm{Aut}(A_0)$. If $\tau(2i-1) = 2j$ then the image of the i th row from the added rows with two 1s will be the j th from these rows and so $\tau(2i) = 2j - 1$. But then looking at the first $q - 1$ rows we see that in the image of the i th row of A_0 the pair in columns $2j - 1$ and $2j$ is (01). Since no row has such a pair in this couple of columns, the automorphism τ cannot map a column with an even number to a column with an odd number. Hence $\tau(2i - 1) = 2j - 1$ and $\tau(2i) = 2j$ for some $i, j \in \{1, 2, \dots, q - 1\}$. This means that any ordered pair of columns goes to another ordered pair of columns in such a way that the image of any row from the set of the first $q - 1$ rows of A_0 is another row from the same set. Therefore $\tau \in Im\varphi$ and moreover $\varphi^{-1}(\tau)$ is an automorphism of the matrix A_0^*. This proves that $\mathrm{Aut}(A_0) = \varphi(\mathrm{Aut}(A_0^*))$ and so $\mathrm{Aut}(A_0) \cong \mathrm{Aut}(A_0^*)$. □

Lemma 4.4. *The automorphism group of the trivial code of length 1 over* $\mathbb{F}_q$ *is a subgroup of* $\mathrm{Aut}(A_0^*)$. *For* $q \leq 16$ *both groups coincide.*

Proof. If G is the automorphism group of the trivial code of length 1 then $G = \langle \sigma, \phi \rangle$ where σ is the multiplication of any element by α and ϕ is the Frobenius automorphism which generate the automorphism group of the field. We can consider the elements of G as permutation of $(\alpha^0, \alpha^1, \alpha^2, \dots, \alpha^{q-2})$, i.e., $G \leq S_{q-1}$.

Let us label the rows (the columns) of the matrix A_0^* from 0 to $q-2$. Then the columns correspond to the elements of the field ordered as $(\alpha^0, \alpha^1, \alpha^2, \dots, \alpha^{q-2})$. Besides, the i th row corresponds to the set $\{\alpha^i, \alpha^{1+i}, \alpha^{p+i}, \dots, \alpha^{p^{m-1}+i}\}$, as a 1 is in position i and 2-s are in the positions, corresponding to the other elements from the set. We can consider σ as a permutation of the columns of A_0^* such that $\sigma = (0, 1, 2, \dots, q - 2)$. Obviously, $\sigma \in \mathrm{Aut}(A_0^*)$.

The automorphism group of the field with $q = p^m$ elements is the cyclic group of order m which is generated by the Frobenius automorphism ϕ defined by $\phi(a) = a^p$, $a \in \mathbb{F}_q$. Let us apply this map to the elements of the matrix A_0^*. The permutation of the columns, corresponding to ϕ is a product of independent cycles such that each cycle corresponds to a cyclotomic class modulo $q - 1$. We have

$$\phi(\alpha^i) = \alpha^{ip}, \phi(\alpha^{1+i}) = \alpha^{p+ip}, \phi(\alpha^{p+i}) = \alpha^{p^2+ip}, \dots, \phi(\alpha^{p^{m-1}+i}) = \alpha^{p^m+ip} = \alpha^{1+ip}.$$

Hence the image of the i-th row is the row with number ip modulo $q-1$. Therefore we can consider the automorphism group of the field as a subgroup of $\mathrm{Aut}(A_0^*)$ and more precisely of the stabilizer of the column with number 0.

By a computer check we verified that the orders of G and $\mathrm{Aut}(A_0^*)$ are equal for the fields with $q \leq 16$ elements. Hence for $q \leq 16$ both groups coincide. □

For all other statements in this section we consider codes over fields with $q \leq 16$ elements.

Let B_i be the set of all codewords in C of weight i, $d \leq i \leq n$. We take the set $\mathcal{B} = B_{d_1} \cup B_{d_2} \cup \dots \cup B_{d_t}$ with the following properties:

1. $d = d_1 < d_2 < \dots < d_t \leq n$,

2. $B_i = \emptyset$ for $d_j < i < d_{j+1}$, $j = 1, \dots, t-1$,

3. $\mathcal{B}$ generates C as a vector space, but $\mathcal{B} \setminus B_{d_t}$ does not generate the code.

To represent a linear code C, we use the subset $\mathcal{B}$ of C. It is stable under the action of $Aut(C)$.

Obviously, if the vector $a \in \mathcal{B}$, then the vector λa for $\lambda \in \mathbb{F}_q \setminus \{0\}$ is also in $\mathcal{B}$. Let $\mathcal{B}' = \{b'_1, b'_2, \dots, b'_K\}$ be a subset of $\mathcal{B}$ such that no two vectors $b'_i, b'_j \in \mathcal{B}'$ are proportional for $i \neq j$, and for any vector $b \in \mathcal{B}$ there is a constant $\lambda \in \mathbb{F}_q \setminus \{0\}$ for which $\lambda b \in \mathcal{B}'$. Let A' be the matrix whose rows are the vectors from $\mathcal{B}'$. To avoid a repetition of rows, to any element α^j in the matrix A' we correlate the matrix A_j^* defined in the beginning of this section. More convenient for us is to denote the matrix A_j^* which corresponds to the element b_{ij} by A_{ij}^* where $b_i = (b_{i1}, b_{i2}, \dots, b_{in})$. Moreover, we add some extra rows and extra columns in the following way to have the following ternary $K(q-1)+n \times K+n(q-1)$ matrix:

$$D_C^* = \left(\begin{array}{cccc|cccc}
A_{1,1}^* & A_{1,2}^* & \dots & A_{1,n}^* & 1 & 0 & \dots & 0 \\
A_{2,1}^* & A_{2,2}^* & \dots & A_{2,n}^* & 0 & 1 & \dots & 0 \\
\dots & \dots & \dots & \dots & & & \dots & \\
A_{K,1}^* & A_{K,2}^* & \dots & A_{K,n}^* & 0 & 0 & \dots & 1 \\
\hline
11\dots1 & 00\dots0 & \dots & 00\dots0 & 0 & 0 & \dots & 0 \\
00\dots0 & 11\dots1 & \dots & 00\dots0 & 0 & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & & & \dots & \\
\underbrace{00\dots0}_{q-1} & \underbrace{00\dots0}_{q-1} & \dots & \underbrace{11\dots1}_{q-1} & 0 & 0 & \dots & 0
\end{array}\right)$$

The last n rows guarantee that an automorphism σ will map any set of $(q-1)$ columns of D_C^* to another block of $q-1$ columns.

From this matrix using the map ρ and adding some more rows we obtain a binary colored matrix

$$D_C = \left(\begin{array}{cccc|cccc}
A'_{1,1} & A'_{1,2} & \dots & A'_{1,n} & 1 & 0 & \dots & 0 \\
A'_{2,1} & A'_{2,2} & \dots & A'_{2,n} & 0 & 1 & \dots & 0 \\
\dots & \dots & \dots & \dots & & & \dots & \\
A'_{K,1} & A'_{K,2} & \dots & A'_{K,n} & 0 & 0 & \dots & 1 \\
\hline
L_{q-1} & 00\dots0 & \dots & 00\dots0 & 0 & 0 & \dots & 0 \\
00\dots0 & L_{q-1} & \dots & 00\dots0 & 0 & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & & & \dots & \\
00\dots0 & 00\dots0 & \dots & L_{q-1} & 0 & 0 & \dots & 0 \\
\hline
11\dots1 & 00\dots0 & \dots & 00\dots0 & 0 & 0 & \dots & 0 \\
00\dots0 & 11\dots1 & \dots & 00\dots0 & 0 & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & & & \dots & \\
\underbrace{00\dots0}_{2(q-1)} & \underbrace{00\dots0}_{2(q-1)} & \dots & \underbrace{11\dots1}_{2(q-1)} & 0 & 0 & \dots & 0
\end{array}\right)
\begin{array}{c}
c_1 \\ c_1 \\ \dots \\ c_1 \\ \hline c_2 \\ c_2 \\ \dots \\ c_2 \\ \hline c_3 \\ c_3 \\ \dots \\ c_3 \\
\end{array}$$

where L_{q-1} is the $(q-1) \times 2(q-1)$ binary matrix

$$L_{q-1} = \begin{pmatrix} 11000\dots000 \\ 00110\dots000 \\ \dots \\ 00000\dots011 \end{pmatrix}.$$

Moreover, the 1's in the last K columns are actually the columns $(11\dots1)^T$.

Similarly to Lemma 4.3 we can prove that $\mathrm{Aut}(D^*_C) \cong \mathrm{Aut}(D_C)$. Let for $j = 1, 2, \dots, n$, B^*_j (B_j) be the set of columns of the matrix D^*_C (D_C) which correspond to the matrices A^*_{ij} (A'_{ij}), $i = 1, \dots, K$.

Lemma 4.5. $\mathrm{Aut}(D^*_C) \cong \mathrm{Aut}(D_C)$.

Proof. Let us extend the map $\varphi : S_{q-1} \to S_{2(q-1)}$ defined in the proof of Lemma 4.3 to $\overline{\varphi} : S_{n(q-1)} \to S_{2n(q-1)}$ and then to $\overline{\overline{\varphi}} : S_{n(q-1)} \times S_K \to S_{2n(q-1)} \times S_K$ in the following way: If $\sigma = \sigma_1\sigma_2 \cdots \sigma_s$ where $\sigma_1, \sigma_2, \dots, \sigma_s$ are independent cycles, then $\overline{\varphi}(\sigma) = \sigma'_1\sigma''_1\sigma'_2\sigma''_2 \cdots \sigma'_s\sigma''_s$ obtained in the same way as in the definition of φ. Then we have $\overline{\overline{\varphi}}(\sigma\tau) = \overline{\varphi}(\sigma)\tau$ where $\sigma \in S_{n(q-1)}$, $\tau \in S_K$. Obviously, $\overline{\overline{\varphi}}$ is a monomorphism. Considering the permutations as matrices, $\overline{\overline{\varphi}}$ acts as

$$P = \left(\begin{array}{c|c} P_1 & 0 \\ \hline 0 & P_2 \end{array}\right) \mapsto \overline{P} = \left(\begin{array}{c|c} \overline{P_1} & 0 \\ \hline 0 & P_2 \end{array}\right),$$

where $\overline{P_1}$ is obtained from P_1 as all 1's are replaced by the identity matrix I_2.

Now let us consider $\tau \in \mathrm{Aut}(D_C^*)$. This means that τ is a permutation of the columns which maps a row of the matrix to another row of the same matrix. If $\tau((j-1)(q-1)+i) = x$, $1 \le j \le n$, $1 \le i \le (q-1)$, then $x \le n(q-1)$ and furthermore $\tau(B_j) = B_s$ for some $s \le n$. This is true because the image of the j th from the last n rows should be one of these last rows, say s-th. It follows that τ acts as a permutation on the family of sets $\{B_1^*, \dots, B_n^*\}$ and so $\mathrm{Aut}(D_C^*) < S_{n(q-1)} \times S_K$. Similarly, $\mathrm{Aut}(D_C) < S_{2n(q-1)} \times S_K$.

Obviously, if $\sigma \in \mathrm{Aut}(D_C^*)$ then $\overline{\overline{\varphi}}(\sigma) \in \mathrm{Aut}(D_C)$ and thus $\overline{\overline{\varphi}}(\mathrm{Aut}(D_C^*)) \le \mathrm{Aut}(D_C)$. Now let $\tau \in \mathrm{Aut}(D_C)$ and $\tau = \tau_1\tau_2$, $\tau_1 \in S_{2n(q-1)}$, $\tau_2 \in S_K$. If $\tau_1(2(i-1)(q-1)+2s-1) = 2(j-1)(q-1)+2r$ then the image of the s th row from the i th matrix L_{q-1} will be the r th row of the j th L_{q-1} and so $\tau_1(2(i-1)(q-1)+2s) = 2(j-1)(q-1)+2r-1$. But then in the image of a row with the proper number of D_C the pair in columns $2(j-1)(q-1)+2r-1$ and $2(j-1)(q-1)+2r$ is (01), which is not possible. Hence $\tau(2i-1) = 2j-1$ and $\tau(2i) = 2j$ for some $i, j \in \{1, 2, \dots, n(q-1)\}$. This means that any ordered pair of columns goes to another ordered pair of columns in such a way that the image of any row from the set of the first $2n(q-1)$ rows of D_C is another row from the same set. Therefore $\tau_1 \in Im\overline{\varphi}$ and moreover $\overline{\varphi}^{-1}(\tau_1)\tau_2$ is an automorphism of the matrix D_C^*. This proves that $\mathrm{Aut}(D_C) = \overline{\overline{\varphi}}(\mathrm{Aut}(D_C^*))$ and so $\mathrm{Aut}(D_C) \cong \mathrm{Aut}(D_C^*)$. □

Theorem 4.6. *The automorphism group of the q-ary linear code C is isomorphic to the automorphism group of the binary matrix D_C.*

Proof. We shall prove that $\mathrm{Aut}(C) \cong \mathrm{Aut}(D_C^*)$. Recall that B_i^* is the set of columns with numbers from $(i-1)(q-1)+1$ to $i(q-1)$, $i = 1, 2, \dots, n$. Any permutation of the coordinate positions of the code C is a permutation of the family of sets of columns $B_1^*, \dots, B_n^*$ in D_C^*.

The multiplication of the elements in a given position in each codeword by a nonzero element of $\mathbb{F}_q$ acts on the matrix as a permutation of the coordinates in the corresponding set B_j^* so that the multiplication by α correspond to the permutation $(1, 2, \dots, q-1)$ of these columns.

An application of a field automorphism to the elements in all coordinate positions gives a permutation in the set of the matrices A_j^*. So a combination of these three transformations gives a permutation of the columns of D_C^*.

If a sequence of the above transformations maps any codeword of C to another codeword, the corresponding permutation of the columns of D_C^* will map a row to another row of the same matrix, having in mind that the subset $\mathcal{B}$ of C is stable under the action of $Aut(C)$ and how the matrix D_C^* is constructed. That's why any automorphism of C gives an automorphism of the corresponding matrix D^*_C.

Now let us consider the opposite case and $\tau \in \mathrm{Aut}(D^*_C)$. This means that τ is a permutation of the columns which maps a row of the matrix to another row of the same matrix. As we already mentioned $\tau = \tau_1\tau_2$ where $\tau_1 \in S_{n(q-1)}$, $\tau_2 \in S_K$. Moreover, τ acts as a permutation on the family $\{B_1, \dots, B_n\}$. Then if P_τ is the permutation matrix corresponding to τ, it has the form

$$P_\tau = \left(\begin{array}{c|c} P^*_\tau & 0 \\ \hline 0 & P_{add} \end{array}\right),$$

where P^*_τ can be considered as a monomial $n \times n$ matrix whose nonzero elements are permutation $(q-1) \times (q-1)$ matrices, and P_{add} is a permutation $K \times K$ matrix. Denote the permutation matrix in the j th column of P^*_τ by P_j. Then

$$(A^*_{1,1}, A^*_{1,2}, \dots, A^*_{1,n}, 10\dots0)P_\tau = (A^*_{1,i_1}P_1, A^*_{1,i_2}P_2, \dots, A^*_{1,i_n}P_n, (10\dots0)P_{add})$$

Since τ is an automorphism of the matrix, we have

$$\begin{aligned}(A^*_{1,i_1}P_1, A^*_{1,i_2}P_2, \dots, A^*_{1,i_n}P_n, (10\dots0)P_{add})& \\ = P_r(A^*_{l,1}, A^*_{l,2}, \dots, A^*_{l,n}, 00\dots010\dots0).&\end{aligned}$$

If $A^*_{ij} = A^*_0P'_{ij}$ then $A^*_{1,i_s}P_s = A^*_0P'_{1,i_s}P_s = P_rA^*_0P'_{l,s}$. It follows that

$$A^*_0P'_{1,i_s}P_s(P'_{l,s})^{-1} = P_rA^*_0 \Rightarrow P'_{1,i_s}P_s(P'_{l,s})^{-1} \in \mathrm{Aut}(A^*_0).$$

Hence $P_s \in \mathrm{Aut}(A^*_0)$, $s = 1, 2, \dots, n$. According to Lemma 4.4, the permutation matrices P_s correspond to a multiplication by a nonzero element of $\mathbb{F}_q$ followed by a field automorphism. This means that τ correspond to a sequence of the transformations from Definition 4.2. As τ maps a row of D^*_C to another row of the same matrix, and these rows correspond to codewords of C, then the corresponding sequence maps a codeword to another codeword. Moreover, since the subset $\mathcal{B}$ of C used for the construction of D^*_C generates the code, τ defines an automorphism of C. □

Corollary 4.7. *The linear codes C and C' are equivalent if and only if the matrices D_C and $D_{C'}$ are isomorphic.*

Proof. We can consider the matrices D^*_C and $D^*_{C'}$. Obviously, if the codes are equivalent, their corresponding matrices are isomorphic.

If the matrices D^*_C and $D^*_{C'}$ are isomorphic, and $\phi : D^*_c \to D^*_{C'}$, because of the structure of the additional rows and columns, ϕ maps the cell A^*_{ij} from the first matrix to a cell from the same type, say A^*_{sl} of the second matrix. The construction of the matrices A^*_{ij} and A^*_{sl} is described in the beginning of this subsection and it shows that these matrices correspond to two nonzero elements of the field, say α^a and α^b. Moreover, the transformations which map A^*_{ij} to A^*_{sl} are permutations of the columns which correspond to multiplications with a nonzero element of the field and a field automorphism. Obviously, the cycle $\tau = (1, 2, \ldots, q-1)^{b-a}$ maps A^*_a to A^*_b, therefore all permutations which map A^*_a to A^*_b form the coset $\tau\mathrm{Aut}(A^*_a)$. Hence the restriction of ϕ on the set of columns B^*_j is in $\tau\mathrm{Aut}(A^*_a)$. Therefore ϕ corresponds to a composition of the transformations in Definition 4.2. It turns out that the codes are also equivalent. □

Linear codes can also be defined in terms of projective geometries [19], which gives another method of code representation. This approach is more convenient for codes with small dimension.

4.4. Linear code equivalence and multisets of points from projective geometry. Let $V(k, q)$ be a vector space of dimension k over $\mathbb{F}_q$. The projective space $PG(k-1, q)$ is the geometry whose points, lines, planes and hyperplanes are the subspaces of $V(k, q)$ of dimension 1, 2, 3, $k-1$, respectively. The dimension of a subspace of $PG(k-1, q)$ is one less than the dimension of a subspace of $V(k, q)$. In a projective space any two different points are incident with exactly one line and every line contains at least three points. The Veblen-Young axiom states that four different lines cannot intersect in exactly five different points.

A collineation of $PG(k-1, q)$ is a bijection from the set of the points of this projective space to itself, such that the images of collinear points are themselves collinear. Any projective linear transformation induces a collineation. Any collineation of $PG(k-1, q)$, $k-1 \geq 2$, can be represented by $x \mapsto Ax^\sigma$, where σ is an automorphism of the field $\mathbb{F}_q$, $x \in PG(k-1, q)$ is considered as a vector-column, and $A \in GL(k, q)$. The general linear group $GL(k, q)$ is the group of non-singular linear transformations of $V(k, q)$. It is isomorphic to the multiplicative group of $k \times k$ invertible matrices with entries from $\mathbb{F}_q$.

A collineation of a projective space to itself is also called an automorphism, and the set of all collineations of $PG(k-1, q)$ forms a group called the collineation group and denoted by $P\Gamma L(k-1, q)$.

Definition 4.5. *A multiset of points from $PG(k-1, q)$ is a family of points in $PG(k-1, q)$, in which a point can appear more than once. Two multisets of points from $PG(k-1, q)$, say $\mathcal{S}$ and $\mathcal{T}$, are said to be equivalent (or projectively equivalent) if there exists a collineation $\pi \in P\Gamma L(k-1, q)$ which maps $\mathcal{S}$ onto $\mathcal{T}$, i.e., $\pi(\mathcal{S}) = \{\pi(P) | P \in \mathcal{S}\} = \mathcal{T}$.*

Let C be a linear $[n, k]_q$ full length code with a generator matrix G. We can consider the columns $g_1, g_2, \ldots, g_n$ of G as representatives of points in the projective geometry $PG(k-1,q)$ (proportional columns represent the same point). Then $\mathcal{S} = \{g_1, g_2, \dots, g_n\}$ is a multiset of points in $PG(k-1,q)$. We say, that the multiset $\mathcal{S}$ is associated to the code C. If each hyperplane of $PG(k-1,q)$ meets $\mathcal{S}$ in at most $n-d$ points and there is a hyperplane meeting $\mathcal{S}$ in exactly $n-d$ points (multiplicities are counted), then S is associated to a linear code of minimum distance d.

It is known that two multisets S_1 and S_2, associated to linear codes C_1 and C_2 respectively, are projectively equivalent if and only if C_1 and C_2 are equivalent [11].

Each projective geometry $PG(k-1,q)$ is fully characterized by its points and hyperplanes. Using this, the geometry can be represented as a graph or as a binary matrix as well. Let $p_1, p_2, \dots, p_{\frac{q^k-1}{q-1}}$ be the points and $H_1, H_2, \dots, H_{\frac{q^k-1}{q-1}}$ be the hyperplanes in $PG(k-1,q)$. We obtain a $\dfrac{q^k-1}{q-1} \times \dfrac{q^k-1}{q-1}$ binary matrix A, which is the point-hyperplane incidence matrix.

$$A = \begin{array}{cc} \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,\frac{q^k-1}{q-1}} \\ a_{2,1} & a_{2,2} & \dots & a_{2,\frac{q^k-1}{q-1}} \\ \dots & & & \\ a_{\frac{q^k-1}{q-1},1} & a_{\frac{q^k-1}{q-1},2} & \dots & a_{\frac{q^k-1}{q-1},\frac{q^k-1}{q-1}} \end{pmatrix} & \begin{array}{c} H_1 \\ H_2 \\ \dots \\ H_{\frac{q^k-1}{q-1}} \end{array} \end{array}$$

where $a_{ij} = 1$, if $p_j \in H_i$ and $a_{ij} = 0$ otherwise. It is known that each row contains exactly q^{k-1} entries with value 1. To represent a linear code $\mathcal{C}$, obtained from $PG(k-1,q)$ via this binary matrix, we color its columns with the vector of colors $c = (c_1, c_2, \dots c_{\frac{q^k-1}{q-1}})$ where the color $c_j = r$, if the point p_j has multiplicity r. We denote the obtained colored matrix by A_C.

Theorem 4.8. *Two linear codes C_1 and C_2 are equivalent if and only if the associated multisets are equivalent if and only if the corresponding colored binary matrices A_1 and A_2 are isomorphic.*

Proof. Let S_1 and S_2 are the multisets associated to C_1 and C_2, respectively. We consider one more colored matrix G_i associated with S_i and C_i, $i = 1,2$. It is a $k \times \dfrac{q^k-1}{q-1}$ matrix whose columns are the points $p_1, p_2, \dots, p_{\frac{q^k-1}{q-1}}$ colored with the same vector of colors $c_i = (c_1^{(i)}, c_2^{(i)}, \dots c_{\frac{q^k-1}{q-1}}^{(i)})$ as A_{C_i}, $i = 1,2$. Since the matrices G_i and A_{C_i} are determined by the points and their multiplicities in S_i, one of these matrices is completely determined by the other one.

If S_1 and S_2 are equivalent, there is a bijection π in $PG(k-1,q)$ which maps the points from S_1 to the points in S_2 and the multiplicity of a point and its image are the same. This means that π acts as a permutation which maps the columns of G_1 to the columns of G_2 which preserves the coloring. Hence π acts in the same way as a permutation which maps the columns of A_{C_1} to the columns of A_{C_2} and preserves the coloring. Thus A_{C_1} and A_{C_2} are isomorphic as colored binary matrices.

Now consider the opposite case when the matrices A_{C_1} and A_{C_2} are isomorphic. Then there is a permutation of the columns which maps the columns of the first matrix to columns with the same color in the second matrix. The same permutation maps the columns of G_1 to columns with the same color in G_2. So this permutation acts on the projective space $PG(k-1,q)$ as a collineation π which preserves the multiplicities of the points as they are in both multisets. It turns out that $\pi(S_1) = S_2$. □

5. Hadamard matrices representation. There already exist many account of the Hadamard equivalence of Hadamard matrices [8, 9, 31] . The problem of deciding whether two Hadamard matrices are equivalent seems to be very difficult. There exists already an approach for representing Hadamard matrices as graphs [24], where the reduction of the equivalence problem of Hadamard matrices to a graph isomorphism problem is realized in polynomial time. We give a different approach for reducing the Hadamard equivalence problem to the problem of binary matrix isomorphism.

Definition 5.1. *An Hadamard matrix H of order n is an $n \times n$ matrix with entries ± 1 satisfying $HH^t = nI$. Two Hadamard matrices H_1 and H_2 are Hadamard equivalent if H_2 can be obtained from H_1 by a sequence of row permutations, column permutations, row negations and column negations. An automorphism of a Hadamard matrix is an equivalence with itself.*

Negation of some columns/rows can be described by an n-tuple of ± 1, in which -1 of a certain position means negation of the column/row with the same index. Denote the set of all n-tuples of ± 1 with Neg. We can present any transformation δ which maps one Hadamard matrix into another by a tuple $\delta = (\pi_c, \pi_r, \nu_c, \nu_r)$, where $\pi_c, \pi_r \in S_n$, $\nu_c, \nu_r \in Neg$. We denote by $Iso(H_1, H_2)$ the set of all transformations δ which map H_1 to H_2, and by $Aut(H_1)$ the automorphism group of an Hadamard matrix H_1.

An $n \times n$ Hadamard matrix can also be defined as an n-subset of the set of all possible ±1 n-tuples. It is important to notice that no two columns or rows of an Hadamard matrix can be proportional.

5.1. Representation as a graph. The first method of representing Hadamard matrices as graphs can be applied not only to Hadamard matrices, but to any $n \times m$ matrix $H = (h_{ij})$ with entries ± 1. For a given matrix H we define by $G = G(H)$ the graph with vertices $\{v_1, v_2, \dots, v_n,\ v'_1, v'_2, \dots, v'_n,$ $w_1, w_2, \dots, w_m,\ w'_1, w'_2, \dots, w'_m\}$ and edges

$$(v_i, w_j), (v'_i, w'_j) \text{ if } h_{ij} = 1, \quad \text{and} \quad (v_i, w'_j), (v'_i, w_j) \text{ if } h_{ij} = -1.$$

In addition we color the vertices $v_1, v_2, \dots, v_n,\ v'_1, v'_2, \dots, v'_n$ with col_c (columns color) and $w_1, w_2, \dots, w_n,\ w'_1, w'_2, \dots, w'_n$ with col_r(rows color). Two matrices H_1 and H_2 are Hadamard equivalent if and only if the corresponding colored graphs $G(H_1)$ and $G(H_2)$ are isomorphic. If an $n \times n$ Hadamard matrix H is given, then the associated graph $G = G(H) = (V_H, E_H)$ has $|V_H| = 4n$ vertices: Thus $(4n)^2 = 16n^2$ memory units are needed to store the matrix as a graph. On the other hand an Hadamard matrix needs only n^2 memory cells if it is put directly in the computer.

5.2. Representation as a binary matrix. Hadamard equivalence has two more operations than the binary matrix isomorphism, namely negation of a row and negation of a column. That's why we correlate a binary matrix H_b to a given Hadamard matrix H in following way. We map any 1 and -1 from the matrix H as follows:

$$1 \to \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad -1 \to \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

In this way we construct the $2n \times 2n$ binary matrix H_b.

Example 5.

$$\text{If } H = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}, \quad \text{then } H_b = \left(\begin{array}{cc|cc|cc|cc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ \hline 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right).$$

The next theorem holds for this representation.

Theorem 5.1. *Two Hadamard matrices H_1 and H_2 are equivalent if and only if the binary matrices H_{1b} and H_{2b} are isomorphic.*

Proof. Any column c_j from H uniquely defines a pair of columns $(c_j^{(1)}, c_j^{(2)})$ in H_b such that the sum of $(c_j^{(1)}$ and $c_j^{(2)})$ is the all ones vector. We can consider the set of columns in the $2n \times 2n$ matrix H_b as an ordered partition $C_{H_b} = ((c_1^{(1)}, c_1^{(2)}), \ldots, (c_n^{(1)}, c_n^{(2)}))$ which contains n ordered pairs of columns. In the same way we can define an ordered partition of the rows $R_{H_b} = ((r_1^{(1)}, r_1^{(2)}), \ldots, (r_n^{(1)}, r_n^{(2)}))$.

$\Rightarrow$) If $H_1 \cong H_2$, then there exists $\delta = (\pi_c, \pi_r, \nu_c, \nu_r) \in Iso(H_1, H_2)$. The permutation $\pi_c \in S_n$ acts as a permutation of the pairs in $C_{H_{1b}}$. Negation of a column corresponds to a transposition in the corresponding pair of $C_{H_{1b}}$. That is why the sequence of transformations π_c, ν_c correspond to a permutation $\tau_c \in S_{2n}$ of the columns of H_{1b} and π_r, ν_r correspond to a permutation τ_r of the rows of H_{1b}. If we apply the transformation δ to H_1 we obtain the second Hadamard matrix H_2. Hence applying the permutation τ_c to the columns and τ_r to the rows of H_{1b} we will have the matrix H_{2b} and therefore these two binary matrices are isomorphic.

$\Leftarrow$) Conversely, let $H_{1b} \cong H_{2b}$. Let τ_c and τ_r be the permutations of the columns and the rows of H_{1b} which map this matrix to H_{2b}. Since the columns in any pair $(c_j^{(1)}, c_j^{(2)}) \in C_{H_{1b}}$ are complements to each other (their sum is the all ones vector) their images form a pair in H_{2b}. Hence τ_c maps $C_{H_{1b}} = ((c_1^{(1)}, c_1^{(2)}), \ldots, (c_n^{(1)}, c_n^{(2)}))$ to $C_{H_{2b}}$. It turns out that we can represent τ_c as a sequence of transformations $\pi_c \in S_n, \nu_c \in Neg$. The same is true for the rows of both binary matrices. Thus (τ_c, τ_r) correspond to a tuple $\delta = (\pi_c, \pi_r, \nu_c, \nu_r)$. Since (τ_c, τ_r) maps H_{1b} to H_{2b}, then the transformation δ sends H_1 to H_2. □

The necessary amount of computer memory is $4n^2$ cells, which is four times less than the amount used in the case of graph representation.

6. Conclusion.

The methods for reducing the problems for isomorphism of combinatorial objects to isomorphism of binary matrices are very useful in our research. Using these methods we study, construct and classify self-dual codes [1], optimal codes [6], Hadamard matrices [5], etc.

Acknowledgments. The authors would like to thank Prof. Stefka Bouyuklieva for her help, useful discussions and valuable advice.

REFERENCES

[1] Bouyuklieva S., I. Bouyukliev. An algorithm for classification of binary self-dual codes. *IEEE Transactions on Information Theory*, **58** (2012), 3933–3940.

[2] Bouyuklieva S., I. Bouyukliev, M. Harada. Some extremal self-dual codes and unimodular lattices in dimension 40. *Finite Fields and Their Applications*, **21** (2013), 67–83.

[3] Bouyukliev I. About the code equivalence. In: Advances in Coding Theory and Cryptology (Eds T. Shaska, W.C. Huffman, D. Joyner, V. Ustimenko), Series on Coding Theory and Cryptology, World Scientific Publishing, Hackensack, NJ, 2007, 126–151.

[4] Bouyukliev I. What is Q-Extension? *Serdica Journal of Computing*, **1** (2007), No 2, 115–130. `http://www.moi.math.bas.bg/~iliya/Q\_ext.htm`

[5] Bouyukliev I., V. Fack, J. Winne. 2-(31,15,7), 2-(35,17,8) and 2-(36,15,6) designs with automorphisms of odd prime order, and their related Hadamard matrices and codes. *Designes, Codes and Cryptography*, **51** (2009), No 2, 105-122.

[6] Bouyukliev I., J. Simonis. Some new results on optimal codes over F_5. *Designs, Codes and Cryptography*, **30** (2003), 97–111.

[7] Brinkmann G. Generating regular directed graphs. *Discrete Mathematics*, **313** (2013), 1–7.

[8] Colbourn C. J., M. J. Colbourn. Deciding Hadamard equivalence of Hadamard matrices. *BIT Numerical Mathematics*, **21** (1981), No 3, 374–376.

[9] Cooper J., J. Milas, W. D. Wallis. Hadamard equivalence. In: Combinatorial Mathematics (Eds D. A. Holton, J. Seberry), Springer-Verlag, 1978, 126–135.

[10] Darga P. T., M. H. Liffiton, K. A. Sakallah, I. L. Markov. Exploiting struc- ture in symmetry detection for CNF. In: Proceedings of the 41st Design Automation Conference, San Diego, CA, 2004, 530–534.

[11] Dodunekov S., J. Simonis. Codes and Projective Multisets. *Electronic journal of combinatorics*, **5** (1998), R37.

[12] Foggia P., C. Sansone, M. Vento. A Performance Comparison of Five Algorithms for Graph Isomorphism. In: Proceedings of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition, Ischia, 2001, 188–199.

[13] Fuelner T. The automorphism grops of linear codes and canonical representatives of their semilinear isometry classes. *AMC*, **3** (2009), No 4, 363–383.

[14] Junttila T., P. Kaski. Engineering an efficient canonical labeling tool for large and sparse graphs. In: Proceedings of the Ninth Workshop on Algorithm Engineering and Experiments (ALENEX07), SIAM, New Orleans, LA, USA, 2007, 135–149.

[15] Junttila T., P. Kaski. Conflict Propagation and Component Recursion for Canonical Labeling. In: Proceedings of the 1st International ICST Conference on Theory and Practice of Algorithms (TAPAS 2011), Springer, 2011, 151–162.

[16] Kaski P., P. R. J. Östergård. Classification algorithms for codes and designs. Springer-Verlag, Berlin Heidelberg, 2006.

[17] Kocay W. On writing isomorphism programs. In: Computational and Constructive Design Theory(Ed. W. D. Wallis), Kluwer, 1996, 135–175.

[18] Kreher D. L., D. R. Stinson. Combinatorial Algorithms: Generation, Enumeration and Search. CRC Press, 1999.

[19] Landgev I. Linear code over finite fields and finite projective geometries. *Discrete Mathematics*, **213** (2000), 211–244.

[20] Leon J. Computing automorphism groups of error-correcting codes. *IEEE Trans. Inform. Theory*, **28** (1982), 496–511.

[21] Leon J. S. An algorithm for computing the automorphism group of a Hadamard matrix. *Journal of Combinatorial Theory*, **A27** (1979), 289–306.

[22] MacWilliams J., N. J. A. Sloane. The Theory of Error-Correcting Codes, North-Holland, Amsterdam, 1977.

[23] Mateva Z. Constructing a canonical form of a matrix in several problems about combinatorial designs. *Serdica Journal of Computing*, **2** (2008), No 4, 349–368.

[24] McKay B. Hadamard Equivalence Via Graph Isomorphism. *Discrete Mathematics*, **27** (1979), 213–214.

[25] McKay B. Practical graph isomorphism. *Congressus Numerantium*, **30** (1981), 45–87.

[26] McKay B., A. Piperno. Practical Graph Isomorphism. II, arXiv:1301.1493 [cs.DM]

[27] McKay B. NAUTY User's Guide (Version 2.4). Australian National University, Canberra ACT 0200, Australia 2009.

[28] Sendrier N. The Support Splitting Algorithm. *IEEE Trans, Info. Theory*, **46** (2000), 1193–1203.

[29] Sendrier N., D. Simos. How easy is code equivalence over $\mathbb{F}_q$? In: Proceedings of the 8th International Workshop on Coding Theory and Cryptography (WCC 2013), Bergen, Norway, 2013, 1–12.

[30] Petrank E. , Ron M. Roth. Is code equivalence easy to decide?, *IEEE Trans. Inform. Theory*, **43** (1997), 1602–1604.

[31] Wallis W. D., J. Wallis. Equivalence of Hadamard matrices. *Israel J. Math.,* **7** (1969), 122–128.

Iliya Bouyukliev
Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
P.O. Box 323
5000 Veliko Tarnovo, Bulgaria
e-mail: iliyab@math.bas.bg

Mariya Dzhumalieva-Stoeva
Faculty of Mathematics and Informatics
Veliko Tarnovo University
2, Theodosi Tarnovski Str.
5000 Veliko Tarnovo, Bulgaria
e-mail: mdzhumalieva@gmail.com

Received June 24, 2014
Final Accepted January 4, 2015