Also available at http://amc.imfm.si
ISSN 1855-3966 (printed edn.), ISSN 1855-3974 (electronic edn.)
ARS MATHEMATICA CONTEMPORANEA 3 (2010) 59–67

Computation of isomorphisms of coherent configurations

Izumi Miyamoto
Department of Computer Science and Media Engineering
University of Yamanashi, Kofu 400-8511, Japan

Received 30 September 2007, accepted 4 February 2010, published online 23 February 2010

Abstract

A program computing isomorphisms between association schemes was applied to speed up the computation of normalizers of permutation groups. A transitive permutation group forms an association scheme while a permutation group which may not be transitive forms a coherent configuration. In this paper we discuss the extension of the above program to compute isomorphisms between coherent configurations and show some typical examples of computing normalizers.

Keywords: Association scheme, permutation group, algebraic computation.

Math. Subj. Class.: 05E30, 20B40

1 Introduction

Computation of isomorphisms between association schemes were considered by the author in [5, 6] in order to classify the isomorphism classes of association schemes of small size. The resulting computer program can be applied to speed up the computation of the normalizers of permutation groups [12, 13] and improvements to the program can be found in [14]. However a transitive permutation group forms an association scheme while a permutation group which may be intransitive forms a coherent configuration. In this paper we discuss the extension of the program written by the author to compute isomorphisms between coherent configurations. Particularly we will show how the program computes isomorphisms between coherent configurations formed by intransitive permutation groups and will show some typical examples of computations as applications.

To begin with we used the relation matrix of a coherent configuration to compute the isomorphisms of the coherent configuration as we used it to compute the isomorphisms of an association schemes [12, 13]. The entries of the relation matrix are the relation numbers

E-mail address: imiyamoto@yamanashi.ac.jp (Izumi Miyamoto) of a coherent configuration. But a coherent configuration may have many relations even though it is of small degree. So if a coherent configuration has more than one fibers (cf. [8]), we set a number to all the relations besides the fibers satisfying a certain condition.

A coherent configuration can be considered as a graph with labeled edges. It is known that when we compute the automorphisms of such a graph, we convert it to some kind of bipartite graph and then we can use some program package for graphs. Our program is written in GAP Programming Language [4] and GAP has a share package named GRAPE [19] for computations with graphs. We will also show how the package GRAPE computes the isomorphisms of coherent configurations in the Appendix.

We should mention about the programming package presented in [3] for computations with coherent configurations. It computes various properties of coherent configurations. The UNIX port of the programming system is named "COCO" and available at [2]. In particular, it computes the order of the automorphism group of a coherent configuration. Recently it has been modified to compute the automorphisms of a coherent configuration [18]. Readers may also refer to [7, 9, 15, 16] for information about coherent configurations.

2 Notations and Preliminaries

Let $\Omega = \{1, 2, \cdots, n\}$ and let R_k, $1 \leq k \leq d$, be subsets of $\Omega \times \Omega$. First we would like to define the system of axioms for the concept of a coherent algebra and its matrix analogue.

Definition 2.1. $(\Omega, \{R_k\}_{k=1,2,\cdots,d})$ is a coherent configuration if it satisfies the following:

CC1. $\{R_1, R_2, \cdots, R_d\}$ is a partition of $\Omega \times \Omega$;

CC2. For some $r < d$, $\{R_1, R_2, \cdots, R_r\}$ is a partition of the diagonal $\{(x, x) | x \in \Omega\}$ of Ω;

CC3. For every k there exists k^* such that $R_{k^*} = \{(y, x) | (x, y) \in R_k\}$;

CC4. There exist constant numbers $p_{i,j,k}$ such that for any $(x, z) \in R_k$ the number of points $y \in \Omega$ such that $(x, y) \in R_i$ and $(y, z) \in R_j$ is equal to $p_{i,j,k}$.

Let A_k, $k = 1, 2, \cdots, d$, be $n \times n$ matrices whose rows and columns are indexed by Ω and whose (x, y) entry $A_k(x, y)$ is defined by

$$A_k(x, y) = \begin{cases} 1 & \text{if } (x, y) \in R_k \\ 0 & \text{otherwise.} \end{cases}$$

Then using these matrices, the definition of a coherent configuration takes the following form.

Definition 2.2. $\{A_k\}_{k=1,2,\cdots,d}$ is a coherent configuration on Ω if the following holds:

CC1. $\sum_{k=1}^{d} A_k = J$ (all 1 matrix);

CC2. $\sum_{k=1}^{r} A_k = I$ (identity matrix) for some $r < d$;

CC3. For every k there exists k^* such that ${}^tA_k = A_{k^*}$;

CC4. $A_i A_j = \sum_{k=1}^{d} p_{i,j,k} A_k$.

We call each R_i a relation and A_i an adjacency matrix. We define the relation matrix A of a coherent configuration $(\Omega, \{R_k\}_{k=1,2,\cdots d})$ by

$$A = 1A_1 + 2A_2 + \cdots + dA_d.$$

Set $\Omega_k = \{x \in \Omega | (x,x) \in R_k\}$ for $1 \leq k \leq r$ in CC2. Then $\{\Omega_k\}_{k=1,2,\cdots,r}$ forms a partition of Ω. We collect relations R_i such that $R_i \subseteq \Omega_k \times \Omega_k$ for each Ω_k, $1 \leq k \leq r$ and denote them $\{R_k, R_{k(1)}, R_{k(2)}, \cdots, R_{k(d_k)}\}$. Then each $(\Omega_k, \{R_k\} \cup \{R_{k(i)}\}_{i=1,2,\cdots,d_k})$ is a coherent configuration and we call it a *fiber* of the configuration $(\Omega, \{R_i\}_{i=1,2,\cdots,d})$. A coherent configuration is said to be *homogeneous* if it has only one fiber, that is, $r = 1$, or equivalently if $A_1 = I$. In particular we use the terminology "*association scheme*" for a homogeneous coherent configuration.

Let g be a permutation on Ω. Then g acts on $\Omega \times \Omega$ naturally by $(x,y)^g = (x^g, y^g)$ for $(x,y) \in \Omega \times \Omega$. For a subset $R \subseteq \Omega \times \Omega$, we define $R^g = \{(x^g, y^g) | (x,y) \in R\}$. Let $(\Omega, \{R_k\}_{k=1,2,\cdots,d})$ and $(\Omega, \{R'_k\}_{k=1,2,\cdots,d'})$ be coherent configurations. We say that they are isomorphic if $d = d'$ and there exists a permutation g on Ω such that for every R_k there exists $R'_{k'}$ such that $R'_{k'} = R_k^g$. Then we can consider two kinds of automorphisms of a coherent configuration. One is a permutation on Ω which fixes every relation R_k of a coherent configuration $(\Omega, \{R_k\}_{k=1,2,\cdots,d})$. The other is a permutation which may move each relation R_k to another relation $R_{k'}$. These kinds of automorphisms are called by various names ([8, 1, 10]). The former is called a strict automorphism, a combinatorial automorphism or simply an automorphism. The latter is called a weak automorphism or a colored automorphism. We note that the third kind of automorphism is considered in [10]. However in the present paper we call the second kind an automorphism for short.

As we mentioned in the introduction, a popular way to construct coherent configurations and association schemes is from permutation groups. Let G be a permutation group on Ω. As noted in the previous paragraph, G acts naturally on $\Omega \times \Omega$. Let $\{R_1, R_2, \cdots, R_d\}$ be the orbits of G defined by this action. Then this partition of $\Omega \times \Omega$ makes a coherent configuration. Here we arrange $R_k \subseteq \{(x,x) | x \in \Omega\}$ for $1 \leq k \leq r$. In particular if G is transitive, then $R_1 = \{(x,x) | x \in \Omega\}$ and the partition becomes an association scheme.

3 Computation of isomorphisms between coherent configurations and automorphisms

Let $(\Omega, \{R_i\}_{i=1,2,\cdots,d})$ be a coherent configuration and let A be the relation matrix of the configuration. We may sometimes abuse notation by calling the relation matrix A a coherent configuration. Our algorithm to compute isomorphisms and automorphisms is based on that in [13]. We use relation matrices of coherent configurations to compute with them. However, as is seen in an example in the following section, a coherent configuration may have too many relations, even if its degree n is not so large. So we modify the relation matrix using the fibers of the configuration. Let $\{\Omega_k\}_{k=1,2,\cdots,r}$ be the partition of Ω defining the fibers of a coherent configuration $(\Omega, \{R_i\}_{i=1,2,\cdots,d})$, where its adjacency matrices satisfy $\sum_{i=1}^{r} A_i = I$ (identity matrix). We consider the union of the relations R_i such that $R_i = \Omega_j \times \Omega_k$ for some subsets Ω_j and Ω_k stated above with $j, k \leq r$ and $j \neq k$. We set $R'_{d'+1}$ to be the union of such relations, if they exist, where the remaining relations are $R'_1, R'_2, \cdots, R'_{d'}$ satisfying $R'_i \neq \Omega_j \times \Omega_k$ for any $j, k \leq r, j \neq k$. We use the relation matrix A' made by the partition $\{R'_1, R'_2, \cdots, R'_{d'}, R'_{d'+1}\}$ to compute the automorphisms of the coherent configuration. We may arrange the relations of A so that $R_i = R'_i$ if $1 \leq i \leq d'$ and $R_i = \Omega_j \times \Omega_k$ for some $j, k \leq r, j \neq k$ if $d' + 1 \leq i \leq d$. The following is an example of A and A'.

$$A = \begin{pmatrix} 1 & 4 & 7 & 9 & 11 & 11 \\ 4 & 1 & 9 & 7 & 11 & 11 \\ 8 & 10 & 2 & 5 & 13 & 13 \\ 10 & 8 & 5 & 2 & 13 & 13 \\ 12 & 12 & 14 & 14 & 3 & 6 \\ 12 & 12 & 14 & 14 & 6 & 3 \end{pmatrix} \quad A' = \begin{pmatrix} 1 & 4 & 7 & 9 & 11 & 11 \\ 4 & 1 & 9 & 7 & 11 & 11 \\ 8 & 10 & 2 & 5 & 11 & 11 \\ 10 & 8 & 5 & 2 & 11 & 11 \\ 11 & 11 & 11 & 11 & 3 & 6 \\ 11 & 11 & 11 & 11 & 6 & 3 \end{pmatrix}$$

Here A is formed by a group $G =$Group$((1,2)(3,4),(5,6))$. A has 3 fibers and 14 relations. $\Omega_1 = \{1,2\}$, $\Omega_2 = \{3,4\}$ and $\Omega_3 = \{5,6\}$. The relations R_i, $i \geq 7$ are not contained in the fibers. Among these relations, if $i \geq 11$, $R_i = \Omega_j \times \Omega_k$ for $(j,k) = (1,3)$, $(2,3)$, $(3,1)$ or $(3,2)$, while $R_7 \cup R_9 = \Omega_1 \times \Omega_2$ and $R_8 \cup R_{10} = \Omega_2 \times \Omega_1$. So $R_i = R'_i$ for $1 \leq i \leq 10$ and $R'_{11} = R_{11} \cup R_{12} \cup R_{13} \cup R_{14}$.

Let g be an automorphism of A' and we will show that g is an automorphism of A. First we note that $R'^g_{d'+1} = R'_{d'+1}$, since any R_i is contained in some $\Omega_j \times \Omega_k$. For any relation R_i of A, let $(x,y) \in R_i$ and suppose that $(x^g, y^g) \in R_{i'}$ for some relation $R_{i'}$ of A. Then we will see $R_i^g = R_{i'}$. If $i \leq d'$, then $R_i = R'_i$ and we have $R_i^g = R'^g_i = R'_{i'} = R_{i'}$. If $i \geq d' + 1$, there exist subsets Ω_j and Ω_k stated above for some $j, k \leq r, j \neq k$ such that $(x,y) \in R_i = \Omega_j \times \Omega_k$. Let $x^g \in \Omega_{j'}$ and let $y^g \in \Omega_{k'}$. Then considering the action of g on the diagonals of the fibers, we see that $\{(z,z) | z \in \Omega_j\}^g = \{(z',z') | z' \in \Omega_{j'}\}$ and $\{(w,w) | w \in \Omega_k\}^g = \{(w',w') | w' \in \Omega_{k'}\}$. This implies that $R_i^g = (\Omega_j \times \Omega_k)^g = \Omega_{j'} \times \Omega_{k'} = R_{i'}$. Therefore g becomes an automorphism obtained from the relation matrix A of the configuration $(\Omega, \{R_i\}_{i=1,2,\cdots,d})$. This computation is shown in Example 3 in the next section.

We apply our program to compute normalizers and the results of our experiments are shown in the next section. In [12, 13] the automorphism groups of the association schemes formed by the actions on the orbits of an intransitive group are computed and the direct product of them together with the isomorphisms among the association schemes is considered, while this time we can compute the automorphism group of the coherent configuration formed by the intransitive group directly. We note that there exist not a few groups of which normalizers are hard to compute even though they are of small degree. Generally our program computes normalizers of such groups smoothly except for a few groups. However Example 1 in the next section may explain the difficulty of the computation of normalizers. In contrast to normalizers, automorphism groups of coherent configurations were computed quickly if they do not have a large number of relations.

The automorphisms of coherent configurations can be computed by a program package for computations with graphs suggested by [17]. We have done an experiment using the program package GRAPE. The computation is shown in the Appendix.

4 Experiments

Let G be a permutation group on the set Ω and let A be the coherent configuration formed by G. Let $Sym(n)$ denote the symmetric group of degree $n = |\Omega|$. The normalizer of a group G in a group H is defined by $\textsc{Norm}(H,G) = \{h \in H | h^{-1}Gh = G\}$. The normalizer permutes the orbits of G on $\Omega \times \Omega$. So the normalizer of G in $Sym(n)$ is contained in the automorphism group of the coherent configuration A formed by G by definition.

We would like to explain how the GAP-default function NORMALIZER, which will be abbreviated as NORM below, computes a normalizer in the symmetric group briefly in an easy example. Suppose that G has two orbits of different size n_1 and n_2, $n_1 + n_2 = n$. GAP computes the actions G_1 and G_2 of G on the two orbits of G. Then it computes N_i =NORM$(Sym(n_i), G_i)$ for $i = 1, 2$. Moreover if G_2 is imprimitive on the orbit of size n_2 with only one block system of block size n', then the normalizer N_2 of G_2 in $Sym(n_2)$ is computed in advance by NORM in the group WREATHPRODUCT$(Sym(n') \times Sym(n_2/n'))$. After these computations it computes the normalizer of G in $N_1 \times N_2$ by using another GAP-function AUTOMORPHISMGROUPPERMGROUP, which computes a normalizer directly.

Example 1. Let K =PRIMITIVEGROUP$(112, 1) \cong PSU(4, 3)$ in the GAP library, and let G =STABILIZER$(K, 112) = \{g \in K | 112^g = 112\}$. Then we have the following data. G is of degree $n = |\Omega| = 111$ and of order $|G|$ =29160. G has two orbits of length 81 and 30, which are denoted by Ω_1 and Ω_2 respectively. G is faithful on both of the orbits. We compute the normalizer N of G in $Sym(111)$ by various methods.

First, we compute it directly by the GAP-function AUTOMORPHISMGROUPPERM-GROUP. It took less than 0.5 seconds for this computation.

Second, we use the GAP-default function NORM. It took too long for this computation, so we interrupted the computation and we looked into each step of this computation explained above. The action G_2 of G on Ω_2 is imprimitive and has only one block system of block size 3. So GAP computes N_2 =NORM(WREATHPRODUCT$(Sym(3), Sym(10))$, $G_2)$. But it took about 4 hours for this computation. After this computation it took about 15 seconds to compute the normalizer N =NORM$(Sym(81) \times N_2, G)$.

Third, we used the program for computation of association schemes [12, 13]. G has two orbits Ω_1 and Ω_2 and the actions of G on these orbits are denoted by G_1 and G_2, respectively as above. So we have two association schemes A_1 and A_2 formed by G_1 and G_2, we can compute their automorphism groups $Aut(A_1)$ and $Aut(A_2)$ by the program in [12, 13] and we will compute N_i =NORM$(Aut(A_i), G_i)$ for $i = 1, 2$. Then we can compute N =NORM$(N_1 \times N_2, G)$. Following this line of computation, we obtained $Aut(A_1)$ of order 233280 and $Aut(A_2)$ =WREATHPRODUCT $(Sym(3), Sym(10))$ of order 219419659468800. At this step we should compute the same difficult normalizer N_2 =NORM(WREATHPRODUCT$(Sym(3), Sym(10)), G_2)$ as in the second computation. The remaining computation was done quickly.

Finally, we construct a coherent configuration A from G and use our program to compute the automorphism group $Aut(A)$ of A. G has 11 orbits on $\Omega \times \Omega$. The relation matrix A is of size $111 \times 111 = 12321$. The matrix for the constants $(p_{i,j,k})$ is of size $11 \times 11 \times 11 = 1331$. It took 7 seconds to compute $Aut(A)$ by our program and $|Aut(A)| = 233280$. Then the computation of N =NORM$(Aut(A), G)$ was done quickly. We note that G is normal in $Aut(A)$ in this example.

Example 2. Let G be the group generated by the permutations

$$(3,16)(4,15)(5,17)(6,18)(7,8)(9,21)(10,22)(11,12)(19,20)(23,24),$$
$$(1,14)(2,13)(5,6)(7,8)(9,22)(10,21)(11,24)(12,23)(17,18)(19,20),$$
$$(1,2)(3,15)(4,16)(7,20)(8,19)(9,21)(10,22)(11,24)(12,23)(13,14).$$

Then G is elementary abelian of order 8 and degree 24, There exist 6 orbits of G of length 4. So $\textsc{Norm}(Sym(24), G) = \textsc{Norm}(\textsc{WreathProduct}(Sym(4), Sym(6)), G)$. It took about 3 minutes to compute this normalizer. Let A be the coherent configuration formed by G. Then $Aut(A)$ is of order 98304. It took 3 seconds to compute $Aut(A)$ and $\textsc{Norm}(Aut(A), G)$. The 6 association schemes which consist of the fibers of A are mutually isomorphic and their automorphism groups are $Sym(4)$. So if we do not use a coherent configuration but use association schemes as in [12], we have to compute $\textsc{Norm}(\textsc{WreathProduct}(Sym(4), Sym(6)), G)$, which is the same as by the GAP-function $\textsc{Norm}$.

Example 3. Let H =$\textsc{TransitiveGroup}(30, 1075)$ in the GAP library and let $G = \{g \in H | 1^g = 1 \text{ and } 15^g = 15\}$, the stabilizer of the points 1 and 15 in H. G stabilizes two more points. G is of order 64 and of degree 26. All the orbits of G are of length 2. G has 186 orbits on $\Omega \times \Omega$. So the relation matrix A of the coherent configuration formed by G is of size $26 \times 26 = 676$, while the matrix of the constant $(p_{i,j,k})$ is of size $186 \times 186 \times 186 = 6434856$. It took about 9 minutes to compute $Aut(A)$, and $|Aut(A)| = 23781703680$. Here we consider the partition $\{R'_i\}_{i=1,\cdots,35}$ introduced in the previous section associated with this case. Let A' be the relation matrix defined by this partition. Then it took 1 second to compute $Aut(A')$. In this case it took about 100 seconds to compute each of $\textsc{Norm}(Aut(A), G)$, $\textsc{Norm}(Sym(26), G)$ and $\textsc{AutomorphismGroupPermGroup}(Sym(26), G)$.

As for computing the normalizers of permutation groups, the author considered various methods [12, 14]. In [12], as is shown in Examples 1 and 2, we construct an association scheme from each orbit of an intransitive group, compute the automorphism groups and consider the direct product of them and the isomorphisms among the association schemes. Here in our program we construct a coherent configuration from the intransitive group and compute the automorphism group instead of the computations relating the association schemes stated above. In Table 1, CCN denotes our method to compute normalizers using an automorphism group of a coherent configuration, ASN denotes the method to compute normalizers using the direct product of automorphism groups of association schemes and the isomorphisms among them shown in [12] and $\textsc{Norm}$ denotes the GAP function to compute normalizers. Our program CCN took more time in some cases in Table 1. We explain briefly about these computations. When we compute the normalizer of H =$\textsc{TransitiveGroup}(27, 1799)$, we use the normalizer of the subgroup $G = \{g \in H | 1^g = 1 \text{ and } 2^g = 2\}$ of H. The orbits of G are $\{\{3, 7\}, \{4, 8\}, \{5, 6\}, \{10, 16, 11, 15, 17, 12, 13, 18, 14\}, \{19, 24, 27, 20, 23, 25, 26, 22, 21\}\}$. Here we see the actions of G on the orbits of length 9. The actions of G on the two orbits of length 9 are E(9):2D_8 of order 144, which is doubly transitive, and M(9)=E(9):Q_8 of order 72, which is also doubly transitive. So the direct sum of two association schemes of degree 9 coincides with the coherent configuration of degree 18 in this case, and the automorphism group interchanges these two orbits of length 9. But the normalizer leaves these two orbits fixed, since the two actions are not mutually isomorphic. In ASN, we check the orders of the actions, while in CCN, we simply compute the automorphism group Aut of the coherent configuration. This is why it took more time for the computation by CCN in this case. We note that Aut was always easily computed within 0.2 seconds, but it was still hard to compute $\textsc{Norm}(Aut, G)$ in some cases. These facts explain the times in Table 1.

n	k	CCN	ASN	NORM	n	k	CCN	ASN	NORM
24	2827	7	145	192	24	24924	5764	0.5	1852
24	2951	0.5	66	>10h	27	1799	10755	0.5	>10h
24	7232	0.7	145	387	27	1822	11160	0.5	>10h
27	333	0.5	142	348	28	413	813	6	446
27	817	2	125	14040	30	494	1285	0.2	0.2
28	160	1	4004	2694	30	512	1255	0.3	1
28	238	1	2707	1401	30	527	1288	0.4	1
28	273	2	5974	3588	30	528	1277	0.3	0.2
28	346	3	5044	1939	30	1075	827	0.3	0.2
30	143	4	416	>10h	30	1083	807	0.3	0.3
30	628	4	151	304	30	1103	816	0.3	0.2
30	1268	25	433	>10h	30	4898	13742	0.7	>10h
30	1721	5	367	>10h	30	4899	55084	0.7	>10h

Table 1: Computing times of some normalizers of TRANSITIVEGROUP(n, k) in $Sym(n)$. (In seconds, ">10h" means more than 10 hours.)

5 Concluding remarks

We use an array of size d^3 to compute the permutations of the relations $\{R_i\}$ induced from isomorphisms. It is hard to compute the array of $p_{i,j,k}$ of size d^3, because d can be large compared to n the size of A. Even though we consider the partition $\{R'_i\}$, the size d'^3 can still be large. For instance, since most of $p_{i,j,k}$'s are 0, in GAP, an array is used such that the (i, j) entry consists of two lists where the indices k with $p_{i,j,k} \neq 0$ and the values $p_{i,j,k}$ are stored. So we think that some improvement can be made in computing the permutation on $\{R'_i\}$.

6 Appendix

A coherent configuration is a labeled graph and we permit permutation of the labels in computing its automorphisms. Similarly, as to hypergraphs, the edges and the labels of a coherent configuration are converted to vertices of some kind of bipartite graph and we use a program computing automorphism groups of graphs in order to compute automorphism groups of coherent configurations. NAUTY [11] is a program for computations with graphs, which computes automorphism groups of graphs and digraphs. GRAPE [19] is a program package which provides the interface to NAUTY for the GAP system. We have done an experiment to compute automorphism groups of association schemes A formed by transitive permutation groups G of small degree in the GAP library. The following rough sketch of our program using GRAPE is suggested by [17].

```
VERTICES are tuples:
    [1], [2], $\cdots$, [$n$] for original points of $\Omega$,
    ARRANGEMENTS$([1..n], 2)$ for distinct ordered pairs of points,
    $[n + 1], [n + 2], \cdots, [n + d]$ for relations.
EDGES:
    $[i] \to [i, j]$, $[i, j] \to [j]$,
    $[i] \to [n + k]$ if $(i, i) \in R_k$ or equivalently $A(i, i) = k$,
```

$[i, j] \to [n + k]$ if $(i, j) \in R_k$ or equivalently $A(i, j) = k$.

$\widetilde{Aut}$ =AUTGROUPGRAPH(G,VERTICES,ONTUPLES,
FUNCTION(u, v) return ISEDGE(u, v, A);end).
Aut =ACTION($\widetilde{Aut}$, [1..n]).

In most cases the automorphism groups Aut can be computed very quickly within 0.2 seconds. But some association schemes are not computed within a reasonable time. For instance, the association scheme formed by WREATHPRODUCT(TRANSITIVEGROUP$(6, 4)$, TRANSITIVEGROUP$(4, 2)$) takes more than 4400 minutes to compute its automorphism group. In the following groups the automorphism groups of the association schemes formed by these groups could not be computed within 5 minutes and computations were stopped:

- WREATHPRODUCT($Sym(4)$, TRANSITIVEGROUP$(4, 2)$);
- WREATHPRODUCT($Sym(4)$, TRANSITIVEGROUP$(5, 2)$);
- WREATHPRODUCT(TRANSITIVEGROUP$(6, 7)$, TRANSITIVEGROUP$(4, 2)$).

Our program computed these automorphism groups within 0.2 seconds each.

7 Acknowledgment

The author would like to express his thanks to Prof. Ted Dobson for his kind advice.

References

[1] P. J. Cameron. Coherent configurations, association schemes and permutation groups, in: A. A. Ivanov, Martin W. Liebeck, Jan Saxl (eds.), *Groups, combinatorics & geometry*, Durham, 2001, 55–72.

[2] I. A. Faradžev in cooperation with M. H. Klin, Unix port: A. E. Brouwer, COCO computer algebra system, http://www.win.tue.nl/~aeb/.

[3] I. A. Faradžev and M. H. Klin, Computer package for computations with coherent configurations, in: *Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation*, 1991, 219–223.

[4] The GAP Groups, *Gap – groups, algorithms and programming, version 4*, Lehrstuhl D für Mathematik, Rheinisch Westfälische Technische Hochschule, Aachen, Germany and School of Mathematical and Computational Sciences, Univ. St. Andrews, Scotland, 1997.

[5] A. Hanaki and I. Miyamoto, Data of association schemes, published at http://kissme.shinshu-u.ac.jp/as/, 1999–.

[6] A. Hanaki and I. Miyamoto, Classification of association schemes of small order, *Discrete Math.*, **264** (2003), 75–80.

[7] D. G. Higman, Coherent configurations I, *Rendiconti del Seminario Matematico della Università di Padova*, **44** (1970), 1–25.

[8] D. G. Higman, Coherent algebras, *Linear Algebra Appl.* **93** (1987), 209–239.

[9] M. Klin, Tutorial "Coherent configurations and association schemes, Part I: Definitions, examples, simple facts", http://www.ricam.oeaw.ac.at/specsem/srs/groeb/schedule_D1.html, 2006.

[10] M. Klin, M. Muzychuk, C. Pech, A. Woldar and P.-H. Zieschang, Association schemes on 28 points as mergings of a half- homogeneous coherent configuration, *European J. Combin.* **28** (2007), 1994–2025.

[11] B. D. McKay, nauty, `http://cs.anu.edu.au/~bdm/nauty/`.

[12] I. Miyamoto, Computing normalizers of permutation groups efficiently using isomorphisms of association schemes, in: *Proceedings of the 2000 International Symposium on Symbolic and Algebraic Computation*, 2000, 200–204.

[13] I. Miyamoto, Computing isomorphisms of association schemes and its applications, *J. Symbolic Comp.* **32** (2001), 133–141.

[14] I. Miyamoto, An improvement of GAP Normalizer function for permutation groups, in: *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, 2006, 234–238.

[15] M. Muzychuk, Tutorial "Coherent configurations and association schemes, Part II: Spectral properties and mergings of classes", `http://www.ricam.oeaw.ac.at/specsem/srs/groeb/schedule_D1.html`.

[16] C. Pech, Tutorial "Coherent configurations and association schemes, Part III: Galois correspondence between permutation groups and coherent configurations", `http://www.ricam.oeaw.ac.at/specsem/srs/groeb/schedule_D1.html`.

[17] I. Ponomarenko, private communication.

[18] S. Reichard, private communication.

[19] L. H. Soicher, GAP package GRAPE – GRaph Algorithms using PErmutation groups, `http://www.gap-system.org/Packages/grape.html`.