Serdica J. Computing **10** (2016), No 1, 79–92

**Serdica**
Journal of Computing

Bulgarian Academy of Sciences
Institute of Mathematics and Informatics

# CONJUGATES FOR FINDING THE AUTOMORPHISM GROUP AND ISOMORPHISM OF DESIGN RESOLUTIONS*

Svetlana Topalova

Abstract. Consider a combinatorial design $D$ with a full automorphism group $G_D$. The automorphism group $G$ of a design resolution $R$ is a subgroup of $G_D$. This subgroup maps each parallel class of $R$ into a parallel class of $R$. Two resolutions $R_1$ and $R_2$ of $D$ are isomorphic if some automorphism from $G_D$ maps each parallel class of $R_1$ to a parallel class of $R_2$. If $G_D$ is very big, the computation of the automorphism group of a resolution and the check for isomorphism of two resolutions might be difficult. Such problems often arise when resolutions of geometric designs (the designs of the points and $t$-dimensional subspaces of projective or affine spaces) are considered.

For resolutions with given automorphisms these problems can be solved by using some of the conjugates of the predefined automorphisms. The method is explained in the present paper and an algorithm for construction of the necessary conjugates is presented.

*ACM Computing Classification System* (1998): F.2.1, G.1.10, G.2.1.

*Key words:* conjugates, design resolutions, parallelisms, automorphism group.

*This work was partially supported by the Bulgarian National Science Fund under Contract No I01/0003.

## 1. Introduction.

***1.1. Definitions and notations.*** Let $V = \{P_i\}_{i=1}^{v}$ be a finite set of *points*, and $\mathcal{B} = \{B_j\}_{j=1}^{b}$ a finite collection of $k$-element subsets of $V$, called *blocks*. $D = (V, \mathcal{B})$ is a *2-design* with parameters 2-($v$,$k$,$\lambda$) if any 2-subset of $V$ is contained in exactly $\lambda$ blocks of $\mathcal{B}$. A *parallel class* is a partition of the point set of the design by blocks. A *resolution* of the design is a partition of the collection of blocks by parallel classes.

A *t-spread* in $PG(n, q)$ is a set of distinct $t$-dimensional subspaces which partition the point set. A *t-parallelism* is a partition of the set of $t$-dimensional subspaces by $t$-spreads. Usually 1-spreads (1-parallelisms) are called line spreads (line parallelisms) or just *spreads* (*parallelisms*). There can be line spreads and parallelisms if $n$ is odd.

The incidence of the points and $t$-dimensional subspaces of $PG(n, q)$ defines a 2-design. There is a one-to-one correspondence between the $t$-parallelisms of $PG(n, q)$ and the resolutions of the design of its points and $t$-dimensional subspaces.

***1.2. Computer-aided research on parallelisms.*** Research on $t$-spreads and $t$-parallelisms is motivated by their various relations to translation planes [4], [8] and by some of their practical applications in coding theory [5] and cryptography [16].

Since not many different types of parallelisms are known, computer-aided constructions are of particular interest. A great deal of the computer-aided investigations imply classifications of $t$-parallelisms with predefined automorphism groups [12], [13], [14], [15], [17], [18], [20], [19], [21], [22], [23]. In all these works the authors use the normalizer of the constructive automorphism group in order to filter away isomorphic parallelisms. A summary of this method follows.

***1.3. Isomorphism of resolutions which possess one and the same predefined automorphism group.*** Let $D$ be a combinatorial design with a full automorphism group $G_D$. Consider resolutions of $D$ with a predefined automorphism group $G_c$. We have to check if there is some permutation $\varphi \in G_D$ such that it maps a resolution with $G_c$ to another resolution with $G_c$. Let $R$ and $R'$ be two resolutions with $G_c$, such that

$$R' = \varphi R.$$

Let $\alpha \in G_c$. Then $\varphi R = \alpha\varphi R$ and thus

$$R = \varphi^{-1}\alpha\varphi R,$$

namely $\varphi^{-1}\alpha\varphi$ is also an automorphism of $R$. That is why $R$ is invariant both under $G_c$ and under $\varphi^{-1}G_c\varphi$. If

$$\varphi^{-1}G_c\varphi = G_c,$$

then $\varphi$ is in the normalizer $N(G_c)$ of $G_c$ in $G_D$, which is defined as

$$N(G_c) = \{g \in G_D \mid gG_cg^{-1} = G_c\}.$$

If $\varphi$ is not in the normalizer, then $\varphi^{-1}G_c\varphi$ is a conjugate subgroup and since $\varphi$ is not an automorphism of $R$, there must exist an automorphism $\psi$ of $R$ such that

$$\varphi^{-1}G_c\varphi = \psi G_c\psi^{-1}.$$

Then $G_c = \varphi\psi G_c\psi^{-1}\varphi^{-1}$ and therefore $\varphi\psi \in N(G_c)$. Since $\varphi\psi R = \varphi R$, to establish isomorphism of two resolutions $R$ and $R'$ with $G_c$ it is enough to check if there is a permutation of $N(G_c)$ which maps $R$ to $R'$.

***1.4. Subject of this paper and applicability of the presented methods.*** There exist general methods for finding the automorphism group of combinatorial structures, the most popular being those of Leon [9] and McKay [10]. There are, however, cases when the method by which the structures are constructed allows determining the automorphism group by some algorithm which is not general, but only applicable with this particular construction. The present paper offers such a construction-related method. Namely, it considers the computation of the orders and generating sets of the automorphism groups of resolutions which were constructed with some predefined automorphisms and by using the normalizer to filter away isomorphic solutions (see 1.3).

If some element of $N(G_c)$ maps the resolution $R$ to itself, it is its automorphism. So by the isomorphism test described in 1.3. we also obtain some of the automorphisms of $R$. It might have, however, more automorphisms, which are not from $N(G_c)$. The full automorphism group $G$ is not determined in [12], [13], [14], [15], and [17]. Here in section 2 we present the algorithm used to compute the order of the full automorphism groups of parallelisms of $PG(3,4)$ ($|G_D|$=$2^{13} \cdot 3^4 \cdot 5^2 \cdot 7 \cdot 17$) with automorphisms of orders 7 and 5 [20], [21] and of parallelisms of $PG(3,5)$ ($|G_D|$=$2^9 \cdot 3^2 \cdot 5^6 \cdot 13 \cdot 31$) with automorphisms of orders 13 and 3 [19], [22].

The method we describe is based on the usage of conjugates. Many papers deal with the problem of determining the conjugacy classes of a big group and of finding one representative of each conjugacy class [1], [2], [3], [6], [7], [11], [24]. But for the problem considered here, we do not need representatives of all conjugacy

classes. We are interested in the generation of all representatives of one and the same conjugacy class. Such a generation is a function implied in the computer algebra software system GAP. The storage of all conjugates, however, needs a lot of memory. This makes its usage impossible if $|G_D|$ is rather big.

A further study of the problem we consider shows that we do not actually need all conjugates, but only a much smaller part of them (necessary set). Section 2 defines the necessary set of conjugates for computing the automorphism group of a resolution $R$ invariant under $G_c$ and shows how this set can be generated by a simple algorithm.

The isomorphism test described in 1.3. is only applicable if $R$ and $R'$ are invariant under one and the same subgroup of $G_D$ and cannot be used if $R$ and $R'$ are invariant under conjugate subgroups of $G_D$. Such a case arises, for instance, when we consider dual parallelisms of parallelisms with a constructive automorphism.

Section 3 explains how the necessary set of conjugates constructed as shown in section 2 can be used to determine if a resolution $R$ with given automorphisms is isomorphic to another resolution $R'$. The method was used to determine the pairs of duals of regular parallelisms of $PG(3,5)$ with automorphisms of order 3 [22].

The paper ends with several comments given in Section 4.

## 2. The full automorphism group of a resolution with given automorphisms.

***2.1. Conjugate approach.*** Consider a resolution $R$ which is invariant under $G_c$. Let $R$ have an automorphism $\psi \notin N(G_c)$. Then it is invariant under $\psi G_c \psi^{-1}$. So we can first find all conjugate subgroups under which the resolution is invariant. The automorphism group which they generate, however, may still not be the full automorphism group of $R$.

Let $G_{N+C}$ be the group of automorphisms of $R$ containing all elements of $N(G_c)$ which are automorphisms of $R$ and all conjugates of $G_c$ under which $R$ is invariant. Let $\alpha$ be an automorphism of $R$ such that

$$\alpha \notin G_{N+C}.$$

Then $R$ is invariant under $\alpha^{-1} G_c \alpha$. It follows from the definition of $G_{N+C}$ that

$$\alpha^{-1} G_c \alpha \in G_{N+C}.$$

This means that there exists some $\beta \in G_{N+C}$ such that

$$\alpha^{-1} G_c \alpha = \beta^{-1} G_c \beta$$

and $G_c = \alpha\beta^{-1}G_c\beta\alpha^{-1}$, namely $\alpha\beta^{-1} \in N(G_c)$. Let $\alpha\beta^{-1} = \gamma \in N(G_c)$. Then

$$\alpha = \gamma\beta,$$

where $\gamma \in N(G_c)$ and $\beta \in G_{N+C}$. That is why we can determine the automorphism group of a resolution by the following steps:

1. Find $G_N$ – the subgroup of $N(G_c)$ under which $R$ is invariant. This can be done by checking for each automorphism of $N(G_c)$ if it preserves the resolution or not.

2. Check if $R$ is invariant under some conjugate subgroup of $G_c$.

3. If no conjugate subgroup of $G_c$ preserves $R$, then the full automorphism group of $R$ is $G_N$ and in this case the next three steps are omitted.

4. Find the group $G_{N+C}$ generated by the generators of $G_N$ and the conjugate subgroups of $G_c$ which preserve $R$

5. For each $\beta \in G_{N+C}$ and each $\gamma \in N(G_c)$ check if $\beta\gamma$ is an automorphism of $R$

6. Find the full automorphism group $G$ from $G_{N+C}$ and the automorphisms found in step 5.

If $G_c$ is a cyclic group of prime order, then each conjugate of subgroup $G_c$ is generated by one of the conjugates of a given element $c \in G_c$. Thus the upper steps 2, 3 and 4 can be replaced by the following ones:

2. Take an element $c \in G_c$ and check if $R$ is invariant under some of its conjugates.

3. If no conjugate of $c$ preserves $R$, then the full automorphism group of $R$ is $G_N$ and in this case the next three steps are omitted.

4. Find the group $G_{N+C}$ generated by the generators of $G_N$ and the conjugates of $c$ which preserve $R$

If $G_c$ is not a cyclic group of prime order, we can take any of its subgroups of prime order. Applying the above steps to it yields the full automorphism group of the resolution. To do this we need all conjugates of a prime order permutation $c \in G_D$.

***2.2. Construction of conjugates.*** Let

$$S_g = \{g_1, g_2, \dots g_m\}$$

be a generating set of $G_D$. Denote by $C_c$ the centralizer of $c$ in $G_D$, and by $n$ the number of conjugates of $c \in G_D$, where

$$n = |G_D|/|C_c|.$$

We construct the conjugates in several rounds. Before round 1 there is one element in the set of the constructed conjugates, the element $c$. The conjugates constructed at round $r-1$ are *extended* at round $r$, i.e. their conjugates are constructed using $S_g$. The algorithm terminates after round $r$ if no new conjugate is found at this round.

**Algorithm 1. Construction of all conjugates**

This algorithm constructs the set $S_{all}$ of all $n$ conjugates. At round $r$ it performs the following:

for each $\delta$ which was added to $S_{all}$ at round $r-1$
{
  for $j = 1, 2, \dots m$
  {
    construct $\delta_j = g_j^{-1} \delta g_j$
    if $\delta_j \notin S_{all}$ then add $\delta_j$ to $S_{all}$.
  }
}

**Lemma 1.** *Algorithm* 1 *generates all conjugates of* $c \in G_D$*, where* $c$ *is of prime order.*

Proof. Let $\alpha^{-1} c \alpha$ be a conjugate of $c$ and let $\alpha = g_{i_1} g_{i_2} \dots g_{i_s}$. Then $g_{i_1}^{-1} c g_{i_1}$ will be added to $S_{all}$ at round 1, $g_{i_2}^{-1} g_{i_1}^{-1} c g_{i_1} g_{i_2}$ at round 2 and $g_{i_s}^{-1} \dots g_{i_2}^{-1} g_{i_1}^{-1} c g_{i_1} g_{i_2} \dots g_{i_s}$ at round $s$. □

***2.3. Conjugates that may not be constructed.*** Our aim is to find the full automorphism group $G$ of the resolution $R$ with an automorphism $c$ of prime order $p$ by the method presented above. We actually need $G_{N+C}$. For that purpose we have to construct the group generated by all conjugates of $c$ which are automorphisms of $R$.

Let $\delta$ be a conjugate of $c$. Since $\delta^k$ preserves $R$ if and only if $\delta$ preserves it, the set of the conjugates we need may contain only one of the powers of each conjugate. We next notice that $c^{-i}\delta c^i$ is an automorphism of $R$ if and only if $\delta$ is. Based on these two observations, we can construct only a set $S$ of conjugates which we call a *necessary set*.

Denote by *necessary set of conjugates* of $c$ a set of conjugates such that:

- if $\delta$ is a conjugate of $c$ and $\delta \notin S$, then $\exists\ k, i$ such that $c^{-i}\delta^k c^i \in S$.
- if $\delta \in S$, then $c^{-i}\delta^k c^i \notin S$ for $k = 2, \ldots p-1, i = 1, \ldots p-1$.

**Lemma 2.** *Consider a resolution $R$ with an automorphism $c$ of prime order $p$. The group generated by all conjugates of $c$ which are automorphisms of $R$ can be found using a necessary set $S$ of conjugates of $c$.*

P r o o f. If $\delta$ preserves $R$ and $\delta \notin S$, then by the definition of the necessary set $\exists\ k, i$ such that $c^{-i}\delta^k c^i \in S$. If $\delta$ preserves $R$, then $c^{-i}\delta^k c^i$ preserves it too. Since $c$ is an automorphism of $R$, $\delta^k$ is an automorphism too, and $\delta \in \left\langle c^{-i}\delta^k c^i \right\rangle$. $\square$

The check if a conjugate preserves $R$ or not is quite complex. Therefore it is very helpful to use the smaller necessary set instead of the set of all conjugates. If we have constructed the set $S_{all}$ of all conjugates by Algorithm 1, then a necessary set of conjugates $S$ can be obtained by a simple sequential check, i.e.

$S = \emptyset$
for each $\delta \in S_{all}$
{
    if $New(\delta)$ add $\delta$ to $S$.
}

The function $New$ returns true if $c^{-i}\delta^k c^i \notin S$ for each $i = 0, 1, \ldots, p-1$ and each $k = 1, \ldots, p-1$.

Algorithm 1, however, needs a lot of memory. In cases when there is not enough memory for Algorithm 1, Algorithm 2 presented below can be used.

**Algorithm 2. Construction of a necessary set of conjugates**

This is a modification of Algorithm 1. It constructs a necessary set of conjugates $S \subseteq S_{all}$. At round $r$ it performs the following:

for each $\delta$ which was added to $S$ at round $r-1$
{
  for $j = 1, 2, \ldots, m$
  {
    for $i = 0, 1, \ldots, p-1$
    {
      construct $\delta_{i,j} = g_j^{-1} c^{-i} \delta c^i g_j$ (1)
      if New$(\delta_{i,j})$ then add $\delta_{i,j}$ to $S$.
    }
  }
}

The function *New* returns true if $c^{-i}\delta_{i,j}^k c^i \notin S$ for each $i = 0, 1, \ldots, p-1$ and each $k = 1, \ldots, p-1$.

**Lemma 3.** *Algorithm* 2 *generates a necessary set* $S$ *of conjugates of* $c \in G_D$*, where* $c$ *is of prime order* $p$*.*

Proof. Algorithm 2 does not save all conjugates (it only saves a necessary set), but actually constructs all of them. To ensure that all conjugates will be constructed at round $r$ we need to extend all (not only the saved) conjugates constructed at round $r-1$.

Let $\delta = \alpha^{-1} c \alpha$ be a conjugate of $c$. Then

$$\delta^2 = \alpha^{-1} c \alpha \alpha^{-1} c \alpha = \alpha^{-1} c^2 \alpha.$$

Therefore

$$\delta^k = \alpha^{-1} c^k \alpha.$$

This means that $\delta$ and $\delta^k$ will be extended in the same way at each round. Thus the presence of only one power of each conjugate in the necessary set does not lead to a loss of conjugates.

Let the conjugate $\delta = \alpha^{-1} c \alpha$ be saved at round $r-1$. Then it is possible that the conjugate

$$\rho = c^{-i} \alpha^{-1} c^k \alpha c^i$$

was constructed but not saved. Row (1) in the algorithm ensures that

$$c^{-i} \alpha^{-1} c \alpha c^i = \rho^{1-k}$$

is extended at round $r$. Since $\rho$ and $\rho^{1-k}$ are further extended in the same way, there is no loss of conjugates. □

## 3. Isomorphism of resolutions invariant under conjugate subgroups.

***3.1. Conjugate approach.*** Let $R$ be a resolution invariant under $G_c$ and let $R'$ be a resolution, such that

$$R = \varphi R'.$$

Our aim is to find $\varphi$. If $G_D$ is rather big, looking for a $\varphi \in G_D$ is too slow. Let $G_c$ be a cyclic group of order $p$ and let $c \in G_c$. Then $\varphi R' = c\varphi R'$ and thus

$$R' = \varphi^{-1} c \varphi R',$$

namely $R'$ is invariant under $\varphi^{-1} c \varphi$.

Suppose we can find all conjugates of $c$ under which $R'$ is invariant, and in particular, we can find $\varphi^{-1} c \varphi$. Suppose we can also find some $\psi \in G_D$ such that

$$\varphi^{-1} c \varphi = \psi^{-1} c \psi.$$

Then $c = \varphi \psi^{-1} c \psi \varphi^{-1}$ and therefore

$$\varphi \psi^{-1} \in N(G_c).$$

Let $\varphi \psi^{-1} = \gamma \in N(G_c)$. Then

$$\varphi = \gamma \psi.$$

That is why we can determine if $R$ and $R'$ are isomorphic by the following steps:

1. Find all conjugates of $c$ under which $R'$ is invariant.
2. If no conjugate of $c$ preserves $R'$, it is not isomorphic to $R$ and the algorithm stops.
3. For each conjugate $\delta$ of $c$ which preserves $R'$
   - find $\psi$ such that $\delta = \psi^{-1} c \psi$.
   - for each $\gamma \in N(G_c)$ check if $\gamma\psi$ maps $R'$ to $R$. If such a map is found, $R$ and $R'$ are isomorphic and the algorithm stops.
4. If no mapping of $R'$ to $R$ is found, the two resolutions are non isomorphic and the algorithm stops.

If $G_c$ is not a cyclic group of prime order, we can take any of its subgroups of prime order and apply the above steps. The next question is whether it is possible to use the necessary set $S$ of conjugates at step 1 in this case.

***3.2. Using the necessary set of conjugates.*** If $\delta$ preserves $R'$, $\delta^k$ also does and vice versa. That is why no problem arises from the fact that if $\delta$ is in $S$, then $\delta^k$ is not. It is possible, however, that $c^{-i}\delta c^i \notin S$ is an automorphism of $R'$ while $\delta \in S$ is not. If we have constructed the necessary set $S$ of conjugates of $c$, we can implement step 1 in the following way:

$S_a = \emptyset$
for each $\delta \in S$
{
    for $i = 0, 1, \ldots, p-1$
    {
        construct $\delta_i = c^{-i}\delta c^i$
        if $\mathrm{Aut}(\delta_i)$ then add $\delta_i$ to $S_a$.
    }
}

where $S_a$ is a subset of the set $S_{aut}$ of all conjugates under which $R'$ is invariant, such that

- if $\delta \in S_{aut}$ and $\delta \notin S_a$, then $\exists\ i$ such that $\delta^i \in S_a$.
- if $\delta \in S_a$, then $\delta^i \notin S_a$ for all $i = 2, \ldots, p-1$.

The function *Aut* returns true if $\delta_i$ is an automorphism of $R'$.

The above described isomorphism test implies that for each $\delta \in S_a$ we can find $\psi$ such that

$$\delta = \psi^{-1} c \psi.$$

This problem can be solved easily if during the generation of the necessary set of conjugates some additional information is saved when each new conjugate is added. Namely, if the conjugates are saved in an array, then when we add the $k$-th conjugate $\delta_{i,j} = g_j^{-1} c^{-i} \delta c^i g_j$ we can also save the number of the father-conjugate $\delta$ and the numbers $i$ and $j$. Then we can find $\psi$ by a simple recursive function:

```
FindMult(int k)
{
    if(j[k]=0) assign ψ = e
    else
    {
        FindMult(father[k])
        construct ψ = ψc^{i[k]} g_{j[k]}
    }
}
```

where $e$ is the identity permutation.

**4. Concluding remarks.** This paper presents just one possible way to compute the full automorphism group of a resolution $R$ with given automorphisms and to establish if it is isomorphic to another resolution of the same design. The method is suitable for resolutions of designs with big automorphism groups. It was developed and proved to be useful in connection with problems concerning the classification of parallelisms of $PG(3,4)$ and $PG(3,5)$ with predefined automorphisms [20], [19], [21], [22]. The advantages of the present method in these cases are based on the fact that the considered resolutions possess predefined automorphisms and this algorithm uses the predefined groups, while general purpose algorithms do not.

The method implies a construction of the necessary set of conjugates of a given automorphism of prime order. Let us count the number $n_n$ of conjugates in a necessary set $S$:

$$n_n = n_c + n_r$$

where $n_c$ is the number of conjugates in $S$ such that $G_c$ is in their centralizer and $n_r$ of the rest. Let us construct from each conjugate $\delta \in S$ the conjugates

$$c^{-i}\delta c^i$$

for $i = 0, 1, \ldots, p-1$. This way we will obtain a set $S_a$ of

$$n_c + pn_r$$

conjugates. Consider a conjugate

$$\delta = \psi^{-1} c \psi$$

from $S_a$. We can obtain from it the conjugates

$$\psi^{-1} c^k \psi$$

for $k = k_1, k_2, \ldots, k_h$, where $h$ is the number of the positive powers of $c$ which are in one conjugacy class with $c$. This way we obtain all $n$ conjugates of $c$ and

$$n = h(n_c + pn_r).$$

Since $n_c$ is relatively very small compared to $n_r$, the number of conjugates in a necessary set is almost $hp$ times smaller than the number of all conjugates in the class.

In conclusion it can be noted that Algorithm 2 is easy to implement and can be applied in cases different from the resolutions automorphism problem for which it was initially developed. It can be used in cases when all conjugates are needed and other algorithms are not applicable because of lack of memory. Each conjugate in a necessary set yields directly $ph$ or $h$ other conjugates and in this way all conjugates can be easily used.

## REFERENCES

[1] Butler G. An inductive schema for computing conjugacy classes in permutation groups. *Math. Comp.*, **62** (1994), 363–383.

[2] Cannon J., D. Holt. Computing conjugacy class representatives in permutation groups. *Journal of Algebra*, **300** (2006), No 1, 213–222.

[3] Cannon J., B. Souvignier. On the computation of conjugacy classes in permutation groups. In: W. Kuchlin (ed.). Proceedings of International Symposium on Symbolic and Algebraic Computation Maui, 1997. Assoc. Comput. Mach. (1997), 392–399.

[4] Eisfeld J., L. Storme. (Partial) t-spreads and minimal t-covers in finite projective spaces. In: Lecture notes from the Socrates Intensive Course on Finite Geometry and its Applications, Ghent, April 2000.

[5] Etzion T., N. Silberstein. Codes and designs related to lifted MRD codes. *IEEE T Inform Theory*, **59** (2013), No 2, 1004–1017.

[6] Felsch V., J. Neubüser. An algorithm for the computation of conjugacy classes and centralizers in p-groups. In: E. W. Ng (ed.). Symbolic and Algebraic Computation. *Lecture Notes in Computer Science*, **72** (1979), 452–465.

[7] Hulpke A. Conjugacy classes in finite permutation groups via homomorphic images. *Math. Comp.*, **69** (2004), No 232, 1633–1651.

[8] Johnson N. Combinatorics of Spreads and Parallelisms, CRC Press, 2010.

[9] Leon J. S. Computing automorphism groups of combinatorial objects. In: M. D. Atkinson (ed.). Computational Group Theory. Academic Press, 1984, 321–335.

[10] McKay B. D. Computing automorphisms and canonical labellings of graphs. In: D. A. Holton, J. Seberry (eds). Combinatorial Mathematics. *Lecture Notes in Mathematics*, **686** (1978), 223–232.

[11] Mecky M., J. Neubüser. Some remarks on the computation of conjugacy classes of soluble groups. *Bull. Austral. Math. Soc.*, **40** (1989), 281–292.

[12] Prince A. Parallelisms of PG(3,3) invariant under a collineation of order 5. In: Norman L. Johnson (ed.). Mostly Finite Geometries. *Lect Notes Pure Appl*, **190** (1997), 383–390.

[13] Prince A. The cyclic parallelisms of PG(3,5). *Eur J Combin*, **19** (1998), No 5, 613–616.

[14] Sarmiento J. Resolutions of PG(5,2) with point-cyclic automorphism group. *Journal of Combinatorial Designs*, **8** (2000), No 1, 2–14.

[15] Sarmiento J. On point-cyclic resolutions of the 2-(63,7,15) design associated with PG(5,2). *Graphs and Combinatorics*, **18** (2002), No 3, 621–632.

[16] Stinson D. Combinatorial Designs: Constructions and Analysis. Springer-Verlag, New York, 2004.

[17] Stinson D. and S. Vanstone. Orthogonal packings in PG(5,2). *Aequationes Mathematicae*, **31** (1986), No 1, 159–168.

[18] Topalova S., S. Zhelezova. 2-Spreads and Transitive and Orthogonal 2-Parallelisms of PG(5,2). *Graphs and Combinatorics*, **26** (2010), No 5, 727–735.

[19] Topalova S., S. Zhelezova. Parallelisms of PG(3,5) with automorphisms of order 13. In: Proc. VII Intern.Workshop Optimal Codes and Related Topics, Albena, Bulgaria, 2013, 176–180.

[20] Topalova S., S. Zhelezova. On transitive parallelisms of PG(3,4). *Applicable Algebra in Engineering, Communications and Computing*, **24** (2013), No 3–4, 159–164.

[21] Topalova S., S. Zhelezova. On point-transitive and transitive deficiency one parallelisms of PG(3,4). *Designs Codes and Cryptography*, **75** (2015), No 1, 9–19.

[22] Topalova S., S. Zhelezova. New regular parallelisms of PG(3,5). *Journal of Combinatorial Designs*, **24** (2016), No 10, 473–482.

[23] Zhelezova S. Cyclic parallelisms of PG(5,2). *Math Balkanica*, **24** (2010), No 1–2, 141–146.

[24] Weller M. Construction of classes of subgroups of small index in p-groups. *Arch. Math.*, **68** (1997), No 2, 89–99.

*Svetlana Topalova*
*Institute of Mathematics and Informatics*
*Bulgarian Academy of Sciences*
*P.O. Box 323*
*5000 Veliko Tarnovo, Bulgaria*
*e-mail:* svetlana@math.bas.bg

*Received July 28, 2016*
*Final Accepted December 12, 2016*