A Group-Theoretic Method for Drawing Graphs Symmetrically*

David Abelson[1], Seok-Hee Hong[2], and Donald E. Taylor[1]

[1] School of Mathematics and Statistics, University of Sydney, Australia,
dabe4090@mail.usyd.edu.au, D.Taylor@maths.usyd.edu.au
[2] School of Information Technologies, University of Sydney, Australia,
shhong@it.usyd.edu.au

Abstract. Constructing symmetric drawings of graphs is NP-hard. In this paper, we present a new method for drawing graphs symmetrically based on group theory. More formally, we define a *n-geometric automorphism group* of a graph that can be displayed as symmetries of a drawing of the graph in n dimensions. Then we present an algorithm to find *all* 2- and 3-geometric automorphism groups of a graph. We implement the algorithm using `Magma` [11] and the experimental results shows that our approach is very efficient in practice. We also present a drawing algorithm to display a 2- or 3-geometric automorphism group.

1 Introduction

Symmetry is one of the most important aesthetic criteria that represent the structure and properties of a graph visually. To construct a symmetric drawing of a graph, we need two steps. The first step, called the *symmetry finding step*, is to find *geometric automorphisms* of the graph, which can be displayed as symmetries of a drawing of the graph. The second step, called the *drawing step*, is to construct a drawing that displays these automorphisms.

However, the problem of determining whether a graph has a nontrivial *strict geometric* automorphism in two dimensions is NP-complete [8]; it is probably strictly harder than graph isomorphism. For planar graphs, there is a polynomial time algorithm [5]. The problem of determining whether a graph can be drawn symmetrically in three dimensions is also NP-complete [6] and a polynomial time algorithm is given for planar graphs [6].

In this paper, we present a group-theoretic method to find *all* 2- and 3-geometric automorphism groups of a graph. First we define an n-geometric automorphism group of a graph that can be displayed as symmetries of a drawing of the graph in n dimensions.

Next we present an algorithm to find all 2- and 3-geometric automorphism groups of a graph, based on the classification of the 2- and 3-geometric automorphism groups in [1]. This is done by first calculating the automorphism group

* This research has been supported by a grant from the Australian Research Council. Full version of this paper is available from http://www.cs.usyd.edu.au/∼shhong/publication.htm [1].

M.T. Goodrich and S.G. Kobourov (Eds.): GD 2002, LNCS 2528, pp. 86–97, 2002.

$\mathrm{Aut}(G)$ of a graph G and then looking for subgroups satisfying the classification. More specifically, we use *conjugacy classes* of the automorphism group to find the subgroups.

The main contribution of this paper is that we provide an algorithm for finding *all* 2- and 3-geometric automorphism groups of a graph. A heuristic for finding an axial symmetry in two dimensions is presented by de Fraysseix [4]. Buchheim and Junger present a branch and cut approach to find either a rotational symmetry or an axial symmetry in two dimensions [2]. However, our method is the first to find *dihedral* groups, which have two generators, in two dimensions and *fourteen* different kinds of groups, with up to three generators, in three dimensions.

We implement the algorithm using `Magma` [11], a computational algebra system for algebra, number theory, geometry and combinatorics, which incorporates `nauty` [12] for finding automorphism groups of graphs. The worst case time complexity is exponential in theory, but experiments show in practice it is very fast. For example, for our first data set, which consists of graphs with $|V| < 50$ and $|\mathrm{Aut}(G)| < 1000$, it takes 0.022 seconds to compute all 2-geometric groups and 0.037 seconds to compute all 3-geometric groups on average. For our second data set, which consists of graphs with $|V| < 50$ and $1,000 < |\mathrm{Aut}(G)| < 51,000,000$, it takes 0.41 seconds to compute all 2-geometric groups and 2.58 seconds to compute all 3-geometric groups on average. Further, we show that our method is much faster than the branch and cut approach [2]. The average times for the graphs from the same data set of [2] are below 0.015 seconds. The worst case time is 2.60 seconds, whereas the worst case time of [2] was 9197.55 seconds.

Finally we present and implement a drawing algorithm to display a 2- or 3-geometric subgroup. Examples of the outputs are illustrated in Figure 1. It shows a drawing of four-cell displaying dihedral symmetry in two dimensions and a drawing of four-cube displaying octahedral symmetry in three dimensions.

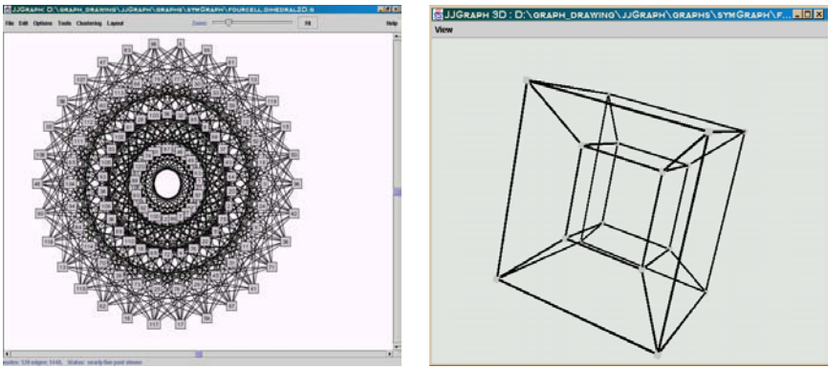

Fig. 1. Examples of our results.

In the next section, we review the background and define n-geometric automorphism group. In Section 3, we describe our methods for finding 2- and 3-geometric automorphism groups. The drawing algorithm is described in Section 4 and Section 5 concludes.

2 Background

2.1 Symmetry

Let $I_n(\mathbb{R})$ be the group of *isometries* of $\mathbb{R}^n$ and let $O_n(\mathbb{R})$ be the subgroup of $I_n(\mathbb{R})$ that fixes the origin. A matrix A is *orthogonal* if and only if $A.A^T = I$. The elements of $O_n(\mathbb{R})$ are represented by orthogonal matrices and we use $SO_n(\mathbb{R})$ to denote the subgroup corresponding to the matrices of determinant 1.

A *symmetry* α of a set of points Q in $\mathbb{R}^n$ is an isometry $\mathbb{R}^n \to \mathbb{R}^n$ such that $\alpha(Q) = Q$. The symmetries of the point set Q form a group $S(Q)$ and by translating Q if necessary we may suppose that $S(Q)$ is a subgroup of $O_n(\mathbb{R})$.

2.2 Symmetric Graph Drawing and Geometric Automorphism Group

An *automorphism* of a graph $G = (V, E)$ is a permutation p of V such that if $\{u, v\} \in E$ then $\{p(u), p(v)\} \in E$. The set of automorphisms of a graph form a group $\mathrm{Aut}(G)$. A *straight-line* drawing D of graph G is an injective function $D : V \to \mathbb{R}^n (n \geq 0)$. A vertex v is placed at $D(v)$ and an edge $\{u, v\}$ is represented as the straight-line segment joining $D(u)$ and $D(v)$.

The concept of geometric automorphism group in two dimensions was introduced by Eades and Lin [3]. An automorphism α of a graph G is *geometric* if there is a drawing D of G which *displays* α as a symmetry of D. Note that not every automorphism is geometric; see Figure 2(a). A subgroup H of $\mathrm{Aut}(G)$ is *geometric* if there is a single drawing of the graph that displays every element of H. Note that the two geometric automorphism groups may not be combinable; see Figure 2(b) and (c).

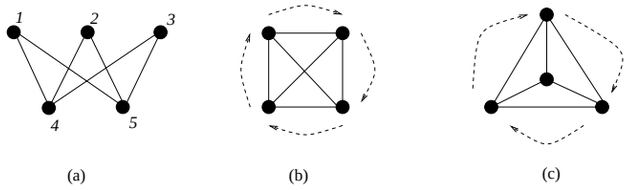

Fig. 2. (a) The automorphism (1,2,3)(4,5) is not geometric. (b) A drawing of K_4 displaying a geometric automorphism group of size 8. (c) A drawing of K_4 displaying a geometric automorphism group of size 6.

2.3 The n-Geometric Automorphism Group

We generalize the notion of a geometric automorphism group [3] to n dimensions. A group $H \subseteq \mathrm{Aut}(G)$ is *n-geometric* if there is a drawing D that displays all the elements of H in n dimensions. Note that because we always assume that D is centered on the origin, H will be represented as a subgroup of $O_n(\mathbb{R})$.

Lemma 1. *A group $H \subseteq \mathrm{Aut}(G)$ is n-geometric with respect to a drawing D if and only if there exists a homomorphism $\phi : H \to O_n(\mathbb{R})$ such that for all $v \in V$ and $h \in H$, $D(hv) = \phi(h)D(v)$.*

A homomorphism $\phi : H \to O_n(\mathbb{R})$ is called a *representation* of H. In matrix terms, each element of H is represented by an orthogonal matrix. The representation is *faithful* if ϕ is injective.

We use basic terminology from group theory [10]. Let H be a group acting on a set U. For $u \in U$, let $Hu = \{x \in U \mid x = gu \text{ for some } g \in H\} \subseteq U$ be the *orbit* of u and $H_u = \{g \in H \mid g(u) = u\} \subseteq H$ be the *stabilizer* of u. The orbits divide the set U into equivalence classes, each of which can be specified by a *representative* element. Two elements a and b of a group H are *conjugate* if there exists $h \in H$ such that $b = hah^{-1}$. *Conjugacy* is an equivalence relation and the equivalence classes of H are called *conjugacy classes*. We now characterize n-geometric groups.

Theorem 1. *A group $H \subseteq \mathrm{Aut}(G)$ is n-geometric if and only if there is an injective homomorphism $\theta : H \to O_n(\mathbb{R})$ such that for a representative $v_1, \ldots, v_k \in V$ of each orbit of H acting on V, there are corresponding distinct points $a_1, \ldots, a_k \in \mathbb{R}^n$ such that $\theta(H_{v_i}) = \theta(H)_{a_i}$.*

3 Finding Geometric Automorphism Groups

We now outline our method of finding all 2- and 3-geometric automorphism groups based on the classification in [1]. It follows from Theorem 1 that a permutation group A acting on a graph is n-geometric if it is isomorphic (as an abstract group) to a finite subgroup H of $O_n(\mathbb{R})$ and

1. if A fixes more than one vertex, then H fixes a point other than the origin;
2. for every vertex v, the stabilizer A_v is isomorphic to the stabilizer in H of a point of $\mathbb{R}^n$.

When the value of n is clear from the context we shall simply refer to A as a geometric group. The group H will be referred to as the *type* of A.

Our search for the geometric subgroups of Aut(G) is facilitated by the fortunate fact that all of the finite subgroups of $O_2(\mathbb{R})$ and $O_3(\mathbb{R})$ have very simple presentations. That is, they can be expressed in the form $H = \langle X \mid \mathcal{R} \rangle$, where X is a list of generators for H and $\mathcal{R}$ is a list of relations between the generators. Furthermore, there are at most three generators and in many cases the relations simply specify the orders of the generators and the orders of their products.

3.1 The Finding Algorithm

To find the geometric subgroups of the automorphism group Aut(G) of a graph $G = (V, E)$ we look for elements that satisfy the relations of its presentation. We are only interested in finding these subgroups up to conjugacy because, as the next lemma shows, conjugate n-geometric subgroups have essentially the same drawings. Let $\mathrm{Sym}(V)$ be the group of all permutations of V.

Lemma 2. *Let H be an n-geometric subgroup of Aut(G) with respect to a drawing D and suppose that $H' = g^{-1}Hg \subseteq \mathrm{Aut}(G)$, where $g \in \mathrm{Sym}(V)$. Then H' is n-geometric with respect to the drawing D' defined by $D'(v) = D(gv)$.*

Computing the conjugacy classes of a group is a hard problem but highly optimized algorithms for this purpose are part of `Magma`. Moreover, in searching for geometric automorphisms it is often possible to restrict to a proper subgroup of Aut(G). To take a simple example, if g is a 2-geometric automorphism of order $m > 2$, then g has at most one fixed point and all the other cycles have length m. Therefore, if we were looking for a 2-geometric automorphism of order 3, say, and if the group had an orbit of length 16, then we could confine our search to the stabilizer of a point in the orbit. These and similar considerations speed up the search considerably but for clarity we leave them out of our descriptions of the algorithms.

2-Geometric Subgroups. In two dimensions the only geometric groups are the cyclic groups $\mathcal{C}_n$ of order n and the dihedral groups $\mathcal{D}_n$ of order $2n$. In all cases, the first step in finding these is to compute the automorphism group Aut(G), in short A, of the graph G.

`Algorithm for the 2-geometric cyclic groups`

1. Use the orbit lengths of Aut(G) to compute an upper bound for the order of a 2-geometric element.
2. Find representatives for the conjugacy classes of Aut(G).
3. Accept all elements of order 2 found in the previous step and all elements of order $m > 2$ with at most one fixed point and all other cycles of length m.

Every pair of elements of order 2 generates a dihedral group. We makes use of the elements found in the previous algorithm, as follows (The *normalizer* $N_A(H)$ of a subgroup H is defined to be set of elements $a \in A$ such that $a^{-1}Ha = H$).

`Algorithm for the 2-geometric dihedral groups`

1. Find representatives for the conjugacy classes of geometric elements using the previous algorithm. In this case we also require that any elements of order 2 fix at most one point.
2. For each element g found in the previous step, compute the normalizer N of $\langle\, g \,\rangle$ in Aut(G).
3. Find representatives for the conjugacy classes of elements of order 2 in N.
4. Accept those elements a found in the previous step that satisfy the relation $(ga)^2 = 1$. The group $\langle\, g, a \,\rangle$ is a 2-geometric dihedral group.
5. Carry out some additional checking to choose a single representative for dihedral groups that are conjugate within N.

3-Geometric Subgroups. There are five types of geometric groups that can be represented by rotations in $\mathbb{R}^3$: cyclic groups $\mathcal{C}_n$, dihedral groups $\mathcal{D}_n$, the tetrahedral group $\mathcal{T}$, the octahedral group $\mathcal{O}$, and the icosahedral group $\mathcal{I}$. Furthermore, $\mathcal{D}_n$ has presentation $\langle\, x, y \mid x^2 = y^2 = (xy)^n = 1 \,\rangle$ and the groups $\mathcal{T}$, $\mathcal{O}$ and $\mathcal{I}$ have presentations $\langle\, x, y \mid x^2 = y^3 = (xy)^k = 1 \,\rangle$, where k is 3, 4 or 5, respectively. Since $\mathcal{D}_1 = \mathcal{C}_2$, we shall only use the type $\mathcal{D}_n$ when n is at least 2.

From each rotation group B we get a larger group B^* by taking the direct product of B with the central inversion $-I$. If we have a geometric subgroup H of type B we find candidates for the groups of type B^* by looking inside the *centralizer* $C_A(H) = \{\, a \in A \mid ah = ha \text{ for all } h \in H \,\}$ for elements of order 2 with at most one fixed point; such an element will be represented by the central inversion. This step is straightforward (but potentially expensive) and so we omit it from the descriptions that follow.

In addition to the groups just described, there are four other types that do not consist entirely of rotations. They can be described by symbols $(H \mid K)$. This means that the group itself is isomorphic to a group of type H and contains a subgroup (of rotations) of index 2 of type K. The possible types are $(\mathcal{C}_{2n} \mid \mathcal{C}_n)$, $(\mathcal{D}_n \mid \mathcal{C}_n)$, $(\mathcal{D}_{2n} \mid \mathcal{D}_n)$ and $(\mathcal{O} \mid \mathcal{T})$.

We note that it is possible for a permutation group A to be represented as a 3-geometric group in more than one way. For example, a cyclic group of order $4m$ that fixes at most one point and with all other orbits of length $4m$ has types $\mathcal{C}_{4m}$ and $(\mathcal{C}_{4m} \mid \mathcal{C}_{2m})$.

`Algorithm for the 3-geometric groups of types` $\mathcal{C}_n$ `and` $(\mathcal{C}_{2n} \mid \mathcal{C}_n)$

1. Find representatives for the conjugacy classes of A and accept those elements g of order n all of whose cycles have length 1 or n. The group $\langle\, g \,\rangle$ is 3-geometric of type $\mathcal{C}_n$.
   But if we include the requirement that no vertex in a drawing lies on an edge joining two other vertices, then additional work needs to be done.
2. If n is even, find all elements $h \in C_A(g)$ with at most one fixed point and whose square is g. The lengths of the cycles of h will be 1, 2 and $2n$. The subgroup $\langle\, h \,\rangle$ is 3-geometric of type $(\mathcal{C}_{2n} \mid \mathcal{C}_n)$.

`Algorithm for the 3-geometric groups of types` $\mathcal{D}_n$, $(\mathcal{D}_n \mid \mathcal{C}_n)$ `and` $(\mathcal{D}_{2n} \mid \mathcal{D}_n)$

1. Find representatives for the conjugacy classes of geometric elements using the previous algorithm.
2. For each element g found in the previous step, compute the normalizer N of $\langle\, g \,\rangle$ in Aut(G).
3. Find representatives for the conjugacy classes of elements of order 2 in N.
4. Accept those elements a found in the previous step that satisfy the relation $(ga)^2 = 1$. The group $H = \langle\, g, a \,\rangle$ is a dihedral group.
5. If H fixes more than one point we require all the orbits of H to have lengths 1, n or $2n$. Then H is a group of type $(\mathcal{D}_n \mid \mathcal{C}_n)$.
6. If H fixes at most one point and $n = 2m$, then it is of type $(\mathcal{D}_{2m} \mid \mathcal{D}_m)$.

7. If H fixes at most one point and if the cycles of g have lengths 1 or n, then H is of type $\mathcal{D}_n$ and also of type $(\mathcal{D}_n \mid \mathcal{C}_n)$.
8. Carry out some additional checking to choose a single representative for dihedral groups that are conjugate within N.

We give the details for type $\mathcal{I}$. The other cases are similar except that a group with presentation $\langle\, x, y \mid x^2 = y^3 = (xy)^4 = 1 \,\rangle$ may correspond to a geometric group of type $\mathcal{O}$ or to one of type $(\mathcal{O} \mid \mathcal{T})$ depending on the nature of its point stabilizers.

`Algorithm for 3-geometric groups of types` $\mathcal{T}$, $\mathcal{O}$, $\mathcal{I}$ `and` $(\mathcal{O} \mid \mathcal{T})$

1. Find representative for the conjugacy classes of elements of order 5. These elements are necessarily 3-geometric.
2. For each element g found in the previous step, find representatives for the conjugacy classes of the elements of order 3 in A under the action of the centralizer $C_A(g)$; that is, we consider two elements h_1 and h_2 of order 3 to be equivalent if for some $a \in C_A(g)$ we have $a^{-1}h_1a = h_2$.
3. For elements g of order 5 and h of order 3 found in the previous steps check whether gh has order 2. If so, the group $\langle\, g, h \,\rangle$ is isomorphic to $\mathcal{I}$.
4. For each group $H = \langle\, g, h \,\rangle$ found in the previous step, check that it fixes at most one point and check that it is geometric by determining whether the stabilizer of a representative for each of its orbits is of an allowed type. In the case of $\mathcal{I}$ the allowed subgroups are those of orders 1, 2, 3 and 5. This excludes the subgroups of orders 4 and 6, 10 and 12.
5. Carry out some additional checking to choose a single representative for groups that are conjugate within Aut(G).

3.2 Experimental Results

We implement the algorithm using `Magma` Version 2.8 [11], and conduct two types of experiments. The aim of the experiments is to test the runtime of the symmetry finding algorithm described above. The runtime, in general, depends more on the size of $\mathrm{Aut}(G)$ than the size of G, and thus we use test data with large automorphism groups.

The first experiment is to find all the 2- and 3-geometric groups for a given graph and then find a geometric automorphism group of maximum size. We use three test data sets. The first two test sets are graphs generated from permutation groups. For a permutation group H acting on a set V we obtain a graph $G = (V, E)$ by taking the edge set E to be an orbit of H on unordered pairs of vertices. That is, choose distinct elements u and v of V and set $E = \{\, \{h(u), h(v)\} \mid h \in H \,\}$. The graph created from H has an automorphism group that contains H. For each group, we choose at most one graph. We only choose graphs that are connected since disconnected graphs can be handled using similar methods in [7]. The groups were taken from a `Magma` database of primitive permutation groups of degree (that is, $|V|$) less than 50. The first test set has 67 graphs with $|\,\mathrm{Aut}(G)| < 1,000$ and a second test set has 34 graphs with $1,000 < |\,\mathrm{Aut}(G)| <$

$51,000,000$. A third test data was a set of graphs with highly symmetric graphs such as regular graphs, the cage graphs, non-Hamiltonian graphs, the platonic solids and incidence planes from *Groups & Graphs* [9].

The first experiment was done on DEC Alpha 600 5/333. Tables 1 and 2 display the experimental results from the first experiment with three data sets. First we present the time for Computing $\mathrm{Aut}(G)$ and the size of $\mathrm{Aut}(G)$. Each row shows the time for finding each of 2- and 3-geometric groups. For each case, we present the average and worst case of time, the average and maximum size of the maximal subgroup.

Table 1. Results for 100 graphs

	$\lvert Aut(G)\rvert < 1,000$				$\lvert Aut(G)\rvert > 1,000$			
	Time		Size		Time		Size	
	Av	Max	Av	Max	Av	Max	Av	Max
$\mathrm{Aut}(G)$	0.00152	0.017	238	820	0.0042	0.017	1812375	50803200
$\mathcal{C}_k^2$	0.017	0.1	23.1	47	0.11	0.634	8.9	24
$\mathcal{D}_k^2$	0.0057	0.034	46.1	94	0.29	3.8	16.1	24
Av for 2D	0.022				0.41			
$\mathcal{C}_k^3$	0.017	0.1	23.1	47	0.11	0.63	8.9	24
$(\mathcal{C}_{2k}\lvert\mathcal{C}_k^3)$	0.005	0.017	5.3	20	0.0058	0.033	7.3	24
$\mathcal{C}_k^{3*}$	0.002	0.0125	6	6	0.34	5.0	8.5	12
$\mathcal{D}_k^3$	0.002	0.016	46.1	94	0.35	5.0	16.1	24
$(\mathcal{D}_k\lvert\mathcal{C}_k^3)$	0.002	0.0202	46.1	94	0.373	5.26	16.1	24
$(\mathcal{D}_{2k}\lvert\mathcal{D}_k^3)$	0.002	0.025	46.1	94	0.362	5.1	16.1	24
$\mathcal{D}_k^{3*}$	0.005	0.033	8	8	0.678	8.7	15.7	28
$\mathcal{T}$	0.0002	0.016	12	12	0.23	2.9	12	12
$\mathcal{T}^*$	0.0003	0.017	0	0	0.0039	0.067	24	24
$(\mathcal{O}\lvert\mathcal{T})$	2e-5	0.0005	24	24	0.033	0.558	24	24
$\mathcal{O}$	2e-5	0.0005	0	0	0.033	0.558	24	24
$\mathcal{O}^*$	5e-5	0.001	0	0	0.0029	0.05	48	48
$\mathcal{I}$	0	0	0	0	0.048	0.617	60	60
$\mathcal{I}^*$	0	0	0	0	0	0	120	120
Av for 3D	0.037				2.58			

Experimental results shows that our method is very efficient. In practice, it computes all 2- and 3-geometric automorphism groups very quickly. For example, for the first data set ($|V| \leq 50$ with $|\,\mathrm{Aut}(G)| < 1,000$), it takes 0.022 seconds to compute all 2-geometric groups and 0.037 seconds to compute all 3-geometric groups on average. When $|\,\mathrm{Aut}(G)|$ becomes larger, then it takes longer. For example, for the second data set ($|V| \leq 50$ with $1,000 < |\,\mathrm{Aut}(G)| < 51,000,000$), it takes 0.41 seconds to compute all 2-geometric groups and 2.58 seconds to compute all 3-geometric groups on average. In fact, the computation of $\mathrm{Aut}(G)$ is very fast using `nauty` [12] inside of `Magma`. Most of the runtime is for looking

Table 2. Results for highly symmetric graphs

	Time		Frequency	Max Size Found	# Non-Conjugate Groups
	Av	Max	Total	Av	Av
Aut(G)	0.3125	6.069	33	51403.0909	
$\mathcal{C}_k^2$	1.2103	24.05	30	8.6333	3.9666
$\mathcal{D}_k^2$	0.1125	2.859	28	17.3571	9.7857
$\mathcal{C}_k^3$	1.2187	24.14	33	8.3030	7.4545
$(\mathcal{C}_{2k}\|\mathcal{C}_k^3)$	0.0051	0.111	26	7	4.8461
$\mathcal{C}_k^{3*}$	0.1911	2.5675	20	8.4	2.8
$\mathcal{D}_k^3$	0.2558	4.6785	29	17.2413	13.7241
$(\mathcal{D}_k\|\mathcal{C}_k^3)$	0.3235	5.8675	31	16.3870	29.2580
$(\mathcal{D}_{2k}\|\mathcal{D}_k^3)$	0.2489	3.8965	30	18.1333	6.3333
$\mathcal{D}_k^{3*}$	0.3276	6.56	10	17.6	6.2
$\mathcal{T}$	0.1721	4.039	14	12	1
$\mathcal{T}^*$	0.0032	0.021	9	24	1.1111
$(\mathcal{O}\|\mathcal{T})$	0.0006	0.0055	9	24	1.3333
$\mathcal{O}$	0.0006	0.0055	6	24	1.5
$\mathcal{O}^*$	0.0026	0.051	5	48	1.2
$\mathcal{I}$	0.0051	0.15	4	60	1
$\mathcal{I}^*$	0.0015	0.03	4	120	1
Av Total for 3D	2.7568				

for the subgroups within Aut(G). This is the reason that the runtime depends more on the size of Aut(G) than the size of G in general.

The second experiment is to find a rotational symmetry of maximum order or a reflectional symmetry with the minimum number of fixed points. The aim of this experiment is to compare our method with a branch and cut method [2]. We use three test sets of [2] including the `rome` test suite with 11529 graphs [13]. The first set `aut` has 3000 graphs with $|V| < 30$, designed to have many automorphisms, but few of them are geometric. These are the so-called *hard* instances of [2]. The second set `sym` has 8000 graphs with $|V| < 80$, generated specifically to have rotational symmetries.

The second experiment was done on a standard laptop, DELL Latitude C600 (750MHz, 256 MB RAM). Tables 3 and 4 display the experimental results from the second experiment with three test sets. The result shows that in general our method is much faster than the branch and cut method [2]. For example, Table 3(b) displays the result for `aut`. It takes 0.015 seconds to find best symmetry on average and 2.60 seconds in worst case. Note that it takes 21.6 seconds on average and 9197.55 seconds in worst case using the method of [2] (see Table 3 in [2]). Table 3(a) displays the result for `sym`. It takes 0.016 seconds to find best symmetry on average and 0.30 seconds in worst case.

Table 4 displays the result for `rome data`. It takes 0.009 seconds to find best symmetry on average and 0.12 seconds in worst case. Note that it takes 4.26 seconds on average and 19.29 seconds in worst case using the method of [2].

Table 3. Results for (a) 8000 graphs in `sym` and (b) 3000 graphs in `aut`

(a) runtimes for `sym`, $1 \leq n \leq 80$

	Time (sec)	
n	Av	Max
1–10	0.0013	0.05
11–20	0.0038	0.21
21–30	0.0124	0.30
31–40	0.0140	0.24
41–50	0.0177	0.22
51–60	0.0219	0.25
61–70	0.0285	0.16
71–80	0.0351	0.27

(b) runtimes for `aut`, hard instances

	Time (sec)	
n	Av	Max
1–5	0.0008	0.01
6–10	0.0027	0.05
11–15	0.0056	0.22
16–20	0.0115	0.70
21–25	0.0362	1.58
26–30	0.0356	2.60

Table 4. Results for all 11529 graphs in `rome`

	Time			Size		
	Av	Max	Graph	Av	Max	Graph
Aut(G)	0.00102125	0.011	grafo10316.100.lgr	5.0617	1920	grafo8507.75.lgr
BestCycle	0.00909117	0.12	grafo5890.48.lgr	1.6825	3	grafo206.12.lgr

4 Displaying a Geometric Automorphism Group

4.1 Choosing a Representation

It is possible to construct different drawings which display a given geometric automorphism group H, depending on the choice of the representation. Let $H \subseteq \mathrm{Aut}(G)$ be a 2- or 3-geometric group. We now describe how the choice of representation effects the drawing.

We may consider only representations of H with a fixed image $T \subseteq O_2(\mathbb{R})$ or $O_3(\mathbb{R})$. Consider two different faithful representations $\phi, \theta : H \to T$. Then $\phi\theta^{-1}$ is an automorphism of H. Hence choosing a different representation is equivalent to composing a fixed representation with an automorphism of H.

Let G be a cycle of length 5 with a cyclic subgroup $H = \langle\, (12345)) \,\rangle = \langle\, p \,\rangle$. Figure 3 shows four different drawings that display H. Each uses a representation that takes a generator of H (p, p^2, p^3 or p^4 respectively) to a rotation by $2\pi/5$. Note that Figure 3 1a) and 1d), 1b) and 1c) are the same up to relabeling. This is because p and p^4, and p^2 and p^3 are conjugate in $\mathrm{Aut}(G)$ by $(25)(34)$. Furthermore p and p^2 are conjugate by $(2345) \in S_5 \setminus \mathrm{Aut}(G)$ thus the drawings that display them use the same points for vertices with different edges.

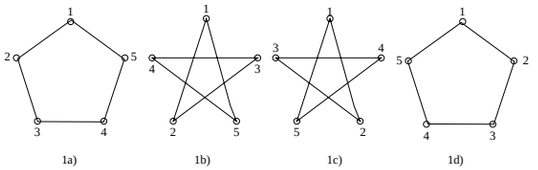

Fig. 3. Using different representations to display the same group

4.2 Symmetric Drawing Algorithm

A simple drawing method for two dimensions is given by [3] where the main idea is to draw each orbit in a circle. We extend this method to three dimensions.

`Algorithm to display a 3-geometric group`
Let $v_1, v_2, \ldots, v_r$ be representatives for the orbits of H acting on the vertex set V of the graph. For each i:

1. Find the stabilizer H_{v_i} of v_i.
2. If $H_{v_i} = H$, define $D(v_i)$ to be the origin.
3. If the dimension of $\phi(H_{v_i})$ is one, define $D(v_i)$ to be a vector of length i fixed by $\phi(H_{v_i})$; that is, an eigenvector of length i for the eigenvalue 1.
4. If the dimension of $\phi(H_{v_i})$ is two, let r be the reflection generating $\phi(H_{v_i})$. Choose $D(v_i)$ to be a vector of length i fixed by r and not fixed by any other element in a subgroup of $\phi(H)$ containing r.
5. If H_{v_i} is the trivial group, define $D(v)$ to be any vector of length i not in the fixed point space of any non-trivial element of $\phi(H)$. (These spaces of fixed points are known in advance and only need to be computed once.)
6. For v in the orbit of v_i, choose $h \in H$ such that $v = hv_i$ and define $D(v)$ to be $\phi(h)D(v_i)$.

Note that the points $D(v)$ are not uniquely determined. At step 3, the orbit can be placed at a different radius and at steps 4 and 5 there is considerable choice for the selected point. The drawing algorithm has been implemented using `magma`, java and `jjgraph`. For smaple outputs, see Figure 4.

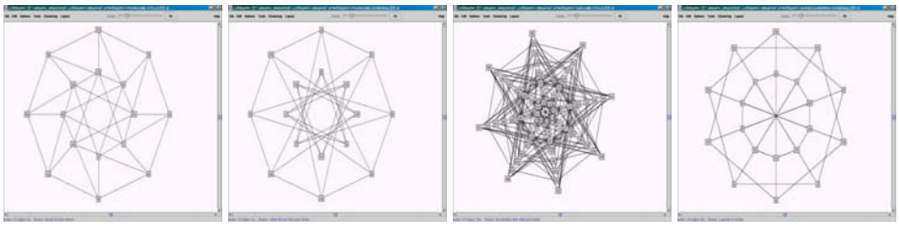

Fig. 4. Four-cube displaying (a) cyclic and (b) dihedral symmetry. (c) six-cube displaying cyclic symmetry (d) the dodecahedron displaying dihedral symmetry.

5 Conclusion

In this paper, we present a group-theoretic method to find *all* 2- and 3-geometric automorphism groups of a graph. We implement the method using `Magma` and the experimental results show that our approach is very efficient in practice. We also present a simple drawing algorithm to display 2- or 3-geometric automorphism groups.

To construct a maximally symmetric drawing of a graph, we need to choose a geometric automorphism group of a maximum size. In fact it is possible to construct different symmetric drawings of a graph, depending on the choice of the 2- or 3-geometric groups, the representation of a given geometric automorphism group, and ordering of the orbits with given representation.

References

1. D. Abelson, S. Hong and D. E. Taylor, A Group-Theoretic Method for Drawing Graphs Symmetrically, Technical Report IT-IVG-2002-01, School of Information Technologies, The University of Sydney, 2002.
2. C. Buchheim and M. Jünger, Detecting Symmetries by Branch and Cut, *Graph Drawing 2001*, Lecture Notes in Computer Science LNCS 2265, pp. 178-188, Springer Verlag, 2002.
3. P. Eades and X. Lin, Spring Algorithms and Symmetries, *Theoretical Computer Science*, 240, pp. 379-405, 2000.
4. H. Fraysseix, An Heuristic for Graph Symmetry Detection, *Graph Drawing'99,* Lecture Notes in Computer Science 1731, pp. 276-285, Springer Verlag, 1999.
5. S. Hong, P. Eades and S. Lee, An Algorithm for Finding Geometric Automorphisms in Planar Graphs, *Algorithms and Computation*, Lecture Notes in Computer Science 1533, pp. 277-286, Springer Verlag, 1998.
6. S. Hong, Drawing Graphs Symmetrically in Three Dimensions, *Graph Drawing 2001*, Lecture Notes in Computer Science LNCS 2265, pp. 189-204, Springer Verlag, 2002.
7. S. Hong and P. Eades, Drawing Planar Graphs Symmetrically IV: Disconnected Graphs, Technical Report CS-IVG-2001-03, Basser Department of Computer Science, The University of Sydney, 2001.
8. J. Manning, *Geometric Symmetry in Graphs*, Ph.D. Thesis, Purdue Univ., 1990.
9. Groups & Graphs, http://130.179.24.217/G&G/G&G.html.
10. W. Ledermann, *Introduction to Group Theory*, Longman, 1973.
11. `Magma`, http://magma.maths.usyd.edu.au.
12. `nauty`, http://cs.anu.edu.au/∼bdm/nauty.
13. Rome test suite ALF/CU, http://www.dia.uniroma3.it/∼gdt.