Construction of Combinatorial Objects

R. Laue, Universität Bayreuth
Lehrstuhl II für Mathematik (Informatik)

preliminary version 6.4.97 , extension of [44]

Abstract:

Isomorphism problems often can be solved by determining orbits of a group acting on the set of all objects to be classified. The paper centers around algorithms for this topic and shows how to base them on the same idea, the homomorphism principle. Especially it is shown that forming Sims chains, using an algorithmic version of Burnside's table of marks, computing double coset representatives, and computing Sylow subgroups of automorphism groups can be explained in this way. The exposition is based on graph theoretic concepts to give an easy explanation of data structures for group actions.

Contents

1 A General Point of View

A natural goal in mathematical theories is a full description of the objects that are investigated. This goal has been successfully achieved in some cases, for example all finite abelian groups and with much more effort all finite simple groups.

More often one restricted the research activity firstly to more modest problems like the pure existence of any object with some prescribed properties, for example in the case of block-designs or even when solutions for some optimization problem are considered. A step further was taken in combinatorics where the number of objects was determined by ingenious methods without any relation to a direct construction idea. For example the famous Pólya-De Bruijn method of counting orbits of a group acting on sets of mappings allowed to determine exact or approximate numbers of different types of graphs, e.g. especially from a more practical point of view, the number of different chemical isomers of a certain type [58, 11, 33].
Now that scientists have more and more powerful computers available in some nontrivial cases the construction problem itself can be attacked successfully. Thus, some of the above mentioned finite simple groups have been constructed by the help of a computer, where the theoretical approach gave very restrictive necessary conditions for the existence of some sporadic simple group. With respect to the mathematical description of chemical structures to a certain extent the above mentioned counting approach could be replaced by a construction process.
The present paper aims at an introduction into some general methods for *construction algorithms*. Concrete results for special types of objects will serve as an example and hopefully give an impact for further applications in different areas.
We restrict our attention to isomorphism problems. More formally we assume that a group is acting on a set of objects such that *the orbits* are just the *isomorphism classes* of objects. The stabilizer of one object then acts as a group of automorphisms on that object. The problems considered are

(1) decide whether two given objects are isomorphic, i.e. lie in the same orbit of the group,

(2) give a full set of representatives for the isomorphism types, i.e. a transversal from the set of orbits.

(3) determine the stabilizer of a given object.

The method consists in exploiting structure theories for algorithmic purposes, basing different algorithms upon the same theoretical foundation.

Let G be a finite group, acting on some finite set Ω. Then the problem of determining the set

$$\Omega/G = \{\omega^G \mid \omega \in \Omega\}$$

of all orbits $\omega^G = \{\omega^g \mid g \in G\}$ can be transformed into graph theory. For that purpose we assume that G is given by a set of generators S. We ask for algorithms determining a set Γ of representatives for Ω/G, the set of orbits $\omega^G = \{\omega^g \mid g \in G\}$ of G on Ω, by

- determining a function

$$f\colon \Omega \to G,$$

which for each $\omega \in \Omega$ computes some $f(\omega) = g \in G$ (as a word over S) such that $\omega^g \in \Gamma$,

- determining for each $\omega \in \Omega$ a set of generators for $N_G(\omega) = \{g \in G \mid \omega^g = \omega\}$, the stabilizer of ω in G.

The link to graph theory is the following definition.

1.1 Definition. *Let G be a finite group acting on a set Ω. Let G be generated by $S = \{g_1, \ldots, g_d\}$ for some $d \in \mathbf{N}$. Then the Cayley Action Graph*[1]*,* **CAG**, *for (Ω, S) is a graph with vertex set Ω and edges (ω, ω^g) for all $g \in S$ and $\omega \in \Omega$. Each edge (ω, ω^g) is labeled by g.*

A Cayley Action Graph is a directed graph. But since each group element g forms cycles on the set of points, a point ω_1 is reachable from a point ω_2 if and only if ω_2 is reachable from ω_1 via appropriate paths. Thus, there is no distinction between weakly and strongly connected components of a Cayley Action Graph, and we therefore speak of connected components.

The Cayley Action Graph is a generalization of the Cayley Graph where a group G acts on itself by multiplication from the right. While in a Cayley Graph by the group axioms we have only one connected component a Cayley Action Graph may have several connected components. Since each $g \in G$ can be expressed as a word $g = g_{i_1} g_{i_2} \ldots g_{i_r}$ with $g_{i_j} \in S$, we have $\omega^g = (\ldots(\omega^{g_{i_1}})^{g_{i_2} \cdots})^{g_{i_r}}$. Thus, ω' lies in the same orbit as ω if and only if both lie in the same connected component of the Cayley Action Graph. The two problems (1) and (2) stated above can now be solved by graph theoretic algorithms. A standard approach for computing connected components of a graph solves the orbit problem.

1.2 Proposition. *Ω/G can be determined in time and space*

$$0(|\Omega| \cdot d)$$

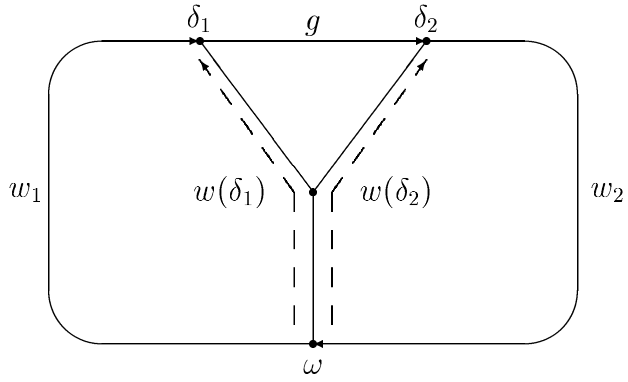
Proof. See, for example, [30, p. 287], and notice that each g_i defines up to $|\Omega|$ arcs of the graph.

□

The method consists in selecting an unvisited node as a representative and constructing by breadth-first search (or depth-first search) a *spanning tree* for the connected component containing the selected node until all nodes are visited. The result is a rooted spanning forest where the roots are the chosen representatives. We now deduce some properties from such fixed spanning trees. For $\delta_1, \delta_2 \in \Omega$ the spanning trees yield paths labeled by words w_1, w_2 such that $\delta_1^{w_1} = \omega_1$ and $\delta_2^{w_2} = \omega_2$ are the corresponding roots.

[1] Termed after a suggestion of G. Butler.

Figure 1: Factorization of w.



The points δ_1 and δ_2 lie in the same orbit iff $\omega_1 = \omega_2$ and then $w_1 w_2^{-1}$ transforms δ_1 into δ_2.

It is well known and easily proved that the right cosets of the *stabilizer* $N_G(\omega)$ of one point ω in G are in one-one correspondence to the points in the orbit of ω. In terms of the spanning tree the words labeling paths of the tree from a root ω to the other points in the connected component of ω represent these cosets of $N_G(\omega)$ in G.

Any element of $N_G(\omega)$ can be written as a word

$$w = g_{i_1} \dots g_{i_r}, \quad g_{i_j} \in S,$$

which labels a path starting at ω and returning to ω. Since each point δ in that circle also belongs to the chosen spanning tree, there exists already a word $w(\delta)$ labeling a tree path from ω to δ.

Therefore if $w = w_1 g w_2$ for some $g \in S$ and $\omega^{w_1} = \delta_1$, $\delta_1^g = \delta_2$ then we have a factorization $w = w_1 w(\delta_1)^{-1} w(\delta_1) g w(\delta_2)^{-1} w(\delta_2) w_2$. Here $w_1 w(\delta_1)^{-1}$, $w(\delta_1) g w(\delta_2)^{-1}$, and $w(\delta_2) w_2$ label circles with ω as starting and end point (cf. Fig. 1).
The path label word $w(\delta_1) g w(\delta_2)^{-1}$ is called a **Schreier Generator** of $N_G(\omega)$. Since we can apply this factorization process to the subwords $w_1 w(\delta_1)^{-1}$ and $w(\delta_2) w_2$ recursively, we obtain the

1.3 Theorem (Schreier). *$N_G(\omega)$ is generated by Schreier generators.*

For a proof see also Lemma 6.2.2 of [29] or Lemma 7.1.2 of [33].

Thus, a set of generators for $N_G(\omega)$ can be obtained from the spanning tree with root ω by all elements $w_1 g w_2^{-1}$ where w_1 and w_2 are path labels of the tree and $g \in S$ such that $\omega^{w_1 g} = \omega^{w_2}$.

Note that we have to apply all generators $g \in S$ to all points in the orbit of ω to obtain all Schreier generators. As basic tools for handling stabilizers and orbits we now have the set of words $w(\delta)$ mapping δ onto the orbit representative and Schreier generators for the stabilizers of the representatives.

We proposed above to use breadth-first search in the CAG to build a spanning tree. This has the effect that a vertex which can be reached from a vertex already in the tree by the generator currently considered will be added immediately to the tree. In a depth-first search this vertex might be reached by some longer path lateron. So the depth of the spanning tree should become smaller using breadth-first search. Consequently the word-length of coset representatives of the stabilizer $N_G(\omega)$ in G written as words in the given set of generators according to the path labels of the spanning tree will be short.
A method, called *double the cube*, allows to find a spanning tree with an explicit upper bound for the depth of the tree [17]. Cubes C_i are constructed iteratively. The starting point is a cube $C_1 = \{id, g_1\}$ for some generator g_1 which does not fix ω. In an iteration step C_i is constructed from C_{i-1}. One has to find first an element $g \in S$ and a point $\gamma \in \omega^{C_{i-1}^{-1}C_{i-1}}$ such that γ^g is not contained in the set of points $\omega^{C_{i-1}^{-1}C_{i-1}}$. If no such g exists then the orbit of ω is already completely determined. Otherwise we have some element $h \in C_{i-1}^{-1}C_{i-1}$ such that $\omega^h = \gamma$. We define $g_i = h \cdot g$, apply g_i to all points found so far, and add the corresponding edges to new points to the spanning tree. Then $C_i = C_{i-1} \cup C_{i-1}g_i$.
The bound for the depth of the tree comes from the fact that $|C_i| = 2^i$ if $C_i \neq C_{i-1}$. This is clear for $i = 1$ and we assume it to be true for $i-1$ where $i > 1$. By the choice of g_i we know that $g_i \notin C_{i-1}^{-1}C_{i-1}$ such that no product $c \cdot g_i$ can lie in C_{i-1} for any $c \in C_{i-1}$. Thus the number of elements in C_i doubles in each iteration step. Since obviously $C_i \subseteq G$, we have an upper bound $log_2|G|$ for i. By a careful choice of a datastructure for the sets $\omega^{C_{i-1}^{-1}C_{i-1}}$ one obtains the following result.

1.4 Theorem. *A spanning tree for a CAG has an upper bound $2log_2|G|$ for the depth of the tree if it is constructed by the strategy of doubling the cube.*

Unfortunately the graph theoretic algorithms are not sufficient to solve our construction problems, since even linear time and space requirements are not tolerable. Let us have a look at a typical example.

Consider the set of multigraphs with set of vertices $X = \{1, 2, \ldots, n\}$ and edge multiplicities up to $k, k \in \mathbf{N}$. Each multigraph of this kind can be described by a mapping $f\colon \binom{X}{2} \to \{0, \ldots, k\}$, assigning to each two-element subset $\{x_1, x_2\} \subseteq X$ an edge multiplicity, i.e. either 0 or some $i,\ 1 \leq i \leq k$.
Two multigraphs described by mappings f_1, f_2 are isomorphic if and only if there is some permutation $\pi \in S_X$, the symmetric group on X, such that $f_1^\pi = f_2$, where

$$f_1^\pi(\{x_1, x_2\}) = f_1(\{x_1^{\pi^{-1}}, x_2^{\pi^{-1}}\})$$

for all $\{x_1, x_2\} \in \binom{X}{2}$. Thus, the orbits of S_X acting on the set of mappings

$$\Omega = \{0, \ldots, k\}^{\binom{X}{2}}$$

are just the isomorphism types.

In many mathematical descriptions of multigraphs these are described by adjacency matrices. We describe shortly the relation of that representation of a multigraph to our description.
If the vertices are arranged in some fixed order, $X = (1, \ldots, n)$ for example, the $i-$th vertex may be used to label the $i-$th row and the $i-$th column of a matrix A. For a mapping $f\colon \binom{X}{2} \to \{0, \ldots, k\}$, describing a multigraph we define for each pair (i, j) of indizes

$$A(i, j) = f(\{i, j\}).$$

Then we obtain a symmetrical matrix, the adjacency matrix of f. This matrix can easily be used as a data-structure to represent f in a computer, though a listing of the neighbours of each vertex and the corresponding edge degrees may be less space and time consuming.
Any permutation $\pi \in S_X$ applied to the row and column labels will produce a new adjacency matrix A^π for a graph isomorphic to the graph described by f. Thus, S_X acts on the set of all adjacency matrices. $A^\pi = A$ is equivalent to $f^\pi \cong f$, such that the automorphisms of f form the stabilizer in S_X of A and the right cosets of Aut$f$ in S_X correspond bijectively to the different adjacency matrices of multigraphs isomorphic to f.

Now let us return to the problem of finding representatives for all isomorphism types of multigraphs for fixed k and n. Here the above described solution of the orbit problem would require $0(k^{\binom{n}{2}})$ space and time, which is infeasible. To solve this problem, we have to exploit the special kind of this group action. Since similar problems occur with other discrete structures, a general method which applies to many kinds of group actions is in order.

In mathematics structure theories often rely on the common concept of homomorphisms. A homomorphic image is a coarse version of the preimage structure and a sequence of successive homomorphisms describes a series of gradually coarser versions of the original structure. Going into the opposite direction then yields a stepwise approximation of the structure in question. This point of view shows that *mathematical structure theory is a well developed mechanism for formalizing the top-down and divide-and-conquer paradigms of computer science.* Homomorphisms can well be used to develop powerful algorithms. We formalize this observation for our problem of group actions.

1.5 Definition. (Homomorphism of group actions) *Let G_1 be a group acting on a set Ω_1 and G_2 be a group acting on a set Ω_2. A pair $\sigma = (\sigma_\Omega, \sigma_G)$ of mappings, where*

σ_Ω maps Ω_1 into Ω_2 and $\sigma_G : G_1 \to G_2$ is a group homomorphism, is a homomorphism of group actions if σ is compatible with both actions, i.e. for all $g \in G_1$ and all $\omega \in \Omega_1$

$$(\omega^g)^{\sigma_\Omega} = \omega^{\sigma_\Omega g^{\sigma_G}}.$$

If both components of σ are surjective σ is an epimorphism, if both components are bijective σ is an isomorphism.

If two group actions $(\Omega_1, G), (\Omega_2, G)$ are **isomorphic** with an isomorphism σ then σ carries orbit representatives and their stabilizers onto corresponding representatives and stabilizers. Thus, if isomorphisms of group actions can be found whole sets of solutions of the orbit problem can be used many times. This can be used to describe large sets of solutions implicitely while still allowing an explicit listing on demand.

Often we only need one group acting on different sets. We therefore specialize to this situation now. Let G be a group acting on two sets Ω_1 and Ω_2. A mapping $\varphi\colon \Omega_1 \to \Omega_2$ is a homomorphism with respect to G if for each $g \in G$ and $\omega \in \Omega_1$

$$\varphi(\omega^g) = \varphi(\omega)^g,$$

i.e. φ is compatible with the group action.

We remark that a homomorphism φ induces a homomorphism of the Cayley Action Graph for G acting on Ω_1 to the Cayley Action Graph for G acting on Ω_2 (see Fig. 2). Any such homomorphism can be exploited for our algorithmic purpose:

1.6 Lemma. *Let $\varphi\colon \Omega_1 \to \Omega_2$ be a homomorphism with respect to a group G acting on Ω_1 and Ω_2.*
Then $\omega, \omega' \in \Omega_1$ lie in the same $G-$orbit if and only if

(1) there exists some $g_1 \in G$ s.t. $\varphi(\omega)^{g_1} = \varphi(\omega')$ and

(2) there exists some $g_2 \in N_G(\varphi(\omega))$ s.t. $\omega^{g_2} = \omega'^{g_1^{-1}}$.

Proof. Let $\omega^g = \omega'$ for some $g \in G$. Then $\varphi(\omega)^g = \varphi(\omega^g) = \varphi(\omega')$. So let $\varphi(\omega)^{g_1} = \varphi(\omega')$ and $\omega^g = \omega'$. Then $\varphi(\omega) = \varphi(\omega')^{g_1^{-1}} = \varphi(\omega^g)^{g_1^{-1}} = \varphi(\omega)^{gg_1^{-1}}$ and $gg_1^{-1} \in N_G(\varphi(\omega))$ and so $g = g_2 g_1$ for some $g_2 \in N_G(\varphi(\omega))$. Then $\omega' = \omega^g = \omega^{g_2 g_1}$ implies $\omega^{g_2} = \omega'^{g_1^{-1}}$.

□

If we have to decide whether two given objects $\omega, \omega' \in \Omega_1$ are in the same orbit and, if so, find an element $g \in G$ mapping the first object to the other can thus be done stepwise. For a first step it suffices to find an element g_1 such that $\varphi(\omega)^{g_1} = \varphi(\omega')$. If no such element exists then the objects are in different orbits. If we have found such a g_1 we replace ω' by ω'^{g_1} and now only have to solve the smaller problem to decide whether ω, ω'^{g_1} are in the same orbit under $N_G(\varphi(\omega))$. This situation appears for example when we want to decide whether two p-groups or two graphs are isomorphic. In the case of a group one can map onto a factor group by a characteristic subgroup in order to simplify the problem. In the case of graphs G. Luks peeled away a set of vertices or edges which is determined by an artificially distinguished vertex. Similar strategies apply in many situations. It should be remarked that the use of invariants is subsumed by taking a trivial group action on the image set.

This observation also allows to factorize the problem of finding orbit representatives. We discuss the two situations where we want to determine orbit representatives firstly from Ω_1 and secondly from Ω_2. We assume that we have already appropriate algorithms for solving the problem in Ω_2 or Ω_1 at hand, respectively.

We use the lemma for our aims of determining a set Γ of representatives for Ω/G, a function $f\colon\Omega \to G$ such that $\omega^{f(\omega)} \in \Gamma$, and determining the stabilizer $N_G(\gamma)$ for each $\gamma \in \Gamma$. Firstly we show how to reduce the generally large problem to a set of comparatively small problems of the same type.

1.7 Algorithm (Splitting orbits). *Let $\varphi\colon\Omega_1 \to \Omega_2$ be a homomorphism with respect to a group G acting on Ω_1 and Ω_2.*
Let A be an algorithm computing $\varphi^{-1}(\omega)$ for each $\omega \in \Omega_2$.
Let B be an algorithm determining for $\Delta \subset \Omega_1$ closed under the action of $U \leq G$

(1) a set Γ_Δ of representatives for Δ/U,

(2) a function $f_\Delta\colon\Delta \to U$ such that $\delta^{f_\Delta(\delta)} \in \Gamma_\Delta$ for each $\delta \in \Delta$

(3) for each $\delta \in \Gamma_\Delta$ a set of generators for $N_U(\delta)$.

Let Γ_2 be a set of representatives for Ω_2/G and their respective stabilizers and let $f_2\colon\Omega_2 \to G$ such that $\omega^{f_2(\omega)} \in \Gamma_2$ for each $\omega \in \Omega_2$.

Method:

(1) Initialize Γ_1 as empty set.

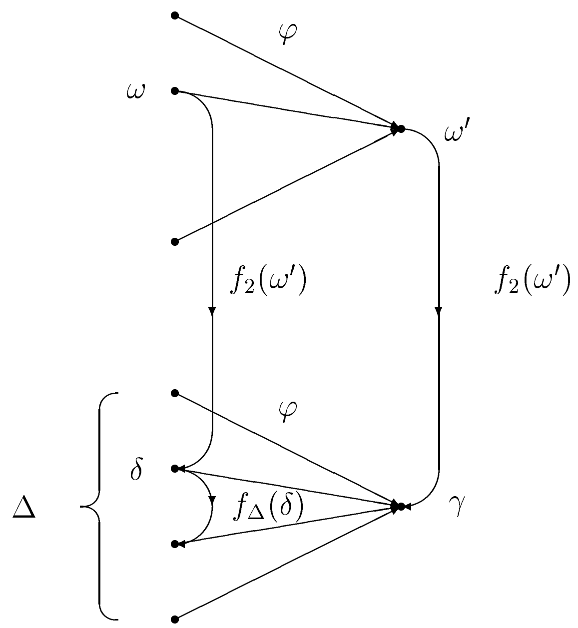
(2) For each $\gamma \in \Gamma_2$

  (a) Use A to compute $\varphi^{-1}(\gamma)$.
  (b) Use B to determine a set $\Gamma_1(\gamma)$ of representatives for $\varphi^{-1}(\gamma)/N_G(\gamma)$ and the corresponding stabilizers in $N_G(\gamma)$. These stabilizers are already the full stabilizers in G.
  (c) Add $\Gamma_1(\gamma)$ to Γ_1.

(3) We obtain $f_1\colon\Omega_1 \to G$ such that $\omega^{f_1(\omega)} \in \Gamma_1$ for each $\omega \in \Omega_1$ as follows:

  (a) Compute $\omega' = \varphi(\omega)$.
  (b) Compute $f_2(\omega')$.
  (c) Compute $\delta = \omega^{f_2(\omega')}$.

Figure 2: Homomorphism of Cayley Action Graphs



(d) *Compute* $\gamma = \omega'^{f_2(\omega')}$.

(e) *Compute* $f_\Delta(\delta)$ *for* $\Delta = \varphi^{-1}(\gamma)$ *using* B,

(f) *Set* $f_1(\omega) = f_2(\omega')f_\Delta(\delta)$.

$\bigtriangledown$

Of course one could apply algorithm B directly to $\Delta = \Omega$ and $U = G$ in 1.7. The decisive point is that the splitting technique reduces the sizes of Δ and U generally to a fraction of the original size, depending on φ. Correspondingly also the running time will become much shorter. A detailed analysis is given in the theorem below. In many situations most orbits of G are very long, such that the stabilizer of one point is small. If already $N_G(\omega) = \{id\}$ for some $\omega \in \Omega_2$ then all points in $\varphi^{-1}(\omega)$ lie in different orbits of G. In this case no further call to algorithm B is necessary.

1.8 Example. As a first illustration of this strategy we again consider the multigraphs from above. Here $G = S_X$ acts on

$$\Omega_1 = \{0, \dots, k\}^{\binom{X}{2}}.$$

If we forget about edge-multiplicities we obviously obtain a simpler problem. This is formalized by $\varphi\colon \{0,\ldots,k\}^{\binom{X}{2}} \to \{0,1\}^{\binom{X}{2}}$, where for each $f\colon \binom{X}{2} \to \{0,\ldots,k\}$

$$\varphi(f)(\{i,j\}) = \begin{cases} 1 & \text{if } f(\{i,j\}) > 0 \\ 0 & \text{if } f(\{i,j\}) = 0 \end{cases}$$

Obviously this φ is compatible with the group action such that the lemma applies.
So in the first step we have to find representatives for the isomorphism types of simple graphs, i.e. multigraphs with edge multiplicities 0 or 1.
For each simple graph f the preimages under φ are obtained by colouring the edges with multiplicities up to k. The next step then consists in finding the orbits of $Aut f$, which is just $N_{S_X}(f)$, on the set of all edge colourings. It is well known, that most simple graphs have a trivial automorphism group and in such a case all colourings represent different isomorphism types. $\diamond$

This approach may be refined by enlarging the highest multiplicity gradually. Then firstly multigraphs with edge multiplicity up to 2 are constructed as described. In the next step only edges of multiplicity 2 are coloured by multiplicity 2 or 3. Of course the inverse is a mapping φ which in general reduces the highest edge multiplicity which is allowed in that step by 1. This is always compatible with the group action such that each single construction step only considers the problem of assigning to the elements of some $T \subseteq \binom{X}{2}$ values from a colour set of cardinality 2 only.
This shows:

1.9 Proposition. *Algorithms which solve the problem of finding representatives of orbits of a group on a set of mappings with range $\{0,1\}$ and the corresponding stabilizers suffice to solve the problem of finding representatives of the isomorphism types of multigraphs and the corresponding stabilizers for any finite edge multiplicity.*

For example colouring the complete graph with edge multiplicity up to 2 is equivalent to the problem of finding all simple graphs.

1.10 Corollary. *For any fixed number of nodes only finitely many orbit problems have to be solved up to isomorphic group action for describing all multigraphs with arbitrarily high edge multiplicity.*

Of course the proposition gives no direct description of time or space complexities. This depends heavily on the number of solutions in each subproblem and the complexity of the unknown algorithm which is applied in the simple steps. So the best we can give is a *relative complexity* in such cases where the cardinalities of the subproblems can be bounded. But this is a point of view which fits well to object oriented programming, where the situation of using unknown algorithms simplifies solving many in some respect similar problems.

In algebraic settings a homomorphism has a kernel and all image points have equally many preimages each forming a coset of the kernel. So there are many natural situations where we have a common global bound for the cardinalities of sets of preimages.
We therefore give a general complexity result for such situations.

1.11 Theorem. *Let G act on $\Omega_0, \ldots, \Omega_n$ and let $b_i \in \mathbf{N}$, $\varphi_i : \Omega_i \to \Omega_{i-1}$ be surjective homomorphisms with respect to G, $|\varphi_i^{-1}(\omega)| \leq b_i$ for all $\omega \in \Omega_{i-1}$.*
Let computing $\varphi_i^{-1}(\omega)$ take time $\leq c_i$.
An algorithm P may compute for each $U \leq G$ and any set Δ on which U acts a set $R = rep(\Delta/U)$ of representatives from the $U-$orbits on Δ and $N_U(r)$ for each $r \in R$, in time $a(|\Delta|)$, where $a: \mathbf{N} \to \mathbf{N}$ is monotonously increasing.
Then computing some $rep(\Omega_i/G)$ stepweise for $i = 0, 1, \ldots, n$ takes time bounded by

$$a(|\Omega_0|) + \sum_{i=1}^{n} a(b_i) \cdot c_i \cdot |rep(\Omega_i/G)|.$$

If there are $b, c \in \mathbf{N}$ with $b_i \leq b, c_i \leq c$ for all i then the time is bounded by

$$a(|\Omega_0|) + a(b) \cdot c \cdot n \cdot |rep(\Omega_n/G)|$$

If moreover $|\varphi_i^{-1}(\omega)| = b, c_i = c$ in each case then

$$n = 0(log_b(|\Omega_n|)).$$

yielding the time bound

$$O(log_b(|\Omega_n|) \cdot |rep(\Omega_n/G)|)$$

Again the proof is easy. For the last statement we point out that for each i

$$|\Omega_i| = b \cdot |\Omega_{i-1}|$$

such that $|\Omega_n| = b^n \cdot |\Omega_0|$ and

$$n = log_b|\Omega_n| - log_b|\Omega_0|.$$

This theorem shows that under very regular conditions the application of the homomorphism principle reduces the time complexity logarithmically compared to the direct application of the unknown algorithm to the original situation.

In some situations the homomorphism principle can also be used in the opposite direction. In such a case orbit representatives are known for the $G-$orbits on Ω_1. For each representative ω we have a description of its stabilizer by some set of generators and for each point $\delta \in \omega^G$ a word $\omega(\delta)$ in the generators of the whole group can be determined mapping ω to δ. G also acts on a set Ω_2 and $\varphi: \Omega_1 \to \Omega_2$ is a surjective homomorphism with respect to G. We want to obtain a corresponding description of the $G-$orbits on Ω_2.

1.12 Algorithm (Fusing orbits). *Let $\varphi\colon \Omega_1 \to \Omega_2$ be a surjective homomorphism with respect to a group G acting on Ω_1 and Ω_2. Let A be an algorithm computing $\varphi^{-1}(\omega)$ for each $\omega \in \Omega_2$. Let Γ_1 be a set of representatives for Ω_1/G. Let B be an algorithm determining a function*

$$f_1\colon \Omega_1 \to G$$

such that

$$\omega^{f_1(\omega)} \in \Gamma_1$$

for each $\omega \in \Omega_1$.
Let N_1 be an algorithm computing for each $\gamma \in \Gamma_1$ a set of generators for $N_G(\gamma)$.
Then we obtain Γ_2 a set of representatives for Ω_2/G as follows:

(1) Initialize $\Gamma_2 = \emptyset$.

(2) Choose as set Δ of candidates for Γ_2 the set of all $\varphi(\gamma)$ for $\gamma \in \Gamma_1$.

(3) Until Δ is empty do
choose $\delta \in \Delta$ and insert δ into Γ_2,
remove δ from Δ,
compute $\varphi^{-1}(\delta)$
and for each $\eta \in \varphi^{-1}(\delta)$
compute $\eta^{f_1(\eta)}$ and
remove $\varphi(\eta^{f_1(\eta)})$ from Δ
define $f_2(\varphi(\eta^{f_1(\eta)})) := f_1(\eta)^{-1}$.

(4) return Γ_2.

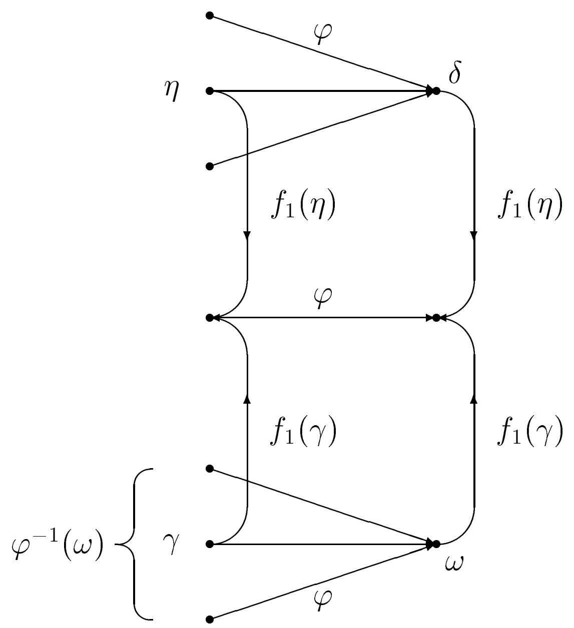
Now f_2 is already defined for the images of representatives from Γ_1.
We obtain $f_2\colon \Omega_2 \to G$ such that $\omega^{f_2(\omega)} \in \Gamma_2$ for each $\omega \in \Omega_2$ as follows:
For $\omega \in \Omega_2$ do:

compute $\varphi^{-1}(\omega)$ using algorithm A.
Choose some $\gamma \in \varphi^{-1}(\omega)$ and compute $f_1(\gamma)$.
Then $f_2(\varphi(\gamma^{f_1(\gamma)}))$ is already defined.
If $\varphi(\gamma^{f_1(\gamma)}) \in \Gamma_2$
then return $f_2(\omega) = f_1(\gamma)$,
else return $f_2(\omega) = f_1(\gamma) f_2(\varphi(\gamma^{f_1(\gamma)}))$.

We obtain N_2 computing for each $\gamma \in \Gamma_2$ a set of generators for $N_G(\gamma)$ as follows:

Compute $\Delta = \varphi^{-1}(\gamma)$ using algorithm A,
choose some $\delta \in \Delta$ which lies in Γ_1,
compute a set T of generators for $N_G(\delta)$ using algorithm N_1.
For each $\omega \in \Delta$ compute $\omega^{f_1(\omega)}$ using algorithm B.
If $\omega^{f_1(\omega)} = \delta$ then add $f_1(\omega)^{-1}$ to T.
return T.

Figure 3: Homomorphism of Cayley Action Graphs



∇

The proof of the correctness is easy and left as an exercise. We remark that in the computation of T the elements $f_1(\omega)^{-1}$ added to T form a full set of representatives for the right cosets of $N_G(\delta)$ in $N_G(\gamma)$. Thus, for a mere system of generators for $N_G(\gamma)$ many representatives will not be needed.

We have presented the algorithms without an explicit datastructure. Of course then it is hard to give a concrete complexity estimation. Instead we only can give relative complexities depending on the complexities of various unknown functions. The unknown functions must be supplied by some class of objects and deliberately their implementation should not be known to the outside of the class. Thus, we require certain predicates to be fulfilled by these functions and by the algorithm again produce as well functions fulfilling certain predicates.

Our group actions require a group G and a set Ω together with a possibility to apply some $g \in G$ to any $\omega \in \Omega$. If the group action is known by the action of some generators, for example using appropriate functions, then we can evaluate the image of any point ω under any word $g_1 g_2 \cdots g_l$ in these generators by computing $\omega_1 = \omega^{g_1}$ and iteratively $\omega_i = \omega_{i-1}^{g_i}$ for $i = 2, \cdots, l$. If the g_i are stored as permutations then this evaluation needs l table lookups. If we have an induced group action each computation of an ω_i in the iteration might need a larger amount of time, but the storing of a full permutation is avoided. Alternatively one might have a datastructure which allows to compute a group element g equal to the given word and then apply this to the point ω. The complexity might be totally different for all cases. But the higher level algorithms presented here could be used without knowledge of the actual datastructures in the lower levels. Of course this is just the object oriented approach, we only emphasize that this approach needs a handling of relative complexities which does not appear as self-understood in the literature as the popular object orientation might suggest. Moreover we not only have to give a complexity analysis for the algorithm computing orbit representatives but also for the functions created there we have to give an estimate for their run time.

Let us look at the fusion of orbits. There we find several prerequisites for the fusion process. Also there are not just sets of points in the result but also normalizers and functions determining for each given point in Ω_2 a group element mapping the point onto its orbit representative. The group elements are given as explicit products of elements given by input functions or inverses of such elements. One can return these products to the datastructure and leave it to the lower level to decide whether an explicit computation of the resulting group element is appropriate or the product is to be stored as a word. Thus we assume that applying some function f will take time $a(f)$.
Running through Γ_1 and computing $\phi(\gamma)$ needs $|\Gamma_1| \cdot a(\phi)$ time. We assume that selecting some candidate as the next representative to be processed takes constant time. Then we need $a(\phi^{-1}) + |\phi^{-1}(\delta)| \cdot (a(f_1)m + a(\phi) + a(invert)$ time for eliminating candidates and computing f_2 for these points. Here we denote by *invert* the function which inverts a group element.
Analogously the time for determining the stabilizers of the orbit representatives is bounded by $|\Gamma_2| \cdot a(\phi^{-1})(a(N_1) + |\phi^{-1}(\gamma)| \cdot a(f_1) \cdot a(invert))$.
The run time of f_2 will be $a(\phi^{-1}) + a(f_1) + a(\phi) + a(mult))$. Here we have assumed that determining the already known group elements $f_1(\gamma)$ and $f_2(\phi(\gamma^{f_1(\gamma)}))$ is included in the bound for multiplying these two elements by *mult*.

A typical application of 1.12 is a constructive version of the combinatorial principle of counting twice.

Suppose $\varphi_1 \colon \Omega \to \Omega_1$ and $\varphi_2 \colon \Omega \to \Omega_2$ are surjective homomorphisms with respect to a group acting on Ω, Ω_1 and Ω_2. Then one can factorize the problem of describing the action of G on Ω via Ω_1 and via Ω_2. Using first 1.7 for factorizing via Ω_1 and then 1.12 for fusing with φ_2 yields a description of the action of G on Ω_2. In some applications this indirect way is a powerful tool. We will demonstrate the principle by the problem of computing double coset representatives after [61].

2 Double cosets

We first apply the homomorphism principle to two well known group actions, multiplication of cosets by group elements and conjugation of subgroups.

Let $U \leq V \leq G$, and $B \leq G$. Then $\Omega_1 = U\backslash G = \{Ug \mid g \in G\}$ and $\Omega_2 = V\backslash G = \{Vg \mid g \in G\}$ are two sets on which B acts by multiplication from the right.

$$\varphi\colon \Omega_1 \to \Omega_2\colon Ug \mapsto Vg$$

is a homomorphism with respect to the action of B. If $V = \cup_{i=1}^{r} Ux_i$ then $\varphi^{-1}(Vg) = \{Ux_ig \mid 1 \leq i \leq r\}$. So provided $\{x_1, \ldots, x_r\}$ is available we can use 1.7 or 1.12 to factorize or fuse orbits. In this way we can step up and down through the subgroup lattice of a group and determine which elements lie in the same $B-$orbit, determine complete sets of representatives for the $B-$orbits and compute the stabilizers in B of points Ug.

These orbits are of special importance, since any $UgB = \cup\{Ugb \mid b \in B\}$ is a double coset of U and B in G. The stabilizer $N_B(Ug)$ is $g^{-1}Ug \cap B$, an intersection of subgroups. Thus, computing stabilizers can be used to determine intersections of subgroups, see [45]. Here we are more interested in double cosets.

2.1 The Split Lemma. *Assume a group B is a subgroup of a group G. The group G act transitively on a set Ω and ω be an arbitrarily chosen point of Ω. Then the orbits of B on Ω correspond bijectively to the double cosets $N_G(\omega)\backslash G/B$ by the mapping $\omega^{gB} \mapsto N_G(\omega)gB$.*

If G acts on any set Ω and $\omega \in \Omega$ then $\varphi\colon N_G(\omega)\backslash G \to \omega^G\colon N_G(\omega)g \mapsto \omega^g$ is a bijection. Moreover B as a subgroup of G also acts on ω^G and on $N_G(\omega)\backslash G$ by multiplication from the right. As φ is a homomorphism for B, determining the $B-$ *orbits* on ω^G is equivalent to determining the double cosets $N_G(\omega)\backslash G/B$.

A slightly different interpretation is that the double cosets correspond to those $B-$orbits that fuse to one $G-$orbit.

As an example consider a group B acting on a set Ω and ask for the orbits of B in the induced action on $\binom{\Omega}{k} = \{T \mid T \subseteq \Omega, |T| = k\}$ for some $k \in \mathbf{N}, 1 < k < |\Omega|$. If G is the symmetric group on $\binom{\Omega}{k}$ then G acts transitively such that $\binom{\Omega}{k} = T^G$ for any choice of $T \in \binom{\Omega}{k}$.

Our analysis tells that

$$N_G(T)\backslash G/B \to \binom{\Omega}{k}/B\colon\ N_G(T)gB \mapsto T^{gB}$$

is a bijection.

If $k \neq \frac{|\Omega|}{2}$ then $N_G(T)$ is a maximal subgroup of G such that 1.7 gives no reduction of complexity. But we can choose some $\omega \in T$ and obtain

$$U = N_G(T) \cap N_G(\omega) = N_{N_G(T)}(\omega),$$

a subgroup of index k in $N_G(T)$.

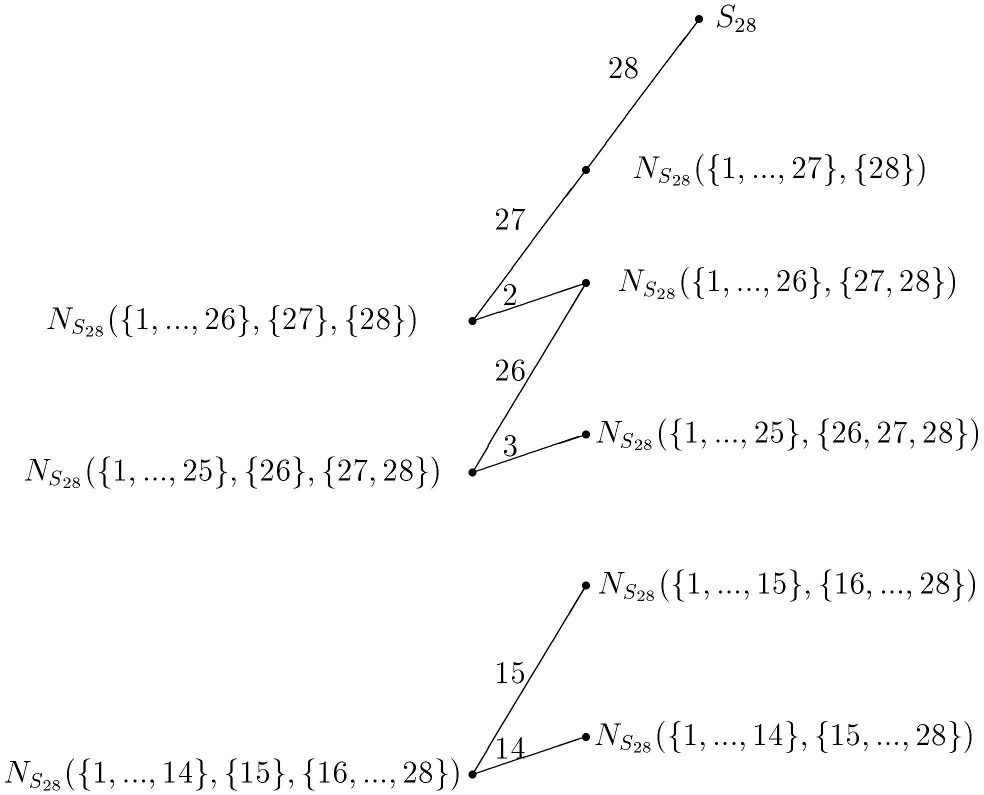


Figure 4:

This U is also contained in $N_G(T-\{\omega\})$ with index $k-1$. Thus, the double cosets $N_G(T)\backslash G/B$ can be computed from the double cosets $N_G(T-\{\omega\})\backslash G/B$ via U as described above.

Since $T-\{\omega\}\in\binom{\Omega}{k-1}$, an obvious iteration leads to $(T_1,T_2,\ldots,T_k)$, where $|T_i|=i$ and $T_i\subset T_{i+1}$, and a series of steps in which $N_G(T_{i+1})\backslash G/B$ can be determined from $N_G(T_i)\backslash G/B$ for $i=1,\ldots,k-1$.

This has been implemented by B. Schmalz [61] and for different data structures representing the acting group S. Weinrich [71].

For permutation groups we show the resulting chain of subgroups with the intermediate subgroups U used for switching from $N_G(T_i)$ to $N_G(T_{i+1})$. We also have labeled the edges describing subgroup relations by the index of the lower subgroup in the larger one.
If $\mathcal{B}=(B_1,B_2,...,B_k)$ is a partition of $\{1,...,n\}$ into blocks B_i the corresponding Young-subgroup of S_n is the normalizer $N_{S_n}(B_1,...,B_k)$ of all these blocks. Then a chain of subgroups may be selected as in the depicted example for $n=28$ (see Fig. 4).
The approach described above has to do orbit calculations where the number of points is at most 28 locally in this example. The number of cosets of the two-block normalizers in

the picture are given by the binomial numbers which grow exponentially. So each local problem is easy but one has to keep a long history of all previous results in memory. This really limits the applicability, since usually one needs these results in no predictable order. Thus each single call to some result in memory may cause a paging process which soon takes too much time.

3 Conjugation and Burnside's Lemma

The second well studied group action is conjugation on the subgroup lattice. If $U \leq G$ and $g \in G$ then $g^{-1}Ug \leq G$ is the subgroup conjugate to U under g.

Let $G = G_0 > G_1 > \ldots > G_n = \{id\}$ be a chain of normal subgroups of G, $G_i \lhd G$. Then for any i the mapping $\varphi_i\colon G_iU/G_i \mapsto G_{i-1}U/G_{i-1}$ for subgroups U of G is compatible with conjugation. One can therefore apply 1.7 to describe the subgroups up to conjugation.

In an iteration step a complete system of representatives V/G_{i-1} of subgroups of G/G_{i-1} is already available together with the stabilizer $N_{G/G_{i-1}}(V/G_{i-1})$.

Now V/G_{i-1} corresponds to V containing G_{i-1} as a subgroup of G. The description of V is usually easily computed from the representations of G and G_{i-1}.

Algorithm 1.7 needs a routine which computes representatives U/G_i for $\varphi_i^{-1}(V/G_{i-1}))$ in G/G_i under the action of $N_G(V)$. Such a U then has the properties $G_{i-1}U = V$ and $G_i \leq U$. If G_{i-1} is contained in every maximal subgroup of V then $U = V$ is the only possibility. Otherwise those maximal subgroups U/G_i of V/G_i not containing G_{i-1} belong to $\varphi_i^{-1}(V/G_{i-1})$.

Moreover any U/G_i such that $G_{i-1}U = V$ is contained in some M/G_i. Determining these subgroups U is usually not easy, but in the case of $p-$groups G, i.e. groups of order p^n for some prime p and $n \in \mathbf{N}$ this can be reduced to solving a system of linear equations and has been implemented in the SOGOS system [45]. This paper also contains a discussion of the case that G is a solvable group. Here [15] and the GAP system [16] contain more advanced methods and implementations.

The general case may be handled using in addition the classification of finite simple groups and their automorphism groups. Here we cannot cite all relevant literature, already the algorithms for $p-$groups and soluble groups form a research area of its own, a good introduction is [13].

Besides this obvious homomorphism onto factor groups one can also use a localization method to obtain representatives from conjugacy classes of subgroups. Here we assign to each group U a unique subgroup $F(U)$, of a special structure, for example the Fitting subgroup in the case of soluble finite groups, the generalized Fitting subgroup in the case of all finite groups, or the Thompson subgroup in case of finite $p-$groups [5]. We obtain $\varphi\colon \mathcal{L}(G) \to \mathcal{L}(G)$ which is compatible with conjugation in G. Then by 1.7 we can deduce from a set of representatives T of the image groups orbit representatives of $N_G(T)$ on the set $\varphi^{-1}(T)$. Usually T is normal in each $U \in \varphi^{-1}(T)$ such that $U \subset N_G(T)$. So only

$N_G(T)$ is needed for further investigation. Since $T \trianglelefteq N_G(T)$, we can go on with $N_G(T)/T$ as described above. For constructive purpose this strategy has been applied in [42] to soluble subgroups U of $GL(n,p)$.

In this paper we are not primarily interested in algorithms for analysing groups but only in using group actions for classification purposes.

Now assume that G is a group acting on a set Ω. The next homomorphism we discuss occurs in any situation where a group G acts on a set Ω.
Let $\mathcal{L}(G)$ be the lattice of subgroups of the group G. Then G acts on $\mathcal{L}(G)$ by conjugation. The Galois-mapping

$$\varphi\colon\ \begin{array}{l}\Omega \to \mathcal{L}(G)\\ \omega \mapsto N_G(\omega)\end{array}$$

which sends each point onto its stabilizer is a $G-$Homomorphism.
Applying lemma 1.6 to this situation reduces the problem of finding orbit representatitives to the following steps (see also [43]).

3.1 Algorithm (Approximation via subgroup lattice).

(1) Compute a set of representatives from the conjugacy classes of subgroups of G. For each representative U compute $N_G(U)$, its normalizer in G.

(2) For each representative U from step 1 compute the set $\varphi^{-1}(U)$ of all points ω in Ω having $N_G(\omega) = U$.
Determine the $N_G(U)/U$ - orbits on $\varphi^{-1}(U)$.

$\bigtriangledown$

We notice that in this situation $N_G(U)/U$ acts semiregularly on $\varphi^{-1}(U)$, since U is by definition the stabilizer of each $\omega \in \varphi^{-1}(U)$. Thus we have an immediate consequence.

3.2 Corollary. *If elements $\omega \in \varphi^{-1}(U)$ are chosen uniformly at random, all orbits of G on Ω where the point stabilizers are conjugate to U, i.e. of type U, occur with equal probability.*

Often it is much easier to determine all points ω which are fixed by U, i.e. $U \leq N_G(\omega)$, than those which have $U = N_G(\omega)$.
It is clear that

$$C_\Omega(U) = \{\omega \mid \omega \in \Omega \wedge \forall u \in U \omega^u = \omega\}$$

and $\varphi^{-1}(U)$ are related by
(3.3) $$\cup_{V:U\leq V}\varphi^{-1}(V) = C_\Omega(U).$$
Thus $\varphi^{-1}(U) = C_\Omega(U) - \cup_{V:U<V} C_\Omega(V)$.
If one is interested in cardinalities, equation 3.3 yields

$$|C_\Omega(U)| = \sum_{V:U\leq V} |\varphi^{-1}(V)|.$$

Then the ζ matrix of the lattice $\mathcal{L}(G)$, i.e. $\zeta(U,V) = \begin{cases} 1 & \text{if } U \leq V \\ 0 & \text{else} \end{cases}$ allows to write these relations shortly for all subgroups:

$$|C_\Omega| = \zeta \cdot |\varphi^{-1}|$$

where

$$|C_\Omega| = (|C_\Omega(U_1)|, |C_\Omega(U_2)|, \ldots, |C_\Omega(U_r)|)^t,$$

$$|\varphi^{-1}| = (|\varphi^{-1}(U_1)|, |\varphi^{-1}(U_2)|, \ldots, |\varphi^{-1}(U_r)|)^t$$

and $U_1, \ldots, U_r$ are all subgroups of G.

If $\mathcal{L}(G)$ is topologically sorted, i.e. $U_i \leq U_j \Rightarrow i \leq j$, then ζ is an upper triangular matrix with entries 1 on the diagonal. This shows that ζ has an inverse, called the Moebius function μ, such that

$$|\varphi^{-1}| = \mu \cdot |C_\Omega|.$$

The remark before corollary 3.2 shows that $\frac{1}{|N_G(U)/U|} \cdot |\varphi^{-1}(U)|$ is just the number of orbits of type U. So if D is the diagonal matrix with all $|N_G(U)/U|^{-1}$ as entries in the given ordering of the subgroups U,

$$D \cdot \mu \cdot |C_\Omega|$$

counts the orbits by stabilizer-type.
Of course this also holds for intervals $[U, G]$ in the subgroup lattice. In special cases we can thus obtain results on fantastically large objects.

3.4 Corollary. *If U is a maximal not normal subgroup of G, then $C_\Omega(U) - C_\Omega(G)$ is a full set of representatives for all orbits of type U.*

This means that for such a large prescribed automorphism group U finding the fixed points already suffices. No isomorphism test for the objects constructed in this way is needed at all.
As we have shown above the approximation of the orbit representative problem via subgroup lattice does not need all subgroups of G but only representatives from the conjugacy classes of subgroups. This does not hold for the inversion technique for determining $\varphi^{-1}(U)$. Of course we only have to determine $\varphi^{-1}(U)$ for representatives U of the conjugacy classes of subgroups. But if we want to determine $\varphi^{-1}(U)$ from the set of fixed points of U we have to subtract points fixed by any subgroup containing U. If U is contained exactly in $V_1, \ldots, V_m$ from the same conjugacy class then $\Delta = \cup_{i=1}^m \varphi^{-1}(V_i)$ is the set of those fixed points of U which have to be eliminated from $C_\Omega(U)$ in the inversion step with respect to the conjugacy class of V. So if we know $g_1, \ldots, g_m$ such that $V^{g_i} = V_i$ then

$$\Delta = \cup_{i=1}^m \varphi^{-1}(V^{g_i}) = \cup_{i=1}^m (\varphi^{-1}(V))^{g_i}.$$

If we are only interested in numbers, $|\varphi^{-1}(V)| = r \cdot |N_G(V)/V|$, where there are exactly r orbits with V as stabilizer of some point. Then $|\Delta| = m \cdot r \cdot |N_G(V)/V|$, and

$$m = \frac{|\Delta|}{r \cdot |N_G(V)/V|}.$$

Thus we must either know the number m of subgroups conjugate to V which contain U or the number of fixed points $|\Delta|$ of U that have a stabilizer conjugate to V. The formula for m also can be interpreted as a divisibility condition, since $m \in \mathbf{N}$.
The above Moebius inversion can be reduced from the lattice of all subgroups to *representatives* of the *conjugacy classes* and even more we could reduce to only those *which in addition occur as a point stabilizer.* Instead of ζ we now form a matrix M which for each (U, V) such that $U < V^g$ for some $g \in G$ contains the entry

$$m(U, V) = |\{V^g \mid g \in G \wedge U \leq V^g\}|.$$

Applying M^{-1} to the vector $(|C_\Omega(U_1)|, \ldots, |C_\Omega(U_s)|)^t$ of fixed point cardinalities for these representatives yields $(|\varphi^{-1}(U_1)|, \ldots, |\varphi^{-1}(U_s)|)^t$ such that multiplying this vector with

$$\operatorname{diag}(\frac{1}{|N_G(U_1)/U_1|}, \ldots, \frac{1}{|N_G(U_s)/U_s|})$$

then gives the vector $r^t = (r(U_1), \ldots, r(U_s))^t$, where there are exactly $r(U_i)$ orbits of G on Ω with U_i as stabilizer of some point.
The matrix M above is usually called the table of marks after Burnside [12]. It contains only information on the subgroup lattice, not on the special group action.
The mere counting result has been described by several authors, see for example [69]. For the Moebius Inversion on the subgroup lattice see [60]. For the constructive approach see [43].

A severe drawback of this approach is the need of a knowledge of all overgroups of a prescribed stabilizer. It is often very difficult to obtain this information. Algorithmically, no general way way working also for large indices seems to be known. In some cases one can use a localization technique to reduce the acting group to a generally smaller group. So. if a guess is made for the automorphism group of the structures searched for a good guess might at least find a correct Sylow subgroup of the automorphism group. Then the following holds.

3.5 Lemma. *Let $\omega_1, \omega_2 \in \Omega$ and $g \in G$ such that $\omega_1^g = \omega_2$. Let P be a Sylow subgroup of $N_G(\omega_2)$. Then $\omega_1^n = \omega_2$ for some $n \in N_G(P)$.*

Proof. Since $P^g \leq N_G(\omega_1)^g = N_G(\omega_1^g) = N_G(\omega_2)$ and $P \leq N_G(\omega_2)$, there is some $x \in N_G(\omega_2)$ such that $P^g = P^x$ by the Sylow Theorem. Then $gx^{-1} \in N_G(P)$ and $g = nx$. Therefore $\omega_2 = \omega_1^g = \omega_1^{nx}$ and $\omega_1^n = \omega_2^{x^{-1}} = \omega_2$.

□

Generally we use the following notion in this situation.

3.6 Definition. *Let a group G act on a set Ω and let $A \leq G$, $H \leq G$. Then H controlls the G-fusion on the set $C_\Omega(A)$ of fixed points of H if $\omega_1^g = \omega_2$ for $\omega_1, \omega_2 \in C_\Omega(A))$ and $g \in G$ implies the existence of some $h \in H$ such that $\omega_1^h = \omega_2$.*

So, if the prescribed subgroup A of the automorphism group of the objects that are searched for contains a Sylow subgroup P of all the automorphism groups of the objects then $N_G(P)$ controlls the G-fusion on the set of all objects fixed by A. In our applications to t-designs it is often possible to show that no design exists if the proposed subgroup P is enlarged to a larger p-group. Then the assumptions of the Lemma are of course fulfilled.

There is a problem if A is not normalized by $N_G(P)$. Then, usually, the set of fixed points of A is not closed under $N_G(P)$. So we cannot just form the orbits of $N_G(P)$ to solve the isomorphism problem. But, having a subgroup A of the automorphism groups which is greater than P has also the advantage to reduce the size of the Kramer-Mesner matrix and, as a consequence, to often reduce the number of solutions of the diophantine system to solve. So, suppose that $\mathcal{D}_1$ and $\mathcal{D}_1$ are two designs admitting A as an automorphism group. We assume the situation of the Lemma such that if the two designs are isomorphic one is mapped onto the other by some $g \in N_G(P)$. Then $A^g \leq Aut(\mathcal{D}_2)$ and also $A \leq Aut(\mathcal{D}_2)$. If A is a large subgroup of the full symmetric group S_n then it happens very often that $< A, A^g > = S_n$ or $= A_n$. If this situation appears for each $g \in N_G(P) \setminus A$ then we conclude that all designs admitting A as an automorphism group are pairwise non-isomorphic. This is remarkable, since we did not have to touch upon any of the designs. Generally, we have the following result, using the notation from above.

3.7 Lemma. *Let H control the G-fusion on $C_\Omega(A)$ and let R be a transversal of the set of right cosets of $N_H(A)$ in H. Then $N_H(A)$ acts on $C_\Omega(A)$. If $\omega_1^g = \omega_2$ for $\omega_1, \omega_2 \in C_\Omega(A)$ and $g \in R$ but $N_H(A)g \neq N_H(A)$ then $A < \langle A, A^g \rangle \leq N_G(\omega_2)$.*

We point out a special case.

3.8 Corollary. *If A contains the normalizer of a Sylow subgroup of G then $C_\Omega(A)$ is a full set of representatives for all orbits where A is contained in the stabilizer of some point.*

An important case where the condition of the corollary is fulfilled is the projective group $PGL(2,p)$ for some prime p. The projective group is the permutation representation of the general linear group $GL(2,p)$ on the set of all $p+1$ subspaces of dimension 1 of the underlying vector space $V = V(2,p)$. The projective group has order $(p+1)p(p-1)$ and contains a Sylow p-subgroup of the full symmetric group S_{p+1}. The normalizer N of a 1-dimensional subspace T of V in $GL(2,p)$ has order $p(p-1)^2$ and contains the centralizer of T and V/T as a normal subgroup. This centralizer is just of order p and therefore a normal subgroup of N. If we reduce modulo the center Z of $GL(2,p)$ which

is of order $(p-1$ we obtain that PZ/Z is a normal subgroup of NZ/Z and NZ/Z has order $p(p-1)$. Now this is just the order of the normalizer of a Sylow p-subgroup of S_{p+1} such that $PGL(2,p)$ contains the normalizer of a Sylow subgroup of S_{p+1}. So whenever we construct objects where $PGL(2,p)$ acts as a group of automorphisms in its natural permutation representation all these objects are pairwise nonisomorphic. Often only $PSL(2,p)$ appears as an automorphism group. For $p>2$ this group has index 2 in $PGL(2,p)$ such that by Lemma 3.4 only the $PGL(2,p)$ orbits of length at most 2 on the set of fixed points of $PSL(2,p)$ have to be considered.
We now turn to results which are specific for designs.

Similar to a result of Aschbacher [**?**] on symmetric designs there is a result for general t-designs which is helpful in recursive situations.

.

3.9 Proposition. *Let $\mathcal{D}$ be a t-(v,k,λ) design and $g \in G$ with v_g fixed points. Let $o(g) = p$ be a prime such that $\lambda < p$ and $k < p+t$. Then if $t \leq v_g$ all blocks of $\mathcal{D}$ consisting of fixed points of g form a t-(v_g,k,λ) design.*

Proof Let $C_V(g)$ be the set of fixed points of g. Let $T \subseteq C_V(g)$ and $\mathcal{D}_T = \{B \mid |b \in \mathcal{D},\ T \subseteq B\}$. Then g acts on $\mathcal{D}_T$ which is of size λ. By the assumptions $\lambda < p$ such that g leaves each $B \in \mathcal{D}_T$ invariant. Since at least the t points of T are fixed, g has to act on the remaining point sets $B \setminus T$ of size $k-t$. Again the assumption $k-t<p$ implies that g fixes also all these point sets. Thus, all blocks from $\mathcal{D}_T$ are subsets of $C_V(g)$.
□

Usually, one is interested in designs with small λ such that $\lambda < p$ is often satisfied. Also, $k-t<p$ is a restriction that will be fulfilled in many cases. Only the assumption that g has at least t fixed points seems to be very special. So, we use the result to produce contradictory situations in a proof. If it is known that no t-(v_g,k,λ) design can exist for all $v-p > v_g \geq t$ then it follows that $v_g < t$. So, in this case g consists of as many p-cycles as possible, say c, and a few additional fixed points.
One such situation appears if $t<p$ and p does not divide $\binom{v}{t}$. Then g in its action on all t subsets must have some invariant t-subset T which because of $t<p$ must consist of fixed points of g.

We say that a prime p is *large* for t-(v,k,λ) if $t<p,\ k-t < p,\ \lambda < p, v<p^2$.

3.10 Proposition. *Let p be large for t-(v,k,λ) and v chosen minimal among these parameter sets such that $p \mid |Aut(\mathcal{D}|$ for some t-(v,k,λ) design $\mathcal{D}$. Then for this $\mathcal{D}$ all $Sylow_p$-subgroups of $Aut(\mathcal{D})$ have order p.*

Proof By the preceding proposition an automorphism g of order p of $\mathcal{D}$ has only $c<t$ fixed points. All other points lie in cycles of length p. Suppose, $Aut(\mathcal{D})$ has a subgroup P of order p^2. We show that then no element other than the identity of P leaves any block of $\mathcal{D}$ invariant. Otherwise, some element g of order p leaves such a block B invariant.

Then g acts on B, a set of k elements. Since the number of fixed points of g in B is smaller than t, and $t < p$ then g must not fix any t-subset of B such that p divides $\binom{k}{t}$. But then p does not divide $b = \binom{v}{t}/\binom{k}{t}\lambda$, since p divides the enumerator at most to the first power. Then there must be orbits of P on the set of blocks whose length is not a multiple of p. But each orbit length of P must be a power of p, such that there exists a block B which is invariant under P. Since $2p > k$, the symmetric group on B contains no subgroup of order p^2. So there is some element g of order p in P acting trivially on B. This contradicts the preceding proposition. Now each orbit of P on $\mathcal{D}$ has length p^2 such that the number of blocks b is divisible by p^2. But since $t < p$ the number $\binom{v}{t}$ is a product of t subsequent factors of which only one may be a multiple of p. Thus, we have a final contradiction.

□

In such a situation $C_{S_v}(g)$ is known to be $S_{v_g} \times (C_p \wr S_r)$. If $r < p$ then this centralizer contains a Sylow subgroup P of S_v. Since then P is abelian, every Sylow p-subgroup is abelian. Each Sylow subgroup containing g thus lies in the centralizer of g and has to coincide with P. We thus then find that g is contained in only one Sylow subgroup. We seek a situation where we can show that g generates already a Sylow subgroup of the automorphism group of each design admitting the prescribed automorphism group G. We obtain from our general results that then any two such designs may only be isomorphic if they are mapped one onto the other by some element of $N_{S_v}(< g >)$. The latter normalizer is an extension of $S_{v_g} \times (C_p \wr S_r)$ by the automorphism group of C_p.

Example: If we let M_{11} act diagonally on two copies of the underlying set of 11 points, formally we have the action of $M_{11} \times < id_2 >$, then we get four solutions for a 5-$(22, 7, 40)$ design with this prescribed automorphism group. Then $p = 11$ is a large prime for this parameter set. Also this v is minimal in the sense of the Theorem. Therefore any two of the four designs may only be isomorphic if some element from the normalizer of a Sylow_{11} subgroup of the automorphism group maps one design onto the other. Adding the permutation of order two which interchanges the two actions of M_{11} we get a group which is larger by a factor of 2. This group is still an automorphism group of two of the designs. The new element obviously lies in the normalizer of a Sylow_{11} subgroup. It interchanges two of the designs found. The remaining two designs admit the interchange of the two actions as an automorphism. These designs are not isomorphic because the normalizer of the Sylow subgroup contains no element mapping one design onto the other. It is also possible to distinguish them by their intersection numbers.

Taking $A = PGL(3, 5)$, a group of order $(5^2 + 5 + 1)(5^2 - 1)(5 - 1)5^3$ acting on the set of 31 one-dimensional subspaces of $V(3, 5)$ as a prescribed automorphism group we found 8-$(31, 1093)$ and 8-$(31, 10, 100)$ designs. To solve the isomorphism problem for these designs we used that A contains a Sylow-31 subgroup P of S_{31}. The normalizer of P in the full symmetric group is the holomorph of P, i. e. the semidirect product of P with its automorphism group. This normalizer is not contained in A but $N_{S_{31}}(P) \cup A = 3 \times 31$. This intersection has 10 right cosets in $N_{S_{31}}(P)$. For representatives g from the nontrivial right cosets we formed $\langle A, A^g$ and in each case obtained A_{31}. Thus, by the above theory, all designs obtained as solutions of the Kramer-Mesner system are pairwise non-isomorphic.

We remark that the constructive approach presented here generalizes to some interesting cases of infinite groups acting on infinite sets. The essential idea is to determine all fixed points of a given subgroup U, subtract all those having a larger stabilizer than U and then determine representatives of the normalizer of U on the remaining set of points. In the case that there are only finitely many conjugacy classes of stabilizers, a finite index of each such stabilizer in its normalizer, and a finite number of orbits this approach leads to a situation which can be handled by a computer algorithm. For example this is the basis of the classifications of crystallographic groups which are of great importance in Theoretical Physics.

If the table of marks is constructed for a full set of representatives for the conjugacy classes of subgroups, it can be used to identify the stabilizer class of any given orbit of G on any set Ω. We have to know the vector of fixed point cardinalities of the U_i on the orbit Ω and obtain a resulting vector r with exactly one entry 1 for the corresponding stabilizer class.

3.11 Example. *Construction of block designs with prescribed automorphism group.*
A $t-(v,k,\lambda)-$design $\mathcal{B}$ consists of blocks of equally many elements from a set Ω, which in some sense approximates the set of all subsets of that size. The parameters make this concrete:

$$|\Omega| = v, \mathcal{B} \subseteq \binom{\Omega}{k}, \ t \leq k, and \forall T \in \binom{\Omega}{t}$$

there exist exactly λ blocks $B \in \mathcal{B}$ such that $T \subseteq B$.
Interesting parameters are for example

$$t-(v,k,\lambda) = 4-(28,6,72).$$

Then any block design with these parameters must have exactly 98280 blocks. This is easily seen by counting twice all pairs (T,B) where $T \in \binom{\Omega}{t}$, $B \in \mathcal{B}$ and $T \subseteq B$.
Constructing all isomorphism types of block designs with a given parameter set like this is usually regarded as hopeless. But as we shall see, prescribing a large automorphism group A makes such a task feasible for a computer. $\diamond$

Firstly, the incidence relation $T \subset B$ for $T \in \binom{\Omega}{t}$ and $B \in \mathcal{B}$ is invariant under A, i.e. $\forall \alpha \in A$ also $T^\alpha \subset B^\alpha$. Therefore we need only consider representatives B from the orbits of A on $\binom{\Omega}{k}$. If $B \in \mathcal{B}$ then also $B^\alpha \in \mathcal{B}$ for each $a \in A$. Also if T is contained in exactly $\lambda(B,T)$ blocks of $B^A = \{B^\alpha \mid \alpha \in A\}$, then the same is true for each T^α for $\alpha \in A$.
Thus, instead of all $T \in \binom{\Omega}{t}$ we only consider representatives from the A-orbits, and instead of all $B \in \binom{\Omega}{k}$ we only consider the set of A-orbits on $\binom{\Omega}{k}$.

A block design with a prescribed automorphism group A then is a union of some $A-$orbits $\mathcal{B}_i$ on $\binom{\Omega}{k}$ such that for a fixed set $\{B_1, \ldots, B_r\}$ of representatives from the chosen $\mathcal{B}_i$ for each representative T from the A orbits on $\binom{\Omega}{t}$ the equation

$$\sum_{i=1}^{r} \lambda(B_i, T) = \lambda \tag{3.12}$$

holds.

The construction problem is reduced drastically by considering only orbit representatives. This observation is due to Kramer-Messner [35]. These authors form a matrix M which has a row for each orbit of A on t-subsets and a column for each orbit of A on k-subsets. The entry in the i-th row and j-th column is just the number of k-subsets in the j-th orbit that contain a fixed t-subset of the i-th orbit. The selection of appropriate k-orbits for a $t-(v,k,\lambda)$ design means to find a $0-1$ vector u which multiplied by M yields a vector with constant entry λ. Thus, instead of considering each t-subset and separately one reduces to only representatives from the A orbits. Analogously, only orbits on k-subsets need to be considered.

We demonstrate this effect by a report on the construction of a 7-$(33,8,10)$ design. ¿From the parameters we find that such a design consists of $5,340,060$ blocks of length 8. They have to be selected out of all $13,884,156$ 8-subsets. Each subset T of the set $V=\{1,\ldots,33\}$ of cardinality 7 must be contained in exactly 10 of the chosen blocks. To find such a design we use a large subgroup $A = P\Gamma L(2,32)$ of S_{33} as a prescribed automorphism group, see [51]. The group A can be presented as generated by the following two permutations:

$$\begin{aligned}\alpha &= (1\ 2\ 4\ 8\ 16)(3\ 6\ 12\ 24\ 17)(5\ 10\ 20\ 9\ 18)(7\ 14\ 28\ 25\ 19)(11\ 22\ 13\ 26\ 21)\\ &\quad (15\ 30\ 29\ 27\ 23)(31)(32)(33)\\ \beta &= (1\ 18\ 30)(2\ 21\ 12)(3\ 10\ 28)(4\ 31\ 32)(5\ 24\ 14)(6\ 7\ 17)(8\ 25\ 27)(9\ 19\ 20)\\ &\quad (11\ 15\ 13)(16\ 23\ 29)(22\ 33\ 26).\end{aligned}$$

The order of A is 163680 such that the orbits of A have at most this length in any permutation representation. Since in an induced action most elements of a permutation group have only very few fixed points, the stabilizers of many points are trivial, that is most orbits will have maximal length. Therefore by regarding orbits instead of the points a reduction by approximately the factor $|A|$ can be achieved. This heuristic also holds in our example. A has 32 orbits on the set of all 7-subsets of V and 22 of them have maximal length. On the set of all 8-subsets of V we find 97 orbits 74 of which have maximal length. Orbit representatives may be computed by the approach we described above using split and fusion. We cite the representatives from [51].
To find these solutions is obviously much harder than to verify that they indeed are solutions. The problem is an instance of the subset sum problem which is known to be

Nr	orbits on 7-subsets of V representative							length
1.	1	2	3	4	5	6	7	81840
2.	1	2	3	4	5	6	8	163680
3.	1	2	3	4	5	6	9	163680
4.	1	2	3	4	5	6	10	163680
5.	1	2	3	4	5	6	11	163680
6.	1	2	3	4	5	6	12	163680
7.	1	2	3	4	5	6	13	163680
8.	1	2	3	4	5	6	14	81840
9.	1	2	3	4	5	6	15	81840
10.	1	2	3	4	5	6	16	163680
11.	1	2	3	4	5	6	17	163680
12.	1	2	3	4	5	6	19	81840
13.	1	2	3	4	5	6	32	163680
14.	1	2	3	4	5	7	9	163680
15.	1	2	3	4	5	7	10	163680
16.	1	2	3	4	5	7	12	163680
17.	1	2	3	4	5	7	13	163680
18.	1	2	3	4	5	7	15	81840
19.	1	2	3	4	5	7	20	163680
20.	1	2	3	4	5	7	24	81840
21.	1	2	3	4	5	8	10	163680
22.	1	2	3	4	5	8	11	163680
23.	1	2	3	4	5	8	12	163680
24.	1	2	3	4	5	8	13	163680
25.	1	2	3	4	5	8	17	81840
26.	1	2	3	4	5	8	24	163680
27.	1	2	3	4	5	8	26	163680
28.	1	2	3	4	5	9	11	163680
29.	1	2	3	4	5	9	12	163680
30.	1	2	3	4	5	9	17	32736
31.	1	2	3	4	5	10	12	32736
32.	1	2	3	4	5	11	16	32736

Nr	orbits on 8-subsets of V representative								length
1.	1	2	3	4	5	6	7	8	81840
2.	1	2	3	4	5	6	7	9	163680
3.	1	2	3	4	5	6	7	10	163680
4.	1	2	3	4	5	6	7	11	163680
5.	1	2	3	4	5	6	7	12	163680
6.	1	2	3	4	5	6	7	13	163680
7.	1	2	3	4	5	6	7	14	163680
8.	1	2	3	4	5	6	7	15	163680
9.	1	2	3	4	5	6	7	16	163680
10.	1	2	3	4	5	6	7	17	163680
11.	1	2	3	4	5	6	7	18	81840
12.	1	2	3	4	5	6	7	19	163680
13.	1	2	3	4	5	6	7	32	163680
14.	1	2	3	4	5	6	8	9	163680
15.	1	2	3	4	5	6	8	10	163680
16.	1	2	3	4	5	6	8	12	163680
17.	1	2	3	4	5	6	8	13	163680
18.	1	2	3	4	5	6	8	14	163680
19.	1	2	3	4	5	6	8	15	163680
20.	1	2	3	4	5	6	8	16	163680
21.	1	2	3	4	5	6	8	17	163680
22.	1	2	3	4	5	6	8	19	163680
23.	1	2	3	4	5	6	8	20	163680
24.	1	2	3	4	5	6	8	21	163680
25.	1	2	3	4	5	6	8	23	163680
26.	1	2	3	4	5	6	8	24	163680
27.	1	2	3	4	5	6	8	26	163680
28.	1	2	3	4	5	6	8	27	163680
29.	1	2	3	4	5	6	8	30	81840
30.	1	2	3	4	5	6	8	32	163680
31.	1	2	3	4	5	6	8	33	163680
32.	1	2	3	4	5	6	9	10	163680
33.	1	2	3	4	5	6	9	11	163680
34.	1	2	3	4	5	6	9	12	163680
35.	1	2	3	4	5	6	9	13	81840
36.	1	2	3	4	5	6	9	14	163680
37.	1	2	3	4	5	6	9	15	163680
38.	1	2	3	4	5	6	9	17	163680
39.	1	2	3	4	5	6	9	18	163680
40.	1	2	3	4	5	6	9	19	163680
41.	1	2	3	4	5	6	9	22	81840
42.	1	2	3	4	5	6	9	23	81840
43.	1	2	3	4	5	6	9	24	163680
44.	1	2	3	4	5	6	9	26	163680
45.	1	2	3	4	5	6	9	27	163680
46.	1	2	3	4	5	6	9	29	81840
47.	1	2	3	4	5	6	9	33	163680
48.	1	2	3	4	5	6	10	11	163680
49.	1	2	3	4	5	6	10	12	163680
50.	1	2	3	4	5	6	10	13	163680
51.	1	2	3	4	5	6	10	15	163680
52.	1	2	3	4	5	6	10	18	163680
53.	1	2	3	4	5	6	10	19	163680
54.	1	2	3	4	5	6	10	20	163680
55.	1	2	3	4	5	6	10	22	81840
56.	1	2	3	4	5	6	10	24	163680
57.	1	2	3	4	5	6	10	25	163680
58.	1	2	3	4	5	6	10	26	163680
59.	1	2	3	4	5	6	10	28	81840
60.	1	2	3	4	5	6	10	32	81840
61.	1	2	3	4	5	6	11	12	81840
62.	1	2	3	4	5	6	11	14	163680
63.	1	2	3	4	5	6	11	16	163680
64.	1	2	3	4	5	6	11	20	81840
65.	1	2	3	4	5	6	11	21	163680
66.	1	2	3	4	5	6	11	22	163680
67.	1	2	3	4	5	6	11	23	163680
68.	1	2	3	4	5	6	11	25	163680
69.	1	2	3	4	5	6	11	26	163680
70.	1	2	3	4	5	6	11	27	81840
71.	1	2	3	4	5	6	11	33	163680
72.	1	2	3	4	5	6	12	13	163680
73.	1	2	3	4	5	6	12	15	81840
74.	1	2	3	4	5	6	12	17	163680
75.	1	2	3	4	5	6	12	20	163680
76.	1	2	3	4	5	6	12	24	163680
77.	1	2	3	4	5	6	12	26	81840
78.	1	2	3	4	5	6	12	32	163680
79.	1	2	3	4	5	6	13	16	163680
80.	1	2	3	4	5	6	14	24	81840
81.	1	2	3	4	5	6	16	17	163680
82.	1	2	3	4	5	6	16	22	20460
83.	1	2	3	4	5	6	16	33	163680
84.	1	2	3	4	5	6	17	19	163680
85.	1	2	3	4	5	6	17	33	163680
86.	1	2	3	4	5	7	9	12	163680
87.	1	2	3	4	5	7	9	17	163680
88.	1	2	3	4	5	7	9	32	163680
89.	1	2	3	4	5	7	10	20	81840
90.	1	2	3	4	5	7	10	32	81840
91.	1	2	3	4	5	7	12	15	163680
92.	1	2	3	4	5	7	12	17	81840
93.	1	2	3	4	5	7	12	24	81840
94.	1	2	3	4	5	7	13	26	163680
95.	1	2	3	4	5	8	10	15	163680
96.	1	2	3	4	5	8	13	19	81840
97.	1	2	3	4	5	9	12	24	81840

Table 1: The 32×97 Kramer-Mesner matrix M

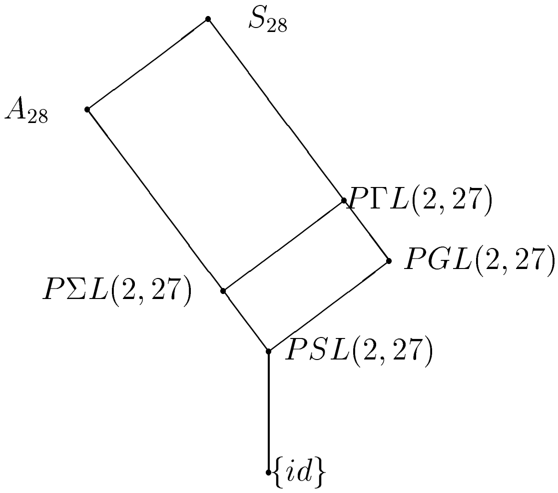
```
2222222222222000000000000000000000000000000000000000000000000000000000000000000000000000000000000
2101110000000211112111111111111000000000000000000000000000000000000000000000000000000000000000000
1110000001000100200000000000100011111211211111111000000000000000000000000000000000000000000000000
0011000000000002100100000000000010001100000001000121311111111100000000000000000000000000000000000
0001200000000000000000100000100111000000000100100100110001000011121111111100000000000000000000000
0000110000000000100000010002000000100100000110011100000001010010000000110021111110000000000000000000
0000013000110100100000101001001000100020001000100101010010000000100010001001000100000000000000000
0000004200000000020000200220000000002000000200000000000020000020000000000000200020000000000000000
0000000220000000002000002000002000002000020000000020000000000200002000022020000000000000000000000
0001000121000001100100000101000000000100010001010010000100000000100111000001010010111000000000000000
0000002001200010011011010000110000001101000110010000010000000000100000000100001010011000000000000
0000000200020200000000220002000020000000220000000200002000000000000000000000200000000200000000000000
0000000001012000000000101000022000000000000000011100000001100101100001001001000102001010000000000000
0200000000000221002000010100000111100000000010000000000000000100011001010000000100001011100000000
0010100000001001001200100000010001000002000000001001011011000100000000000001010000000102110000000
0000110011000010100001020100000000000000010100000001010000000101001000000000010000000010211001110000
0011020000000010010000000100100000010001000000000000101000010000010001100001000101021200000100100
0000000220000000000240002000000000000000000000000000000000000020022000000002202000000000002000000
0000010010011000010110010011001110000100000000100000101000100000000010000001011000100000000100011000
0200000200000200000200000200020002000000000202000000000000000200000000000000200020000000000000020000
0001000001010011000000000000100001100010001000010000100000100001100000010000000002001003101010 0100
0010100000001100010200100100000000010001100001011001001100000000000200000100000000000010000010000300
0200000000000001000001000000000010220100000110000000000000100100100110000000100011010021000010000100
0011000000000000100001000000010100011010000000100010000000101000000201000000101000000010100200010012
0000000200000000000002002000000000000000200000000200000000200000000020020220000000000000200020000200
0010000000000000000001100001010000000000000010000000100101000020010000001101110011000000000111000000211
0000000000010000010000011010000000000000100010200000000200110000200000010010000111000100110001001010
0000000000000000000010010100010110000100000100001010101100000001000101000201001100000000000003001100
0000000010000000100000000000101000110100010010000111001000100000001001000110001000000000111001011001
0000000005000000000000000000000000000000050000000000050005000000000000000000000000000000000000005000000 1
0000000000000000000000000000000000000000000000005000000000500000000500050000000000050000000000000001
0000000000000000000000000000000000000000000000000050000000000000500000000000000000000005005000000500001
```

with the solution vectors v for $\lambda = 10$ and $\lambda = 16$ respectively:

```
0011100010100100110001101000010010000000101101100010111100000000100100101000011011101100101011100 0
1100011101011011001110010111101101111111010010011101000011111110110110101111001000100110101000111
```

NP-complete [20]. We remark that the number of blocks in $\mathcal{B}$ can be computed from the parameters and on the other hand is the sum over all orbit lengths $|\mathcal{B}_{\rangle}|$ of the selected orbits on the k-subsets. This is just an additional linear equation for the system 3.12. Here we shall not discuss the problem of solving the above system of linear diophantic equations, see [6, 62]. In our context we want to show how a large automorphism group A can be used to solve the isomorphism problem.

Of course two $t - (v, k, \lambda)$ designs $\mathcal{B}, \mathcal{B}'$ are isomorphic if and only if they only differ in the names of the block entries. If the entries are taken from the same set Ω this means that there exists a permutation $\pi \in S_\Omega$ such that replacing each entry in the blocks of



$\mathcal{B}$ by the image under π results in the blocks of $\mathcal{B}'$. Thus the isomorphism types can be described by orbits of S_Ω on the set of all subsets of $\binom{\Omega}{k}$. The solutions of (G) form just the set of fixed points of A in this induced action of S_Ω on the set of all block designs. If we determine by (1) all designs with stabilizer exactly A, then the orbits of $N_{S_\Omega}(A)/A$ on this set give exactly the different isomorphism types with an automorphism group conjugate to A in S_Ω.
For our above mentioned parameters this has been done by Schmalz[62] [63] where A is one of the groups $PSL(2,27), PGL(2,27), P\Sigma L(2,27), P\Gamma L(2,27)$. In all these cases $N_{S_{28}}(A) = P\Gamma L(2,27)$.
The number of isomorphism types of block-designs with these automorphism groups can be obtained by Moebius-inversion from the numbers of fixed points as we showed above. Here Schmalz found the following concrete numbers:

$$\begin{pmatrix} \frac{1}{6} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 13078960 \\ 704 \\ 58 \\ 8 \end{pmatrix} = \begin{pmatrix} 2179701 \\ 232 \\ 25 \\ 8 \end{pmatrix}$$

13078960 is the number of fixed points of $PSL(2,27)$,
704 is the number of fixed points of $PGL(2,27)$,
58 is the number of fixed points of $P\Sigma L(2,27)$, and
8 is the number of fixed points of $P\Gamma L(2,27)$.

No other subgroup of S_{28} containing any such A has any fixed point. In fact only S_{28} and A_{28} have to be considered. The number of isomorphism types is then given by the resulting vector on the right hand side.
It should be remarked that Schmalz has found by these methods interesting new parameter sets for which he could determine all isomorphism types for some prescribed automorphism group. In particular he obtained some new parameter sets with $t = 6$.

3.13 Example. An illustrative example is the determination of chemical isomers of a certain kind. We consider a problem which has gained interest in the last years. Chemists were able to produce molecules of the form of a platonian body, the dodecahedron was found as an organic molecule about 10 years ago, and also the archimedian body obtained from an icosahedron by cutting off each vertex and replacing it in this way by a cycle of length 5. The resulting body is known as a football to many of us. Since B. Fuller has constructed famous buildings with similar structures chemists often call this body a Buckminster Fullerene.
For such a rigid body one can consider the vertices as places where some ligands may be planted or some spin orientations may be placed. Of course turning the resulting molecules around will transform one such constellation into a similar one with the same chemical properties. We are thus interested in finding the different possibilities, which can not be identified by a rotation.
Mathematically we assign to each vertex some colour, i.e. we have a mapping

$$f\colon \text{Vertex set} \rightarrow \text{Colour set}.$$

Two mappings f_1, f_2 are equivalent, if and only if there exists an automorphism α of the body such that

$$\alpha^{-1} \cdot f_1 = f_2.$$

This is the mathematical problem that Pólya treated in his famous paper "Kombinatorische Anzahlbestimmungen für Gruppen, Graphen und chemische Verbindungen" [58].
While Pólya counted the orbits of a permutation group on the set of all mappings, we are interested in the construction of a set of orbit representatives. So our approach will be applicable to all situations where the theory of counting could be used before.
In this setting we can use equation (1) to obtain a more explicit description. Suppose $U \leq G$ partitions Ω into the orbits $\Omega_1, \Omega_2, \ldots, \Omega_l$. Then exactly those mappings $f\colon \Omega \rightarrow Y$, Y a colour set, are fixed by U, which are constant on each Ω_i. We can therefore give a closed form for $C_{Y^\Omega}(U)$:

$$C_{Y^\Omega}(U) = \prod_{i=1}^{l} \cup_{y \in Y} \{y\}^{\Omega_i}$$

Here $\{y\}^{\Omega_i}$ is the unique mapping which colours each point of Ω_i with colour y, and $\cup_{y \in Y} \{y\}^{\Omega_i}$ is the set of all mappings which are constant on Ω_i.

If $F_1 \subseteq Y^{X_1}$ and $F_2 \subseteq Y^{X_2}$, $X_1 \cap X_2 = \emptyset$, then

$$F_1 \cdot F_2 = \{f \in Y^{X_1 \cup X_2} \mid f \mid_{X_1} \in F_1 \wedge f \mid_{X_2} \in F_2\}.$$

In this way we can describe each $\varphi^{-1}(U)$ in 3.1 by subtracting from $C_{Y^\Omega}(U)$ all those mappings which are constant on the orbits of any subgroup V which is a proper overgroup of U. We worked this out in [43] for the example of a dodecahedron. A discussion of the Buckminster Fulleren C_{60} is contained in [46]. $\diamond$

For the general Pólya situation when a group G acts on a set F of mappings, $F \subseteq Y^X$, a refinement using homomorphisms can be obtained by restricting mappings to $G-$orbits on X.
If $X = X_1 \cup X_2$ and X_1, X_2 are invariant under G, especially if X_1 is a $G-$orbit, then

$$\varphi: Y^X \to Y^{X_1},\ f \mapsto f \mid_{X_1}$$

is a $G-$homomorphism.
The homomorphism principle then suggests first to colour X_1, i.e. compute representatives and their corresponding normalizers in G and recursively then solve the colouring problem for X_2 under the action of the normalizers of representatives from Y^{X_1}/G.
Usually the normalizers are smaller than G such that they have smaller orbits. Thus, in the recursion the sets which have to be coloured tend to become smaller.*If the normalizer has become trivial then all colourings of the remaining points represent different isomorphism types.* In this case, an explicit listing of all possibilities can be replaced by a routine which produces all these mappings one by one if needed. In case of many data this will save memory space of a computer and thus reduce the paging necessity.

As we have seen in section 2 the double cosets AgB of two subgroups A, B of a group G correspond to the orbits of B on $A\backslash G$ with respect to multiplication from the right. Now we refine our approach by 3.1, [46] . For that purpose we need to determine for representatives U of the conjugacy classes of subgroups of B those right cosets Ag with $N_B(\{Ag\}) = U$. Again it is easier to describe instead the fixed points of U on $A\backslash G$.

3.14 Lemma. *Let $K = \{g \in G | AgU = Ag\}$. Then*

$$K = \bigcup_{i=1}^{l} N_G(A) g_i N_G(U)$$

for appropriate $g_i \in K$. Each such g_i conjugates U onto a subgroup of A which is unique up to conjugation under $N_G(A)$:

$$g_i U g_i^{-1} \leq A \text{ for } i = 1, \cdots, l.$$

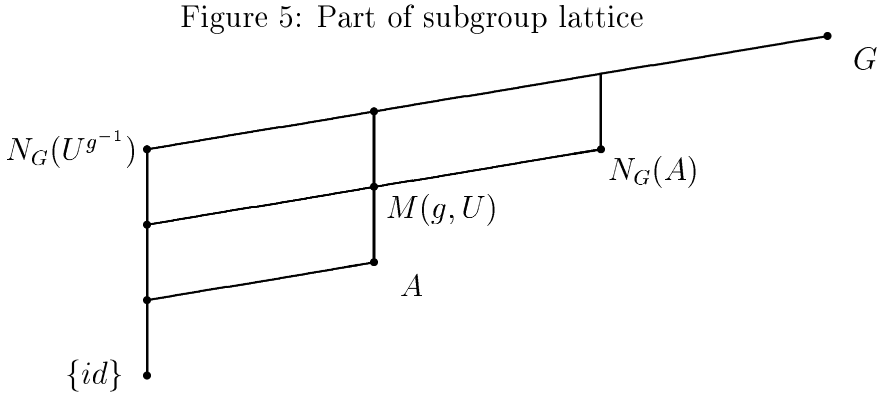
Proof. Let $g \in K$. Then $AgUg^{-1} = A$ and $gUg^{-1} \leq A$. If $m \in N_G(A)$ and $n \in N_G(U)$ then

$$AmgnU = mAgUn = mAgn = Amgn$$

such that $mgn \in K$. Therefore K consists of complete double cosets of $N_G(A)$ and $N_G(U)$ in G. If $x = mgn$ then xUx^{-1} is conjugate to gUg^{-1} under m^{-1}.

$\square$

Figure 5: Part of subgroup lattice



The lemma gives a first coarse description of the fixed points of U on $A\backslash G$. We have to find elements g_i conjugating U onto subgroups of A up to conjugation under $N_G(A)$. For each such g_i the double coset $N_G(A)g_iN_G(U)$ has to be refined to give more detailed informations on the fixed points of U.

3.15 Lemma. *Let*

$$N_G(U) = \bigcup_{j=1}^{s} N_{A^g}(U)x_jN_B(U), \text{ and } N_G(A) = \bigcup_{i=1}^{r} y_iM(g,U),$$

where $M(g,U) = A(N_G(gUg^{-1}) \cap N_G(A))$. *Then*

$$N_G(A)gN_G(U) = \bigcup_{i=1}^{r}\bigcup_{j=1}^{s} Ay_igx_jN_B(U).$$

For representatives x_j, x_k from different double cosets of $N_{A^g}(U)$ and $N_B(U)$ the double cosets $Aygx_jN_B(U)$ and $Aygx_kN_B(U)$ are different for any $y \in N_G(A)$. If $M(g,U) = AN_G(gUg^{-1}) \cap N_G(A)$ then the union $\bigcup_{i=1}^{r} Ay_igN_G(U)$ is disjoint.

By this Lemma local computations in the normalizers of U and A allow to deduce from one fixed point of U several new fixed points. We even can give a redundancy free list of representatives in important cases.

Proof of the Lemma. The intersection $N_G(gUg^{-1}) \cap N_G(A)$ is a subgroup of $N_G(A)$ and therefore normalizes A. Thus, $M(g,U)$ is a subgroup of $N_G(A)$. We claim that each $y' \in yM(g,U), y \in N_G(A)$, lies in the double coset $AygN_G(U)$.
Since $N_G(gUg^{-1}) = gN_G(U)g^{-1}$, for each $m \in M(g,U)$ there exists some $a \in A$ and some $n \in N_G(U)$ such that $m = agng^{-1}$. Therefore

$$AmgN_G(U) = Aagng^{-1}gN_G(U) = AgnN_G(U) = AgN_G(U).$$

If $y' = ym, m \in M(g,U)$, then $y^{-1}y' = m$ and

$$AgN_G(U) = AmgN_G(U) = Ay^{-1}y'gN_G(U) = y^{-1}Ay'gN_G(U),$$

since $y \in N_G(A)$. This shows that

$$Ay'gN_G(U) = yAgN_G(U) = AygN_G(U).$$

Let $x_j, x_k \in N_G(U)$. If $x_k \in N_{A^g}(U)x_jN_B(U)$ then

$$Aygx_kN_B(U) = yAgx_kN_B(U) = yAgg^{-1}Agx_kN_B(U)$$

$$= yAgg^{-1}Agx_jN_B(U) = Aygx_jN_B(U).$$

If, on the other hand, $Aygx_kN_B(U) = Aygx_jN_B(U)$ then $Agx_kN_B(U) = Agx_jN_B(U)$ and $A^gx_kN_B(U) = A^gx_jN_B(U)$. There exist $a \in A, n \in N_B(U)$ such that $x_k = a^gx_jn$. Then $a^g = x_kn^{-1}x_j^{-1} \in N_G(U)$ and $a^g \in A^g$ imply $a^g \in N_G(U) \cap A^g = N_{A^g}(U)$. Therefore $x_k, x_j \in N_G(U)$ implies $x_k \in N_{A^g}(U)x_jN_B(U)$ such that $N_{A^g}(U)x_kN_B(U) = N_{A^g}(U)x_jN_B(U)$.

We now assume that the modular law holds for $M(g,U)$, e.g. $M(g,U) = AN_G(gUg^{-1}) \cap N_G(A)$. Then from $AmgN_G(U) = AgN_G(U)$ we have $mg = agn$ for some $a \in A$ and $n \in N_G(U)$, and $m = agng^{-1} \in AN_G(U^{g^{-1}})$. If $m \in N_G(A)$ then

$$m \in AN_G(gUg^{-1}) \cap N_G(A) = A(N_G(gUg^{-1}) \cap N_G(A)).$$

This characterizes $M(g,U)$ as the set of those elements m for which $AmgN_G(U) = AgN_G(U)$ holds.

Now let $y, y' \in N_G(A)$ such that $Ay'gN_G(U) = AygN_G(U)$. Then

$$Ay^{-1}y'gN_G(U) = y^{-1}Ay'gN_G(U) = y^{-1}AygN_G(U) = AgN_G(U).$$

This shows that $y^{-1}y' \in M(g,U)$ and $yM(g,U) = y'M(g,U)$. Therefore under this condition the cosets $yM(g,U)$ correspond bijectively to the double cosets $AygN_G(U)$, into which $N_G(A)gN_G(U)$ splits.

□

Of course, by the homomorphism principle fixed points which lie in the same orbit under $N_B(U)$ belong to the same orbit. Therefore we need not split further. Now we have to test whether for such a coset Ag with $AgU = Ag$ the subgroup U is the full stabilizer in B. This holds if and only if $A^g \cap B = U$. Equivalently we could also test whether Ag belongs to some double coset constructed for a prescribed stabilizer V containing U properly. Therefore we have the following result.

3.16 Theorem. *Let $A, B \leq G$, and let R be a set of representatives from the conjugacy classes of subgroups of B. Then*

$$A\backslash G/B = \bigcup \{Ay_i g x_j B | U \in R, N_G(A)gN_G(U) \in N_G(A)\backslash G/N_G(U) \text{ and}$$

$$g^{-1}Ag \geq U, N_{A^g}(U)x_jN_B(U) \in N_{A^g}(U)\backslash N_G(U)/N_B(U) \text{ and}$$
$$(gx_j)^{-1}Agx_j \geq U, y_iM(g,U) \in N_G(A)/M(g,U)\}.$$

The lemma allowed to determine the fixed points of the chosen subgroup U and in an algorithm all fixed points having a bigger stabilizer than U have to be subtracted. By the homomorphism principle any of the remaining fixed points lie in the same double coset if and only if they lie in the same orbit of $N_B(U)$. The given mathematical formulation in the theorem is descriptive and hides several constructive tasks by the set notation.

In a special situation the determination of fixed points by Lemma 3.7 becomes easy. If there is only one conjugacy class of subgroups of A into which the subgroup U can be conjugate in G by some g then by the Frattini-argument $N_G(A) = AN_{N_G(A)}(gUg^{-1})$. This means that the subgroup $M(g,U)$ in 3.7 is just $N_G(A)$. Therefore in this case only a transversal of $N_{A^g}(U)\backslash N_G(U)/N_B(U)$ has to be determined.

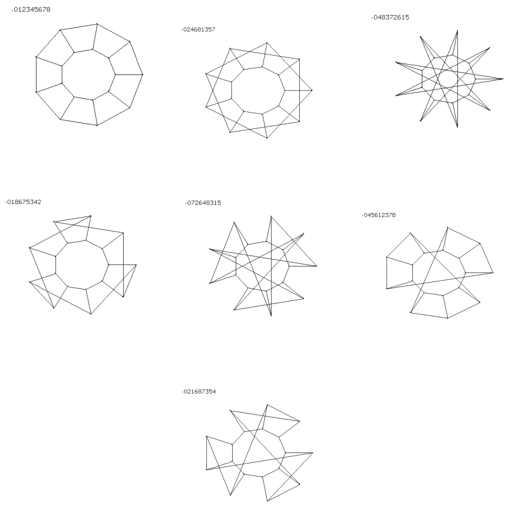
As an example we consider the double cosets of $A = D_9$ with $B = D_9$ in S_9, where we represent D_9 as generated by $c = (1,2,3,4,5,6,7,8,9)$ and $t = (1,2)(3,9)(4,8)(5,7)$. Then A contains just one conjugacy class of subgroups of each isomorphism type occurring. Since $A = B$ we can always choose $g = id$ for one element conjugating U into A.

In the first case we consider $U = B$. The index of D_9 in its normalizer is 3 such that we find 3 double cosets with stabilizer B of an appropriate coset of A in G. A set of representatives is formed by id, $(2,3,5,9,8,6)(4,7)$, and $(2,5,8)(3,9,6)$.
As the next case we consider $U =< c >$ where the normalizer coincides with that of D_9. Then the fixed points of U are all fixed points of B such that there are no double cosets of this stabilizer type.
The case $U = S_3$ where $S_3 =< c^3, t >$ is isomorphic to the non abelian group of order 6 is more interesting. U has index 6 in its normalizer in G while $U = N_B(U) = N_A(U)$. The transversal of double cosets in $N_G(U)$ is just a transversal of the cosets of U in its normalizer $N_G(U)$ in this case. Thus we obtain 6 fixed points of U of which 3 are already fixed points of B. The remaining fixed points give the double cosets of stabilizer type S_3 and may be represented by $(3,9)(4,7)(5,8), (2,8)(4,7)(6,9)$, and $(2,5)(3,6)(4,7)$.
The case $U = C_3$ where $C_3 =< c^3 >$ behaves totally different. Here U is a normal subgroup in B and the index of $B = N_B(U)$ in $N_G(U)$ is 18. We have to determine a set of representatives for the double cosets $AxN_B(U)$ which are contained in $N_G(A)idN_G(U) = N_G(U)$. Since we will have to eliminate those fixed points Ax of U with a larger stabilizer than U, we integrate the determination of the stabilizer into the determination of the fixed points. We apply the homomorphism principle in determining first the double cosets $N_G(A)xB$ and splitting these afterwards. These double cosets can be classified by

a prescribed stabilizer. There is only one class with stabilizer B which splits just into the 3 fixed points of B we found above. At least one coset Ag which is a fixed point of S_3 lies outside of $N_G(A)$, since we found that all those inside were the fixed points of B. Therefore there must be one orbit with stabilizer S_3 of some coset such that the orbit length is 3. This orbit splits into the three orbits with stabilizer S_3 that we described above. Since the index of $N_G(A)$ in $N_G(U)$ is 6, there remain two points which must lie in one orbit with stabilizer C_9. In the splitting step we find that the only stabilizer contained in C_9 of an orbit up to length 6 is U. So we obtain only one double coset with stabilizer C_3 of some point, a representative is $(2,3)(4,7)(5,9)(6,8)$.
The last non trivial stabilizer is a cyclic group of order 2, we choose $U =< t >$. As in the case of S_3 also here $U = N_B(U)$. The index in $N_G(U)$ is 192 giving as many fixed points. The only minimal overgroup in B is S_3 with 6 fixed points. Thus $< t >$ is the stabilizer of 186 points. We do not list representatives for these 186 double cosets.
A simple arithmetic shows that the remaining cosets fall into 1026 double cosets each consisting of 18 cosets of A in G.
A way to obtain representatives in the latter two cases is to apply the Leiterspiel in a simple case. For that purpose we first determine the double cosets of $Q =< N_G(C_3), (4,7)(5,8) >$ and B in G classified by stabilizer as before and then split these into the double cosets of A and B in G. The index of Q in G is 280, A is a subgroup of index 72 of Q. The cosets fall into double cosets of the following stabilizer type: 1 of type D_9, 3 of type S_3, 12 of type $< t >$, and 9 with trivial stabilizer.
Since the index of A in Q is 72, each of the 9 double cosets of length 18 splits into 72 double cosets of this length of A and B in G. For the splitting of the double cosets of length 9 we have to consider the action of $< t >$ on the set of cosets Axh into which a coset Qh fixed by $< t >$ splits. In each of the 12 cases we get again 12 fixed points of $< t >$ and 30 orbits of length 2.
Thus, we have already $9 * 72 + 12 * 30 = 1008$ double cosets of length 18. The remaining 18 double cosets of this length come from the action of S_3 on the set of 72 cosets Axh contained in a representative coset $Q * h$ with stabilizer S_3. In each of the 3 cases we get 6 orbits of length 6, i.e. double cosets of trivial stabilizer type. We remark that also in this approach Lemma 3.7 may be applied with the smaller group U instead of B.

All computations for this example were done using the Cayley system [14].

The example may easily be generalized to bigger dihedral groups. It demonstrates a constructive approach to cycle permutation graphs [39] which may be described by identifying the beads of two necklaces bijectively. In [39] the number of isomorphism types is determined by the Cauchy-Frobenius Lemma which is completely non constructive. One can consider the necklaces as graphs with beads as vertices and edges between vertices of which the beads are neighbours in the necklace. The identifications can be interpreted as connections between the vertices of the two graphs. There result trivalent graphs. In order to avoid irregular isomorphisms we distinguish the vertices of the two necklace graphs by white and black labels respectively. Then the above described representatives give the isomorphism types of graphs with the given labeling. For big stabilizers we find



the following graphs. First row has stabilizer D_9, second row has stabilizer S_3, and the last graph belongs to the stabilizer C_3.
The Theorem has interesting applications not only for algorithms but also for theoretical purposes. We mentioned already the special case that U is conjugate to subgroups of only one conjugacy class of A. If in addition $A = N_G(A)$ then the following corollary can be applied.

3.17 Corollary. *Let $U \le A \le G, A = N_G(A)$, and for each $g \in G$ such that $U^g \le A$ there exist some $a \in A$ with $U^g = U^a$. Then there are exactly $|N_G(U)|/|N_A(U)|$ subgroups in the conjugacy class of A containing U.*

For a proof apply the theorem with $B = U$ and note that by the assumptions only one double coset $N_G(A) id N_G(U) = AN_G(U)$ has to be split.

For example $PSL(3,3)$ contains a Sylow-13-subgroup P of S_{13}. If U is the normalizer of P in $PSL(3,3)$ then $|U| = 39$ and $|N_{S_{13}}(P)| = |Hol(C_{13})| = 156$. Therefore U is contained in exactly $156/39 = 4$ subgroups conjugate to $PSL(3,3)$. Similarly the Mathieu group M_{24} is a maximal subgroup of S_{24} and contains only one conjugacy class of subgroups isomorphic to $PSL(2,23)$. Since the normalizer of $PSL(2,23)$ in S_{24} is $PGL(2,23)$, there are exactly $|PGL(2,23)|/|PSL(2,23)| = 2$ subgroups conjugate to M_{24} and containing $PSL(2,23)$.
Double cosets appear in many applications. We discuss two general situations and some applications.

3.18 The Fundamental Lemma *Assume that the group G acts transitively on the set Ω and let U be a subgroup of G. Then the mapping*

$$\theta : (\omega^g)^U \mapsto N_G(\omega) g U$$

from the set Ω/U of orbits of U on Ω to the set $N_G(\omega)\backslash G/U$ of $(N_G(\omega), U)$-double cosets in G is a bijection, where $\omega \in \Omega$ is an arbitrarily chosen fixed reference point in Ω, and $N_G(\omega)$ denotes the stabilizer of that point ω.

Proof: We recall the bijection

$$\omega^g \mapsto N_G(\omega)g$$

between the points in a G-orbit and the right cosets of the stabilizer $N_G(\omega)$ of ω in G. In this bijection a point $(\omega^g)^u = \omega^{gu}$, for some $u \in U$, is mapped onto the coset $N_G(\omega)gu$. Thus, the full orbit ω^{gU} corresponds to the set of cosets $N_G(\omega)gu$ where $u \in U$. The union of the latter cosets forms just the double coset $N_G(\omega)gU$, which shows that θ is a bijection.

□

3.19 The Gluing Lemma *Let ω_1 and ω_2 be two isomorphic objects and $f_0 : \omega_1 \to \omega_2$ be a fixed reference isomorphism. Then the set $Iso(\omega_1, \omega_2)$ of all isomorphisms is obtained by appending all automorphisms of ω_2 to f_0. Let a group A act on ω_1 and a group B act on ω_2 as automorphism groups. Then $A \times B$ acts on $Iso(\omega_1, \omega_2)$ by $f^{(a,b)} = a^{-1}fb$ for all $f \in Iso(\omega_1, \omega_2)$. The mapping*

$$\phi : A\backslash Iso(\omega_1, \omega_2)/B \to f_0(f_0^{-1}Af_0)\backslash Aut(\omega_2)/B$$

sending the orbit of $f_0\sigma$ under $A \times B$ to $f_0(f_0^{-1}Af_0)\sigma B$ is a bijection.

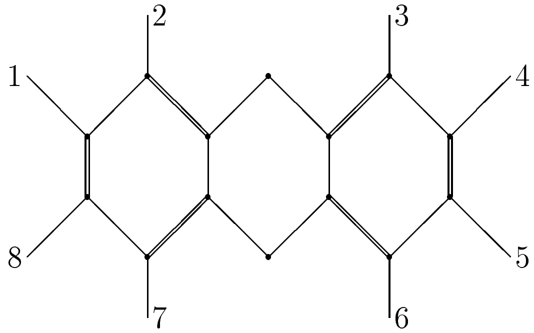
Proof: If f_1 is another isomorphism from ω_1 to ω_2 then $\rho = f_0^{-1}f_1$ is an automorphism of ω_2 such that $f_0\rho = f_1$. Also for each automorphism ρ of ω_2 the mapping $f_0\rho$ is an isomorphism from $|omega_1$ to ω_2. Now let $f_1 = f_0\rho_1$ and $f_2 = f_0\rho_2$. Then there exists a pair $(a, b) \in A \times B$ such that $f_1^{(a,b)} = f_2$ if and only if $f_0\rho_2 = a^{-1}f_0\rho_1 b$ and

$$\rho_2 = f_0^{-1}a^{-1}f_0\rho_1 b \in (f_0^{-1}Af_0)\rho_1 B.$$

□

3.20 Applications to chemistry We start with an example from mathematical chemistry, where both lemmata can be applied. The problem is the construction of all the 22 isomers of dioxine which has the chemical formula $C_{12}O_2H_4Cl_4$. All dioxines have a common skeleton of three 6-rings as it is shown in Figure 6.
All the vertices of degree four represent carbon atoms C and the two vertices of degree two stand for oxygene atoms O. The skeleton has, as it is easy to see, eight free places at the end of eight edges. These places are numbered 1 to 8, we call these places *free valences.* According to the chemical formula $C_{12}O_2H_4Cl_4$, these free valences are occupied by the remaining four hydrogen atoms H and the four chlorine atoms Cl. The connectivity isomers of dioxine are therefore formed by distributing these remaining eight atoms over the free valencies in all the possible and essentially different ways. It is

Figure 6: Skeleton of dioxine



clear that *essentially different* refers to the symmetry group of the skeleton. Thus any bijection

$$\{1,2,3,4,5,6,7,8\} \to \{H,H,H,H,Cl,Cl,Cl,Cl\}$$

from the set of free valences to the multiset of atoms mathematically describes a connectivity isomer of dioxine, but not all of them are essentially different, since symmetry operations of the skeleton map a molecule onto an equivalent one and also any permutation of the atoms of the same kind does not change the connectivity isomer. Thus we have two groups: A acting on the set of the eight free valencies by applying symmetry operations of the skeleton and B acting on the multiset of atoms by permuting all atoms of the same element among themselves in all possible ways. Using this and the Gluing Lemma, we can easily solve the classification problem in question, obtaining the essentially different maps that represent the connectivity isomers. It is easy to see that the group A is isomorphic to $C_2 \times C_2$ and B is isomorphic to $S_4 \times S_4$. Using a reference bijection which we keep fixed we get a set of representatives of the different types by composition with representatives from the double cosets of the corresponding subgroups of S_8. There are exactly 22 such double cosets, they yield the isomers shown in Figure 7.

Another interpretation of this problem reads as follows: The symmetric group S_8 acts transitively on the set of mappings $\{1,2,3,4,5,6,7,8\} \to \{Cl,H\}$ having exactly 4 preimages of both H and Cl. The stabilizer N of a particular reference mapping ϕ_0, mapping the first four numbers to Cl and the latter to H, say, is the direct product of the symmetric group on $\{1,2,3,4\}$ by the symmetric group on $\{5,6,7,8\}$. By the Fundamental Lemma, the double cosets of this stabilizer with the permutation group $C_2 \times C_2$ induced by the automorphism group of the skeleton on the set of places correspond bijectively to the classes of mappings describing the different isomers.

This example may also be used to demonstrate another technique which is useful in construction problems. ¿From the index of N in S_8 we find that altogether 70 mappings have to be classified with respect to the action of a group of order 4. Each orbit

Figure 7: The 22 isomers of dioxin

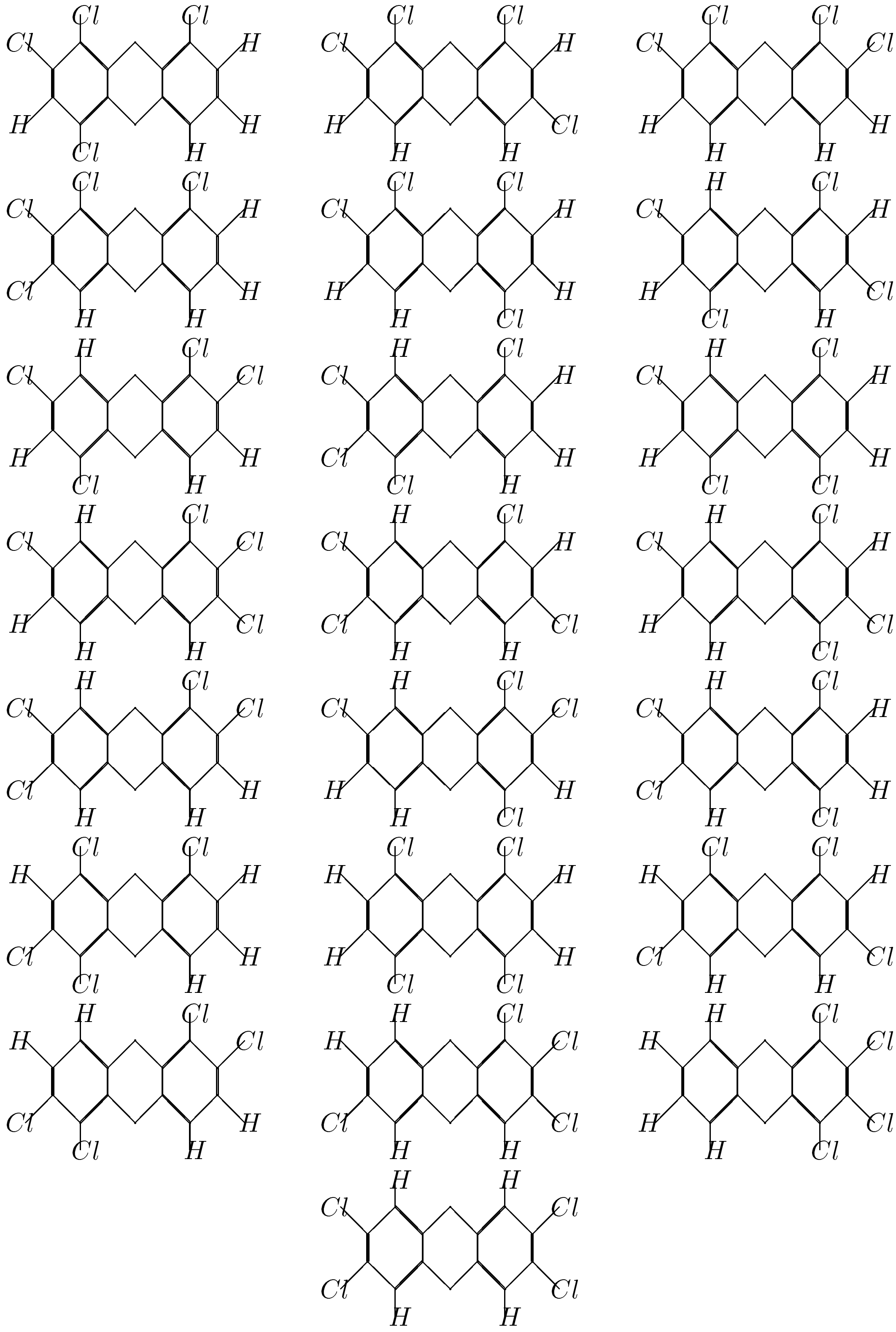


Figure 8: *Insert a Subgraph*

corresponds bijectively to the cosets of the stabilizer of a chosen representative. Thus we immediately know the length of the orbit from the order of the stabilizer . Since the stabilizer is just formed by the automorphisms of the corresponding dioxine, we can easily find this order. Thus the strategy is to construct new representatives and to check if the sum of the known orbit lengths already amounts to the overall number, which is 70 in our problem. It is an easy exercise to check that the 22 isomers listed in figure I form a complete set of representatives in our case. For the general case, this technique is described by the following result.

3.21 The Class Equation *Let a finite group G act on a finite set Ω. Let Γ be a set of points from pairwise different orbits. If*

$$1/|G| = 1/|\Omega| \sum_{\gamma \in \Gamma} 1/|N_G(\gamma)|$$

then Γ is a full set of orbit representatives.

Proof: By the discussion above we have for a full set of representatives Γ that

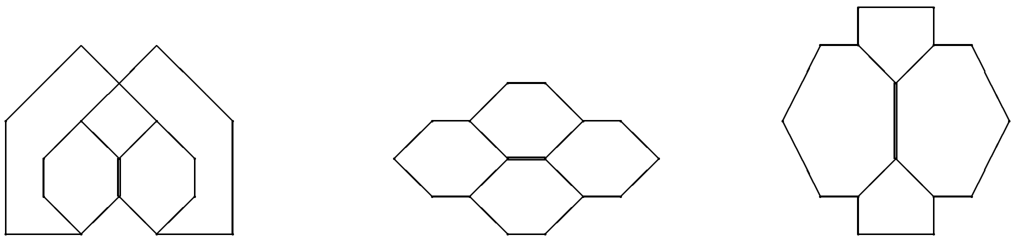
$$|\Omega| = \sum_{\gamma \in \Gamma} |G|/|N_G(\gamma)$$

such that a simple division yields the claimed equation. Clearly any missing summand causes that the equation cannot hold.

□

Here is another application of the Gluing Lemma to mathematical chemistry. It amounts to the problem of replacing a vertex of a graph by a subgraph. To the vertices in question we assign - like in the above case - free bonds which have to match with the corresponding bonds in the subgraph.
The central vertex of the graph on the right hand side of Fig. 8 has to be replaced by the subgraph on the left hand side. In order to do that (in all the possible and essentially different ways!) we have to identify the free bonds 1,2,3,4 with the edges named a,b,c,d in the graph in a suitable way. There are $4! = 24$ possible bijections between these two sets. We obtain equivalent results if we replace the subgraph or the graph by an

Figure 9: Subgraph inserted



isomorphic copy. Thus we let the two automorphism groups act on the set of bijections as it is described in the Gluing Lemma. The group acting on $\{1,2,3,4\}$ is the dihedral group D_4 which may be described as a wreath product of C_2 by C_2. Since this group has order 8, it has 3 cosets in S_4. The other group, induced on the set $\{a,b,c,d\}$, is the Kleinian 4-group of order 4. The Kleinian 4-group is contained in the dihedral group and it is a normal subgroup of S_4. Therefore the double cosets of both groups in S_4 reduce to the cosets of D_4 in S_4. The corresponding results are the graphs in Fig. 9.

4 Sims Chains and Automorphism Groups

The last situation, where we want to discuss homomorphisms must usually first be created. But then it proves to be a very powerful instrument.
If G acts on Ω then G also acts on $\Omega \times \Omega$, and generally on Ω^k for any $k \in \mathbf{N}$ by

$$(\omega_1, \ldots, \omega_k)^g := (\omega_1^g, \ldots, \omega_k^g), \text{ for } (\omega_1, \ldots, \omega_k) \in \Omega^k,$$

i.e. componentwise.
In such a situation we can designate any $T \subseteq \{1, \ldots, k\}$, $|T| = t$, and project

$$\pi_T \colon \Omega^k \to \Omega^t,\ (\omega_1, \ldots, \omega_k) \mapsto (\omega_{i_1}, \ldots, \omega_{i_t}),$$

where $T = \{i_1, i_2, \ldots, i_t\}$ and $i_1 \leq i_2 \leq \ldots \leq i_t$. Any such projection is a $G-$homomorphism, s.t. the homomorphism principle applies. We have by lemma 1.6

$$\begin{array}{rrcl} & (\omega_1, \ldots, \omega_k) & \sim & (\omega_1', \ldots, \omega_k') \\ \Leftrightarrow & \exists g_1 : (\omega_{i_1}, \ldots, \omega_{i_t})^{g_1} & = & (\omega_{i_1}', \ldots, \omega_{i_t}') \wedge \\ & \exists g_2 \in N_G((\omega_{i_1}, \ldots, \omega_{i_t})) & = & C_G(\{\omega_{i_1}, \ldots, \omega_{i_t}\}) \\ & (\omega_1, \ldots, \omega_k)^{g_2} & = & (\omega_1', \ldots, \omega_k')^{g_1^{-1}}. \end{array}$$

We can approach the orbits of G on Ω^k by a series of projections onto some components. If we know orbits on such projection sets, the orbits in the preimage sets can be obtained by only looking at stabilizers and their action. This important observation is due to C.

Sims (1972) [67], who used the approach for handling large permutation groups. He chose as projections

$$\Omega^k \to \Omega^{k-1} \to \Omega^{k-2} \to \ldots \to \Omega$$

the mappings which in each case forget the last component.

4.1 Proposition. (Sims) *Let $(\omega_1, \ldots, \omega_k), (\omega'_1, \ldots, \omega'_k) \in \Omega^k$. Then there exists some $g \in G$ such that*

$$(\omega_1, \ldots, \omega_k)^g = (\omega'_1, \ldots, \omega'_k)$$

if and only if

$$\omega_1^g = \omega_1^{g_1}$$

for some $g_1 \in G$ and for all $i > 1$ there are elements g_i such that

$$N_G((\omega_1, \ldots, \omega_i))g_i \in N_G((\omega_1, \ldots, \omega_i)) \backslash N_G((\omega_1, \ldots, \omega_{i-1}))$$

$$\textit{and } \omega_i^{g g_1^{-1} \ldots g_{i-1}^{-1}} = \omega_i^{g_i} .$$

A group G acts faithfully on Ω if only the identity element of G fixes each point of Ω. We now assume G to act faithfully. Then we can arrange the elements of Ω in a sequence $(\omega_1, \ldots, \omega_{|\Omega|})$ such that

$$N_G((\omega_1, \ldots, \omega_{|\Omega|})) = C_G(\Omega) = \{id\}.$$

Then each element of G forms just one coset of $N_G((\omega_1, \ldots, \omega_{|\Omega|}))$ in G, i.e. the bijection of the cosets and the orbit of $(\omega_1, \ldots, \omega_{|\Omega|})$ gives in fact a bijection between the elements of G and the image points of $(\omega_1, \ldots, \omega_{|\Omega|})$.
If $\pi \in S_\Omega$ is any given permutation we can form $(\omega'_1, \ldots, \omega'_{|\Omega|}) = (\omega_1^\pi, \omega_2^\pi, \ldots, \omega_{|\Omega|}^\pi)$ and decide by the above proposition whether

$$(\omega'_1, \ldots, \omega'_{|\Omega|})^g = (\omega_1, \ldots, \omega_{|\Omega|})$$

for some $g \in G$. If and only if such a g exists the given π lies in G, i.e. $\pi = g^{-1}$.
Besides this test for $\pi \in G$ we can easily tell the order of G :

$$\begin{aligned} |G| &= \prod_{i=1}^{|\Omega|} |N_G((\omega_1, \ldots, \omega_i)) \backslash N_G((\omega_1, \ldots, \omega_{i-1}))| \\ &= \prod_{i=1}^{|\Omega|} |\omega_i^{N_G((\omega_1, \ldots, \omega_{i-1}))}| \end{aligned}$$

where $N_G((\omega_1, \ldots, \omega_{i-1})) := G$ for $i = 1$.
The first version is just the application of Lagrange's formula $|G| = |U| \cdot |U \backslash G|$ for any $U \leq G$ to the sequence of stabilizers. The second equation results from the bijection between orbit and right cosets of a point stabilizers. We shall later see that in some

cases one can describe the orbits without listing a full set of representatives such that the second formula is easier to evaluate.
A series

$$G_0 = N_G(()) > G_1 = N_G((\omega_1)) > \ldots > G_{|\Omega|} = N_G((\omega_1, \ldots, \omega_{|\Omega|}))$$

together with a description of the orbits of ω_i under G_{i-1} for all i is called a *Sims Chain.* It is important to notice that each element $g \in G$ has a unique representation $g = g_{|\Omega|} g_{(|\Omega|-1)} \ldots g_1$, where each g_i is one of the chosen right coset representatives for

$$N_G((\omega_1, \ldots, \omega_i)) \backslash N_G((\omega_1, \ldots, \omega_{i-1})).$$

Thus it is easy to run through all elements. We also note that the *completeness of all sets of representatives already implies that the set of elements of this form is multiplicatively closed.*

While in our examples considered so far the acting groups were well known, in general the situation is more complicated. As demonstrated with the example of the football or the dodecahedron there may be some basic structure to which some new components should be added. Then the full automorphism group G of the basic structure has an induced action on the set of different possibilities to add new components. We can find in such a situation by the methods presented here or some different approach like orderly generation [59] orbit representatives of G to solve the isomorphism problem.

Therefore we now discuss techniques for *constructing the automorphism group* of some finite discrete structure. Two papers that had a strong influence on our approach are [48], using orbits of subgroups to keep the number of branches in a backtracking small, and [52] who used iterated classification to reveal consequences of choices taken within the backtracking strategy.

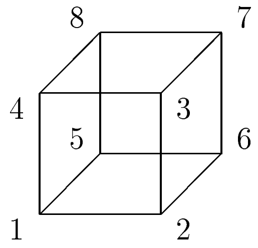
We start by showing how a Sims chain for an automorphism group can be constructed in a simple version.

For this purpose we use a geometrical object for which automorphisms can easily be described from our imagination. We concentrate on the description of the group, especially the fact that the full automorphism group is described.
The algorithm starts with $N_G((\omega_1, \ldots, \omega_{|\Omega|}) = \{id\}$ and stepwise computes

$$N_G((\omega_1, \ldots, \omega_{i-1})) \text{ from } N_G((\omega_1, \ldots, \omega_i)) \text{ for } i = |\Omega|, |\Omega| - 1, \ldots, 1,$$

where $N_G(()) = G$ for $i = 1$. By 4.1 it suffices to give representatives for the right cosets of $N_G((\omega_1, \ldots, \omega_i))$ in $N_G((\omega_1, \ldots, \omega_{i-1}))$ in each step. Such a representative corresponds by the above bijection to a point ω_j from the orbit of ω_i under $N_G((\omega_1, \ldots, \omega_{i-1}))$. We thus only need to find one appropriate automorphism fixing $\omega_1, \ldots, \omega_{i-1}$ and mapping ω_i onto ω_j, if it exists.

4.2 Example. *The automorphism group of a cube.*
For the sequence $(\omega_1, \ldots, \omega_8)$ we choose (1,2,3,4,5,6,7,8). We regard the cube as a rigid body such that no reflection is allowed as an automorphism.



First we see that fixing 1 and 2 keeps the whole body fixed such that $N_G((1,2)) = \{id\}$. Now we prescribe $\pi(1) = 1$ and look for the possible images of point 2. Of course. $\pi(2)$ must be a neighbor of $\pi(1) = 1$. Thus only $\pi(2) \in \{2,4,5\}$ have to be considered. The identity map, a rotation by 120^o with fixed points 1 and 7, and its square give the desired representatives.
The next step prescribes the image point j of 1. For each choice of j *only one* mapping has to be found. It is easily seen that the rotations

$$\rho = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 4 & 1 & 2 & 3 & 8 & 5 & 6 & 7 \end{pmatrix}$$

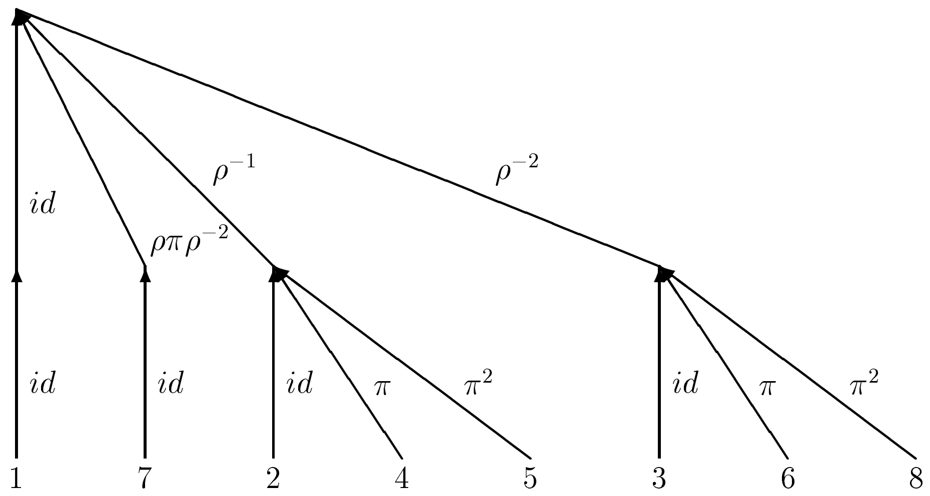
and

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 5 & 1 & 4 & 8 & 6 & 2 & 3 & 7 \end{pmatrix}$$

only need to be combined in different powers to map 1 onto each point. We obtain therefore 8 different permutations. Since now no further step is possible, the whole automorphism group is described. We find that its order is $3 \cdot 8 = 24$ and in fact the isomorphism type of this group may be recognized as that of S_4, the symmetric group on 4 letters. $\diamond$

Like in this example the stabilizer of only a few points may already be trivial. After C. Sims we call $(\omega_1, \ldots, \omega_k)$ *a base for* G if $N_G((\omega_1, \ldots, \omega_k)) = \{id\}$. By the above analysis G acts regularly on the orbit of a base.
Instead of a representation of a Sims chain by one representative from each right coset of $N_G((\omega_1, \ldots, \omega_i))$ in $N_G((\omega_1, \ldots, \omega_{i-1}))$, for each i, M. Jerrum [32] found a description which needs much less space.
The reason why we needed the representatives was that we wanted to decide whether for a given g there exists some $\pi_i \in G_{i-1} = N_G((\omega_1, \ldots, \omega_{i-1}))$ such that $\omega_i^{\pi_i} = \omega_i^g$. Now suppose we know all orbits of G_i on Ω for some fixed i. Then $\omega = \omega_i^g$ lies in some orbit $\omega^{G_{i-1}}$. This orbit is completely contained in $\omega_i^{G_{i-1}}$ if only one point $\omega' \in \omega^{G_{i-1}}$ can be reached from ω_i by some $\pi_i \in G_{i-1}$. It thus suffices to store only one permutation $\pi_i \in G_i$ for the whole orbit ω^{G_i} such that $\omega_i^{\pi_i} \in \omega^{G_{i-1}}$. Assuming ω is a distinguished representative of its orbit $\omega^{G_{i-1}}$ and ω can be reached from each point of that orbit we

only have to add the information that $\omega^{\pi_i^{-1}} = \omega_i$. Then from each point of $\omega^{G_{i-1}}$ we can reach ω_i.
If we look at this iterative procedure from a fixed point $\delta \in \Omega$, we see that δ has no attached permutation as long as δ is a distinguished representative. Only once δ can get a permutation attached to it, mapping δ to some other representative. In this way *each point may obtain only one assigned permutation* such that only $|\Omega|$ permutations may occur in the description of the group.
For our example we first consider the orbits of the subgroup generated by the rotation π by 120^o.
The orbits are $\{1\}, \{2,4,5\}, \{3,6,8\}, \{7\}$. We choose the smallest numbers as distinguished representatives and note $4^\pi = 2, 5^{\pi^2} = 2, 6^{\pi^2} = 3, 8^\pi = 3$. To describe the orbit of 1 under G in the next step we now only need to know permutations mapping 2,3,7 onto 1. The rotation $\rho = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 1 & 6 & 7 & 8 & 5 \end{pmatrix}$ fuses all four orbits. Therefore the point 1 can be mapped onto all other points. The former orbit representatives 2,3,7 are mapped onto 1 by the permutations $\rho^{-1}, \rho^{-2}, \rho\pi\rho^{-2}$ respectively. Thus, for a membership test we only need to know π and ρ and some words in these permutations.



The corresponding data structure is called a *labeled branching*. It can be interpreted as a special form of the Cayley Action Graph, where we should enlarge the given set of generators by the words we computed to label the fusion of orbits. This data structure belongs to the Union-Find data structure family of computer science which allows a fast handling of disjoint sets. Each set has a distinguished representative which is considered as the root of a tree. All elements of the sets form the nodes of the tree. Contrary to usual directed trees here arcs connecting two nodes are directed from the son to the parent node. This has the advantage that *a node may have many sons* and then the path lengths from any node to the root of its tree will become short. Union of two disjoint

sets can be done by adding only one arc from one of the former roots to the other root. One can add to this data structure a second family of pointers. These pointers link all nodes of one tree to a list. The union of two trees only needs a constant amount of additional time to keep this list if pointers to the end of the lists are maintained. With this modification one can run through all points of the set which belong to nodes of the same tree. For a detailed analysis of data structures for permutation groups see [4].
For a graphical representation of a permutation group in this form we may take the points as nodes and introduce an arc from ω_i to ω_j labeled by a permutation π if $\omega_i^\pi = \omega_j$.
For any chain of subgroups $G = G_0 > G_1 > \ldots > G_k = \{id\}$ we can represent the fusion of orbits of G_i to orbits of G_{i-1} for all i by such a labeled branching.
To each G_i there corresponds a forest, i.e. a set of subtrees, which describes the orbits of G_i. For $i = k$ the forest consists of roots only. If $G_{k-1} =< \pi_1^{(k-1)}, \ldots, \pi_{r_{k-1}}^{(k-1)} >$ then each orbit of G_{k-1} is represented by a tree of height ≤ 1 with arc-labels $\pi_j^{(k-1)}$.
If the G_i-orbits are already presented by trees of height $\leq k - i$ then the corresponding trees for G_{i-1} are obtained by adding arcs for permutations in G_{i-1} mapping the former roots onto some point not in the existing tree. Thus the path from any node to the root of its tree may grow only by one arc, i.e. the height may only grow by 1.
It is not true in general that any chain of subgroups can be fully described by fusions of orbits. For example in the symmetric group S_n on the set of points $\{1, \ldots, n\}$ the cyclic subgroup $C_n = \langle(1, \ldots, n)\rangle$ is already transitiv, such that the chain $G_0 = S_n > G_1 = C_n > 1$ is not appropriately represented.
But for certain chains we can rely on a theorem of Wielandt which is a stronger version of the bijection between an orbit and the cosets of the stabilizer of one point. First we have to introduce the notion of a block.

4.3 Definition. *Let a group G act on some set Ω. A subset $\Delta \subseteq \Omega$ is a* block *of G on Ω if for each $g \in G$ $\Delta \cap \Delta^g = \Delta$ or $\Delta \cap \Delta^g = \emptyset$.*

Let $\omega \in \Omega$ and G act transitively on Ω. Then any subgroup U containing $C_G(\{\omega\})$ defines a set $\Delta = \omega^U = \{\omega^u \mid u \in U\}$. If $\Delta^g \cap \Delta \neq \emptyset$ then there exists $u_1, u_2 \in U$ such that $\omega^{u_1 g} = \omega^{u_2}$ and $u_1 g u_2^{-1} \in C_G(\{\omega\}) \subseteq U$. This implies $g \in U$ and $\Delta^g = \Delta$. Now let $\omega \in \Delta, \Delta$ block. From $\omega^g = \omega$ we obtain $\Delta^g \cap \Delta \neq \emptyset$ and therefore $\Delta^g = \Delta$, such that $C_G(\{\omega\}) \subseteq N_G(\Delta)$. We obtain the following result.

4.4 Theorem. (Wielandt) *Let a group G act transitively on some set Ω. Let $\omega \in \Omega$. Then $\Delta \mapsto N_G(\Delta)$ defines an isomorphism between the lattice of blocks containing ω and the sublattice of the subgroup lattice of G of those subgroups which contain $C_G(\{\omega\})$.*

4.5 Corollary. *If a group acts on some set Ω such that $U \subseteq G$ is just the* stabilizer of one point, *then any chain $G = G_0 > G_1 > \ldots > G_k = U$ of subgroups can be uniquely presented by a labeled branching for this chain.*

We apply this corollary to $G_{i-1} = N_G((\omega_1, \ldots, \omega_{i-1}))$ as well as to the point stabilizer $G_i = N_G((\omega_1, \ldots, \omega_i))$ in G_{i-1}. Then any chain of subgroups between G_i and G_{i-1} corresponds to a chain of blocks. The labeled branching then describes how these blocks considered as orbits of the corresponding block normalizers are fused to form new blocks. Thus, a labeled branching can be used to present any chain of subgroups which refines a chain of point stabilizers. In order to keep intermediate steps small in our algorithms we are usually interested in *fine chains*, i.e. chains which are not refinable. For an implementation of this idea see [41].
If Δ is a block containing ω then the mapping

$$\varphi\colon \omega^G \to \Delta^G\colon \omega^g \mapsto \Delta^g$$

is a homomorphism with respect to the action of G on both orbits. On the other hand each homomorphism $\varphi\colon \Omega_1 \to \Omega_2$ with respect to a group action defines blocks Δ by $\Delta = \varphi^{-1}(\omega)$ for $\omega \in \Omega_2$. Therefore the Theorem of Wielandt also describes the homomorphisms for transitive group actions.

For a group G given by a set S of generators there is an algorithm by Atkinson [2] which allows to find the smallest block Δ of G containing two given points $\omega_1, \omega_2 \in \Omega$. Basically the algorithm not only constructs Δ but also all blocks Δ^g for $g \in G$. Since these blocks form an orbit of G on the set of all blocks, we can at the same time find a set of Schreier generators for $N_G(\Delta)$.
The tool for Atkinsons algorithm is the same data structure as in the labeled branching, i.e. the *Union-Find* data structure. The blocks are built by fusing disjoint subsets which have been detected by a test for the block-property. As above these subsets are represented by a tree and a fusion of two trees only needs one additional arc from one of the roots to the other.

The starting point is the partition formed by $\{\omega_1, \omega_2\}$ and each $\{\omega_i\}$ for $\omega_i \in \Omega - \{\omega_1, \omega_2\}$. In this step choose ω_1 as the root of the tree for $\{\omega_1, \omega_2\}$ and all other trees consist only of a root. The algorithm builds up the orbit of Δ by iteratively applying all generators $g \in S$ to subsets Λ^w, where $\{\omega_1, \omega_2\} \subseteq \Lambda \subseteq \Delta$ and w is some word in S. If Λ^{wg} intersects two subsets Ω_1, Ω_2 of the present partition these subsets have to be fused by the definition of a block. We obtain a new word wg such that Δ^{wg} contains Ω_1 and Ω_2. If there are already words w_i such that $\Omega_i \subseteq \Delta^{w_i}$ then we know that $\Delta^{wg} = \Delta^{w_i}$ and wgw_i^{-1} is a Schreier generator of $N_G(\Delta)$ for each such i. Also if Δ^{wg} is already contained in some Ω_1 and $\Omega_1 \subseteq \Delta^{w_1}$ then wgw_1^{-1} is a Schreier generator of $N_G(\Delta)$. The algorithm stops if all subsets in the partition which can be reached by applying some word w in S to Λ yield no more fusions.

This idea can be implemented by keeping a list L of pairs of nodes $\{\alpha, \beta\}$ such that one node is the father of the other node in some tree and still each $g \in S$ has to be applied to $\{\alpha, \beta\}$. For each tree which contains at least two nodes there is a word $word(\rho)$ attached to the root ρ of the tree such that $\rho \in \Delta^{word(\rho)}$. In the beginning L consists just of (ω_1, ω_2), there is one tree with at least two nodes, the tree containing ω_1 and ω_2. The word attached to the root ω_1 is the empty word representing the identity element of G.

A loop is performed until the list L is empty. An iteration step of the loop consists in taking a pair $\{\alpha, \beta\}$ out of L and applying each $g \in S$ to the pair. In order to decide whether α^g and β^g belong to the same tree the roots $\rho_1 = root(\alpha^g)$ and $\rho_2 = root(\beta^g)$ are determined.

If $\rho_1 \neq \rho_2$ then the two trees have to be fused. In order to keep the height of the trees small the tree with the lowest height becomes a subtree of the other by adding an arc from its root to the root of the other. Then the pair (ρ_1, ρ_2) is added to L.

If $\rho_1 = \rho_2$ then no fusion is needed.

We now consider Schreier generators. Each time a pair (ρ_1, ρ_2) is inserted into L this pair describes two former roots now becoming one root and one of its sons. Thus, if (α, β) is taken out of L α is the father of β and both elements lie in $\Delta^{word(\alpha)}$. This implies that $\rho = root(\alpha^g) \in \Delta^{word(\alpha)g}$. Thus, if already $\rho \in \Delta^{w_1}$ for some word w_1 detected earlier then $word(\alpha)gw_1^{-1}$ is a Schreier generator of $N_G(\Delta)$. This applies always to the case $\rho_1 = \rho_2$ in the test. If $\rho_1 \neq \rho_2$ then the new root ρ gets $word(\alpha)g$ attached to it and only if ρ was already labeled by some word w_1 we get a Schreier generator.

4.6 Algorithm (Atkinson's Algorithm).

Input:set S *of generators of a permutation group* G *on a set* Ω,
  ω_1, ω_2 *two different points from* Ω.
Initialize:
  $Schreier - generators = \emptyset$,
  $L = ((\omega_1, \omega_2))$,
  $word(\omega_1) = ()$,
  $father(\omega_2) = \omega_1, height(\omega_1) = 1$,
  $forall \omega \in \Omega - \{\omega_1\}$
  $word(\omega) = Null, height(\omega) = 0$.
Repeat [at this point all pairs in L *still have to be tested]*
  take (α, β) *out of* L
  [α *is the father of* β, $word(\alpha) \neq NULL$ *]*
  for each $g \in S$ *do*
    $\rho_1 = root(\alpha^g)$; $\rho_2 = root(\beta^g)$;
    if $word(\rho_1) \neq NULL$ *then*
    insert $word(\alpha)gword(\rho_1)^{-1}$ *into* $Schreier - generators$.
    if $word(\rho_2) \neq NULL$ *then*
    insert $word(\alpha)gword(\rho_2)^{-1}$ *into* $Schreier - generators$.
    if $\rho_1 \neq \rho_2$ *then*
    [thispairhas to be inserted into L, *and the trees must be fused]*
      if $height(\rho_1) \leq height(\rho_2)$ *then*
        define an arc from ρ_1 *to* ρ_2
        insert (ρ_2, ρ_1) *into* L
        if $word(\rho_2) = NULL$ *and* $word(\rho_1) = NULL$ *then*
        $word(\rho_2) = word(\alpha)g$;
        if $word(\rho_2) = NULL$ *and* $word(\rho_1) \neq NULL$ *then*

word(ρ_2) = *word*(ρ_1);
if height(ρ_1) = *height*(ρ_2) *then*
add 1 *to height*(ρ_2);
if height$(\rho_2) \leq$ *height*(ρ_1) *then*
define an arc from ρ_2 *to* ρ_1
insert (ρ_1, ρ_2) *into* L
if word$(\rho_1) = NULL$ *and word*$(\rho_2) = NULL$ *then*
word(ρ_1) = *word*$(\alpha)g$;
if word$(\rho_1) = NULL$ *and word*$(\rho_2) \neq NULL$ *then*
word(ρ_1) = *word*(ρ_2);
UNTIL L *is empty.*

Now the tree which contains ω_1 *contains the points of* Δ *as nodes and Schreier − generators contains a set of Schreier generators of* $N_G(\Delta)$. $\bigtriangledown$

The algorithm is easily seen to be correct, since to each pair (α, β) where β is a son of α all generators have been applied such that for each tree the whole image under each generator was considered. The overall time complexity is dominated by the search processes for the roots. By the rule of fusing trees used any tree with k nodes has height at most $\lceil log_2(k) \rceil$. Therefore all searches can be done in $O(nlog(n))$ time. A more advanced technique using path compression could reduce $log(n)$ to a function related to the inverse of Ackermann's function which is at most 3 for all practical values of n.

In constructing an automorphism group G from the bottom, i.e. the identity subgroup, one has to extend the current group G_i by some new generator g of G_{i-1}. Then g fuses some orbits of G_i as we saw before in our example. To find these fusions one can intersect each cycle of g with the orbits of G_i. All orbits with a nontrivial intersection have to be fused.
In the special situation where $\langle g, G_i \rangle$ has the orbits of G_i as blocks these fusions are more easily determined. Then already the image of only one point determines the set of images of all points of that block. This set is again a block and as such it is an orbit of G_i which is easily determined by the labeled branching. To find an orbit of $\langle g \rangle$ with respect to its action on the set of blocks one only has to determine the image blocks in this way for the powers of g.
There is a situation where such blocks occur in a natural way.

4.7 Proposition. *Let* N *be a normal subgroup of a group* G. *If* G *acts on a set* Ω *then the orbits of* N *are blocks of* G.

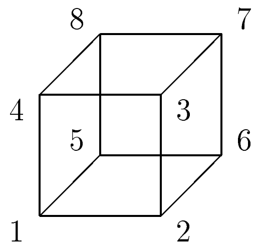
Proof. For $g \in G$ and an orbit $\Delta = \omega^N$ of N also $\Delta^g = \omega^{Ng} = \omega^{gN}$ is an orbit of N. Since orbits are disjoint, they are blocks.

$\square$

By the Sylow theorem each finite group G contains subgroups of the highest prime-power orders p^n dividing $|G|$. Thus we may whish to construct a Sylow-subgroup P of G instead of a full automorphism group G. In this case we can refine any chain of subgroups of P such that

$$P = P_0 > P_1 > \ldots > P_K = \{id\}, \quad P_i \lhd P_{i-1}, |P_{i-1}/P_i| = p$$

for all i.
Then we can use that the orbits of a normal subgroup of a group are blocks of the whole group in each extension step from P_i to P_{i-1}.
For the problem of finding automorphisms the additional restriction that a Sylow-p-subgroup is to be found gives good extra informations. If $P_{i-1} = \langle P_i, g\rangle$ then $g^p \in P_i$ and g has to fuse either p blocks of equal length or leave blocks fixed. Since each P_i is assumed to be uniquely determined by its labeled branching, no $g \notin P_{i-1}$ can fix all orbits of P_i.
We show how this principle works in our example of the automorphism group of the cube. We want to construct a Sylow-2-subgroup P of the full automorphism group G. This time we vary the problem by allowing also reflections as automorphisms.



We find that $(1,2,3)$ is a base of the automorphism group G. Any nontrivial $\tau \in N_P((1,2))$ has to map 3 onto some neighbor of 2 which is not fixed. Then $\tau = (1)(2)(3,6)(4,5)(7)(8)$ is the only choice. The cycles of τ form the orbits of $N_P((1,2))$. In the next step we look for some σ fixing 1 and mapping 2 onto another orbit of length 1 of $N_G((1,2))$. Since 2 also has to be a neighbor of $1^\sigma = 1$ no such σ exists. Thus $N_P((1)) = N_P((1,2))$.
Now we have to move 1 by some ρ onto some other fixed point of τ.
In addition the orbit of 1 under τ has to be of length 2.
The first choice $1^\rho = 2$ is realized by

$$\rho = (1,2)(3,4)(5,6)(7,8).$$

The orbits of $\langle \tau, \rho\rangle$ are

$$\{1,2\}, \{3,4,5,6\}, \{7,8\}.$$

In the next extension step we look for some η mapping $\{1,2\}$ onto the other orbit of length 2, i.e. onto $\{7,8\}$. We choose

$$\eta = (1,7)(2,8)(3,5)(4,6).$$

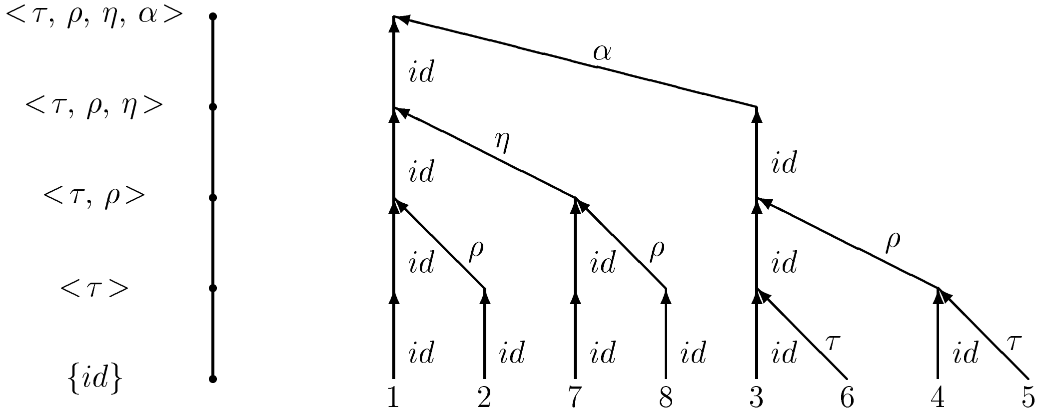
Now we are left with only two orbits of length 4, i.e. $\{1,2,7,8\}$ and $\{3,4,5,6\}$. We find a last extension step with element α fusing these orbits:

$$\alpha = (1,5,8,4)(2,6,7,3).$$

For

$$\begin{array}{rrcl} & P & = & \langle \tau, \rho, \eta, \alpha \rangle \\ > & N_P(\{1,2,7,8\}) & = & \langle \tau, \rho, \eta \rangle \\ > & N_P(\{1,2\}) & = & \langle \tau, \rho \rangle \\ > & N_P((1)) = N_P((1,2)) & = & \langle \tau \rangle \\ > & N_P((1,2,3)) & = & \{id\} \end{array}$$

we obtain the following labeled branching:



It has to be remarked that our definition of a labeled branching differs from that given originally by Jerrum because we note *all* orbits of each group in the chain. Our version is especially useful for the construction of automorphism groups.
In our example it was easy to find automorphisms with some restrictions for the possible image points. Generally, just this is a very difficult problem which we now consider in more details.
Suppose some structure S_1 is described by a set of $k-$tuples $(\omega_1,\ldots,\omega_k) \in \Omega^k$ with entries from an appropriate set Ω. A second structure S_2 may be given by $k-$tuples $(\delta_1,\ldots,\delta_k) \in \Delta^k$ with entries from a set Δ. A bijective mapping $\varphi\colon \Omega \to \Delta$ which transforms the $k-$tuples of S_1 into the $k-$tuples of S_2 is then an isomorphism. If in particular $S_1 = S_2$ then such a φ is an automorphism. In our computation of automorphism groups we did not only look for an arbitrary automorphism. We had additional restrictions like $\omega_1^\varphi = \omega_1, \ldots, \omega_{i-1}^\varphi = \omega_{i-1}$ for some i.

More generally we assume here that on Ω and on Δ we have sequences $(\Omega_1,\Omega_2,\ldots,\Omega_l)$ and $(\Delta_1,\Delta_2,\ldots,\Delta_l)$ of disjoint subsets such that the mapping φ has to map Ω_i onto Δ_i for $i=1,\ldots,l$.
We want to refine these sequences by means of the sets of tuples of the structures such that φ still is compatible with these sequences.

So let S_1 and S_2 consist of t tuples

$$\begin{aligned} S_1 = & \ \{(\omega_{i1}, \ldots, \omega_{ik}) \mid i = 1, \ldots, t \text{ and } \omega_{ij} \in \Omega\}, \\ S_2 = & \ \{(\delta_{i1}, \ldots, \delta_{ik}) \mid i = 1, \ldots, t \text{ and } \delta_{ij} \in \Delta\} \quad \text{ for some } k \in \mathbf{N}. \end{aligned}$$

Then we can classify each S_i as following. Since $\Omega = \cup_{i=1}^{l} \Omega_i$, each entry ω_j in a tuple $(\omega_1, \ldots, \omega_k)$ belongs to some $\Omega_{S(j)}$. Therefore $(\omega_1, \ldots, \omega_k) \in \Omega_{S(1)} \times \Omega_{S(2)} \times \ldots \times \Omega_{S(k)}$ and also $(\omega_1^{\varphi}, \ldots, \omega_k^{\varphi}) \in \Delta_{S(1)} \times \Delta_{S(2)} \times \ldots \times \Delta_{S(k)}$ if φ is an isomorphism mapping Ω_j onto Δ_j for each j. The sequences $(S(1), S(2), \ldots, S(k))$ give thus a classification of the tuples of S_1 and S_2 such that φ is compatible with these classes.
Now we use this information to refine the Ω_i and Δ_i. Suppose the tuples are arranged in classes $C_1, \ldots, C_n$ for S_1 and $D_1, \ldots, D_n$ for S_2 such that φ has to map C_i onto D_i. Then for each $\omega \in \Omega$ we obtain a matrix $A(\omega) = (a_{ij}(\omega))$ such that

$$a_{ij}(\omega) = |\{(\omega_1, \ldots, \omega_k) \in C_i| \quad \omega_j = \omega\}|.$$

Again this matrix of natural numbers is invariant under φ. Therefore it can be used for refining the classes Ω_i, Δ_i respectively.
A simpler version can be used if $S_1 \subseteq \binom{\Omega}{k}, S_2 \subseteq \binom{\Delta}{k}$, i.e. S_1 and S_2 consist of $k-$element subsets of Ω and Δ respectively. If $\Omega = \Omega_1 \cup \Omega_2 \ldots \cup \Omega_l$ then each $T \in S_1$ is contained in some $\binom{\Omega_1}{a_1} \cup \ldots \cup \binom{\Omega_l}{a_l}$, that means T contains exactly a_i elements from Ω_i for $1 \leq i \leq l$. Then $(a_1, \ldots, a_l)$ is invariant under φ and thus can be used to classify S_1 and analogously S_2 into $S_1 = C_1 \cup \ldots \cup C_n,\ S_2 = D_1 \cup \ldots \cup D_n$. For the refinement of the Ω_i we now count for each $\omega \in \Omega$ the subsets in the $i-$th class of S_1 which contain ω. The vector

$$a(\omega) = (a_1(\omega), \ldots, a_n(\omega))$$

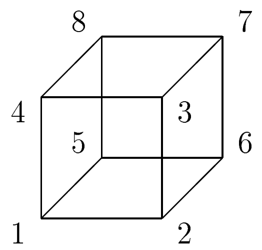
where

$$a_j(\omega) = |\{T \in C_j | \omega \in T\}|$$

can thus be used to distinguish between elements in the same class Ω_i of Ω. This gives a classification which is compatible with φ.
More generally these techniques can be combined in various ways for structures which are defined by several components consisting of sets of tuples or subsets. One can even allow nested structures, but this is more involved.

4.8 Example. In order to describe a cube as an oriented body in a simple manner we note for each vertex together with a neighbor the next neighbor with respect to the orientation. The structure thus consists of 3-tuples (a, b, c) where neighbor c is the next neighbor of neighbor b for vertex a. We want to determine all oriented structure automorphisms α such that $1^{\alpha} = 1$.

Oriented Cube



1 2 5		3 2 4		5 1 6		7 3 8				
1 5 4		3 4 7		5 6 8		7 8 6				
1 4 2		3 7 2		5 8 1		7 6 3				
2 3 6		4 1 8		6 2 7		8 4 5				
2 6 1		4 8 3		6 7 5		8 5 7				
2 1 3		4 3 1		6 5 2		8 7 4				

First we partition $\Omega = \{1, 2, \ldots, 8\}$ into $\Omega_1 = \{1\}, \Omega_2 = \{2, 3, \ldots, 8\}$. Then the set of triples is classified:
$C_1 = \{(1, 2, 5), (1, 5, 4), (1, 4, 2)\}$, $C_2 = \{(2, 1, 3), (4, 1, 8), (5, 1, 6)\}$,
$C_3 = \{(2, 6, 1), (4, 3, 1), (5, 8, 1)\}$, C_4: all other tuples.
¿From this we obtain the following invariant matrices:

$$A(1) = \begin{pmatrix} 300 \\ 030 \\ 003 \\ 000 \end{pmatrix}, \ A(2) = \begin{pmatrix} 011 \\ 100 \\ 100 \\ 122 \end{pmatrix}, \ A(3) = \begin{pmatrix} 000 \\ 001 \\ 010 \\ 322 \end{pmatrix}, \ A(4) = \begin{pmatrix} 011 \\ 100 \\ 100 \\ 122 \end{pmatrix},$$
$$A(5) = \begin{pmatrix} 011 \\ 100 \\ 100 \\ 122 \end{pmatrix}, \ A(6) = \begin{pmatrix} 000 \\ 001 \\ 010 \\ 322 \end{pmatrix}, \ A(7) = \begin{pmatrix} 000 \\ 000 \\ 000 \\ 333 \end{pmatrix}, \ A(8) = \begin{pmatrix} 000 \\ 001 \\ 010 \\ 322 \end{pmatrix}.$$

Now α has to be compatible with the following classes of $\Omega : \Omega_1 = \{1\}, \Omega_2 = \{7\}, \Omega_3 = \{3, 6, 8\}, \Omega_4 = \{2, 4, 5\}$.
An iteration of this process results in no proper refinement of classes. Now we try to put $2^\alpha = 4$. This means that we can apply the refinement process firstly to the sequence $(\{1\}, \{7\}, \{3, 6, 8\}, \{2\}, \{4, 5\})$ and secondly to $(\{1\}, \{7\}, \{3, 6, 8\}, \{4\}, \{2, 5\})$. If α can be extended to an automorphism then the elements from the $i-$th class in the first sequence have to be mapped onto the elements of the $i-$th class in the second sequence. Therefore this also holds true for the refined classes. We carry out the first classification:
$C_1 = \{(1, 2, 5)\}$, $C_2 = \{(1, 4, 2)\}$, $C_3 = \{(1, 5, 4)\}$, $C_4 = \{(7, 3, 8), (7, 6, 3), (7, 8, 6)\}$,
$C_5 = \{(3, 7, 2)\}$, $C_6 = \{(8, 7, 4), (6, 7, 5)\}$, $C_7 = \{(6, 2, 7)\}$, $C_8 = \{(3, 2, 4)\}$,
$C_9 = \{(3, 4, 7), (8, 5, 7)\}$, $C_{10} = \{(6, 5, 2)\}$, $C_{11} = \{(8, 4, 5)\}$, $C_{12} = \{(2, 1, 3)\}$, $C_{13} = \{(2, 6, 1)\}$, $C_{14} = \{(2, 3, 6)\}$, $C_{15} = \{(4, 1, 8), (5, 1, 6)\}$, $C_{16} = \{(4, 3, 1), (5, 8, 1)\}$, $C_{17} = \{(4, 8, 3), (5, 6, 8)\}$.
Now we derive refinements of the classes $\{3, 6, 8\}$ and $\{2, 5\}$: Entry 3 occurs on the first place just once in classes $5, 8, 9$. For entry 6 these classes are $6, 7, 10$ and for entry 8 these classes are $6, 9, 11$. So all three elements fall into different classes. For entry 2 these classes are $12, 13, 14$ while entry 5 occurs on he first place in classes $15, 16, 17$. Thus all resulting classes of Ω consist of just one element and the same holds for the refinement of the classes for the image structure. Therefore if α can be extended to an automorphism then the extension is uniquely determined:

$$\alpha = (1)(2, 4, 5)(3, 8, 6)(7).$$

The interested reader may verify that no reflection is allowed as an automorphism with this presentation of the cube. $\diamond$

An implementation of this method may use lexicographical orderings of the different types of invariants. Then forming the sequences of classes with respect to the ordering of the invariants can be achieved by the bucketsort algorithm of computer science, see p.80 of [54]. If n words of length k with letters from an alphabet of l letters have to be sorted, this algorithm takes $o(k \cdot n + l)$ time. In our case we always have $l \leq k \cdot n$.

4.9 Algorithm (Iterated classification).
Input:

(1) Sequence $(\Omega_1, \ldots, \Omega_m)$ forming a partition of a set Ω.

(2) Set $S = \{t_1, \ldots, t_n\}$ of $k-$tuples with entries from Ω.

Output: Sequence $(\Omega_1, \ldots, \Omega_r)$, refinement of the input partition.

Method: Repeat the following steps, until the number m of classes remains constant.

(1) For each tuple $t \in S$ and for each entry ω_i from t determine Ω_j s.t. $\omega_i \in \Omega_j$. Form the vector of class-indices by replacing ω_i by j.

(2) Sort all vectors of class-indices, mark each tuple by its rank in this ordering of the vectors. Let c be the number of classes.

(3) Initialize matrices $A(\omega)$ of size $k \cdot c$ with entries 0 for each $\omega \in \Omega$.

(4) Run through all tuples $t = (\omega_1, \ldots, \omega_k)$ and through all ω_i of t. If t belongs to class j add 1 to the entry in the $j-$th row and the $i-$th column of $A(\omega_i)$.

(5) Sort all matrices $A(\omega)$, determine the rank of each $A(\omega)$ with respect to this ordering.

(6) Run through each Ω_i and sort all elements ω with respect to the rank of $A(\omega)$. Split each Ω_i according to the different ranks in this ordering.

(7) Determine the new number m of classes in the resulting partition of Ω.

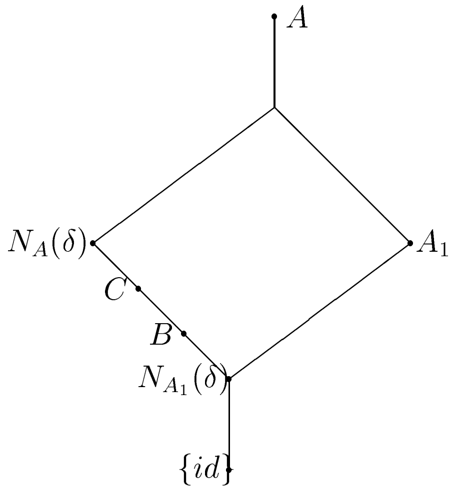
∇

Since we use bucketsort in each sorting routine, we find that each step 1. to 7. can be performed in time $0(k \cdot n)$. The overall complexity is thus $0(k \cdot n \cdot (r - m + 1))$.
The reader may modify this algorithm for the case of $k-$subsets instead of $k-$tuples.
The iterated classification algorithm was developed for graphs by B.D. McKay. His algorithm for computing the automorphism group of a graph seems to be the fastest presently available, and it is widely used.

So far we have discussed the general strategy to build up a labeled branching from the bottom and in each step to find automorphisms α for which some image points are already prescribed or where the set of possible image points is restricted. After any choice of an image point ω^α we can apply the iterated classification algorithm to make use of the implications resulting from this choice and the structure of the given object. The implications then generally will reduce the candidate sets of possible image points under α. Thus, in a backtrack search the set of alternatives will be reduced.
A second source of simplifications is the part of the group of automorphisms that is already known at the actual step.
Let A be a group of automorphism that is already known. Suppose we want to describe all automorphisms β that map a distinguished point ω onto some point δ from a certain candidate set Δ. Then if $\omega^\beta = \delta$ and $\delta^\alpha = \zeta \in \Delta$ for some $\alpha \in A$ the point ζ can be reached from ω by applying $\beta\alpha$. We thus need to test only one point δ from each orbit of A as a candidate for ω^β. Therefore a labeled branching describing each orbit of A can be used in this step.
In backtracking then the proposal $\omega^\beta = \delta$ has to be tested for extendability to an automorphism. Of course we use the iterated classification but we also want to use orbit informations to reduce the candidate set for the next step. If $\gamma \in N_A(\delta)$ then $\omega^{\beta\gamma} = \delta^\gamma = \delta$. Thus if β can be extended to an automorphism also $\beta\gamma$ can and $\beta\gamma$ also maps ω onto δ. We see that the next step requires to know the orbits of $N_A(\delta)$.

Generally we cannot assume that $N_A(\delta)$ occurs in the subgroup chain of our labeled branching for A. Therefore the orbits of $N_A(\delta)$ are not known explicitely. The technique for solving this problem is known as a base-change.

Suppose we have a labeled branching for A with respect to the base $(\omega_1, \ldots, \omega_n)$. We need a labeled branching for $(\delta, \omega_1, \ldots, \omega_n)$, where ω_i is omitted if $\delta = \omega_i$. This can be achieved by computing a labeled branching with respect to this base for $A_i = C_A(\{\omega_1, \ldots, \omega_i\})$ stepping upwards the chain of subgroups, i.e. for $i = n, n-1, \ldots, 0$. Arguing by induction we assume that we have already such a labeled branching for A_1. We also know the orbit of δ under A. Thus, we only need to find representatives for the cosets of $N_{A_1}(\delta)$ in $N_A(\delta)$. We extend $N_{A_1}(\delta)$ gradually to a group B which will become $N_A(\delta)$ eventually. We initialize with $B = N_{A_1}(\delta)$. Since the cosets of A_1 in A are represented in the labeled branching, we can run with some element $a \in A$ through a set of representatives. If $\delta^a = \delta^b$ for some $b \in A_1$ then $c = ab^{-1} \in N_A(\delta)$ can be taken as a representative. Of course c fuses orbits of the already known subgroup B with $N_{A_1}(\delta) \leq B \leq N_A(\delta)$. Thus we can represent $C = \langle B, c\rangle$ by a labeled branching. If $|C| \cdot |\delta^A| = |A|$ then $C = N_A(\delta)$, else we continue with the next choice of a and replace B by C. By the theorem of Schreier



1.3 this procedure stops with the desired labeled branching. For a detailed presentation of this technique and a timing analysis see [10].
We have presented two possible strategies for cutting branches in the backtrack tree corresponding to the attempts of defining image points for an automorphism. Generally it is difficult to find polynomial time bounds for such a backtrack strategy. At least the two algorithms for reducing the set of candidates and thus cutting branches are polynomially bounded. Of course in well behaved cases backtrack may find an automorphism very fast and calling the cutting routines will slow down the algorithm then. It thus depends on the actual application whether it is advisable to implement these procedures.

It should be noted that a further refinement of the strategy is possible again by the homomorphism principle. Depending on the structure of the object one can look for homomorphic images, which preserve automorphisms. Then isomorphism testing or constructing the automorphism group can be done by 1.6 via the homomorphic image. This strategy has been thoroughly carried out in the development of constructing $p-$groups after M.F. Newman and his school, see [57]. In the field of graph isomorphism testing this approach has been proposed by [50] for testing graphs with bounded degree for isomorphism in polynomial time. Generally, many recursive strategies for constructing larger objects from smaller ones can be used for this strategy, for constructions of soluble groups see [42].

5 Orderly Generation

Generally the problem of generating a system of representatives of all isomorphism types of a certain class of objects has been considered by many authors. We mention only a few approaches which are strongly related to our work. Mostly the approaches use some kind of orderly generation of which we will present our subset oriented version below. The basic ideas are from R.C. Read [59] and I. Faradzhev [19, 31], and an important step forward was made by L.A. Goldberg [21], who showed that graphs of a prescribed number of vertices can be generated by adding vertices of maximal degree with a polynomial delay. This was a successful attempt to use structure information in the orderly generation approach. A different strategy had been chosen in the famous **DENDRAL** project which was an early version of a generator of chemical isomers [49], [25]. There not only the number of vertices is prescribed but also the degrees of the vertices are known. This comes from the identification of vertices with atoms of a certain type and therefore of a certain valency. The strategy used in the DENDRAL project can be described by the homomorphism principle. The assumed solutions can be simplified by homomorphisms in several steps and afterwards the inverse direction has to be followed up to construct the solutions. Thus the steps are as follows. In two first steps the vertices of degree 1 and 2 are removed and then cyclic components are formed by removing bridges. A mathematical analysis of the given brutto formula, i.e. the prescribed sequence of degrees of the vertices, gives the possible number of cyclic components, the possible edge degree series for each cyclic component, and the number of interconnections of these components. The isomers are then built up to isomorphism from a catalogue of cyclic structures in a number of steps using some computational group theory, especially double coset computations. Thus, a structural analysis of the problem allowed to break down the problem into smaller pieces which could be handled easier.

The approach presented here for generating graphs with a prescribed degree sequence uses the same basic tools. On one hand the mathematical idea of homomorphism is exploited to use structure information algorithmically. This leads to a recursive construction of graphs from regular graphs, which compared to DENDRAL allows an unbounded number of simplification steps. On the other hand orderly generation is used in the remaining homomorphically irreducible cases. The result is a generator which is extremely fast in many situations but which is slow compared to existing generators as for example B.D. McKay's makeg [53] on the basis of the well known isomorphism testing program **NAUTY** [52] or the present **MOLGEN** generator [26, 72] in smaller cases due to the mathematical overhead. A complexity analysis still is missing and should evolve out of a study of the best recursion strategy within the general framework presented.

We start with some basic principles and then show how they are used in generating graphs up to isomorphism. Generally the problem of generating objects up to isomorphism can be interpreted as the problem of finding orbit representatives from a group action. Since algorithms mostly also need the stabilizers of the chosen representatives, we understand by a solution of the orbit representative problem a determination of a set of representatives together with their stabilizers.

We represent simple graphs as subsets from the set of all 2-element subsets of a vertex set V. Then two such simple graphs are isomorphic iff they lie in the same orbit of S_V. Of course we make algorithmic use of homomorphisms. But in the irreducible cases we need another tool, i.e. orderly generation [59, 19]. We now suppose that a group G acts on a finite set X. We impose on X an ordering $<$ such that also the set 2^X of all subsets of X is lexicographically ordered. This ordering will not be compatible with the action of G, in general. It is therefore quite astonishing that it can be used in solving orbit problems. Each orbit S^G for some $S \in 2^X$ contains a lexicographically minimal element S_0 which we denote as the canonical representative with respect to $<$. In short we say $S \in canon_<(2^X, G)$ iff $S \leq S^G$. Then we have the following fundamental lemma [27],[37].

5.1 Lemma. *If $S \in canon_<(2^X, G)$, $T \subset S$, and $T < S$ then also $T \in canon_<(2^X, G)$.*

Proof: Let $S = T_1 \cup T_2$ and $T_1 < S$ but T_1 not a canonical representative. Then there exists some $g \in G$ such that $T_1^g < T_1$. If $T_1^g = \{x_1, \cdots, x_t\}$ where $x_1 < x_2 < \cdots < x_t$ then for some $i \leq t$ we have $T_1 = \{x_1, \cdots, x_{i-1}, x_i', \cdots, x_t'\}$ and $x_i < x_i'$. Since $S^g = T_1^g \cup T_2^g \supseteq \{x_1, \cdots, x_i\}$, we obtain $S^g < T_1 < S$ contradicting the hypothesis on S.

□

Thus, we only have to enlarge representatives T of smaller cardinality by elements x which are larger than each element in T to obtain candidates for representatives of greater cardinality. This approach can be refined by noticing that there are some further elements y larger than each element in T which can be excluded as x.

5.2 Lemma. *Let $T = \{x_1, \cdots, x_t\}$ be canonical, where $x_1 < x_2 < \cdots < x_t$. Then for $y \in x^{N_G(\{x_1, \cdots, x_i\})}$ for $x_i < x < x_{i+1}$ and $i < t$ the set $T \cup \{y\}$ is not in $canon_<(2^X, G)$. If $i = t$ then if y is not minimal in its orbit under $N_G(T)$ the set $T \cup \{y\}$ is not in $canon_<(2^X, G)$.*

Proof: Let $y = x^g$ for some $g \in N_G(\{x_1, \cdots, x_i\})$ and some x with $x_i < x < x_{i+1}$, $i < t$. Then

$$(\{x_1, \cdots, x_i\} \cup \{x\}) < \{x_1, \cdots, x_i\}^g \cup \{x^g\} \leq T \cup \{y\}$$

such that the subset $\{x_1, \cdots, x_i, y\}$ of T is smaller than T but not canonical. Therefore by lemma 5.1 also $T \cup \{y\}$ is not canonical. The second case is obvious.

□

The candidates obtained after removing the cases of the preceding lemma are often called semicanonical in the special case of graph generation [38, 24, 56].
A test for minimality for each remaining candidate S now has to decide whether there exists some $g \in G$ such that $S^g < S$. The obvious strategy is to run through G with g until either $S^g < S$ or all elements of G have been tested. Of course there must be chosen some ordering in which the elements of G have to be considered. We take a Sims chain with respect to the set X ordered ascendingly as a base $B = (b_1, \cdots, b_n)$. This chain consists of transversals for the left cosets of $G_i = C_G(\{b_1, \cdots, b_i\})$ in $G_{i-1} = C_G(\{b_1, \cdots, b_{i-1}\})$ for $i = n, \cdots, 1$. We order these representatives r by b_i^r. Then we can run through all rG_i in this order of $r's$ and for fixed r in ascending order through G_i for $i = n, \cdots, 1$. There is a case where some elements of G need not be considered in this procedure [24].

5.3 Lemma. *Suppose $S < S^U$ for some subset U of G and $S^g = S$ for some $g \in G$. Then also $S < S^{gU}$, since $S^{gU} = S^U$.*

Thus, for a subgroup U which has already been tested the whole left coset gU can be omitted if $S^g = S$ is detected. Sometimes the elements of X play a different role in a bigger context. Then one has the additional condition that each x^g has to belong to a certain class of elements of X which gives further restrictions for the choice of group elements.

In the preceding Lemma elements stabilizing the subset S were very useful. In general one is interested in all elements stabilizing S. They form the stabilizer subgroup which can be built up during the generation process. Suppose, the stabilizer $N_G(S)$ of each representative S is already known and for each S' a group element g can be found mapping S' onto the representative S from its orbit. Then adding a new element y to S results in a candidate T for a new representative. We have chosen y so that it is minimal in its orbit under $N_G(S)$ and $S < y$. So, we can form $N_G(S) \cap N_G(\{y\})$ and obtain a subgroup of $N_G(T)$. This subgroup is just the stabilizer of y in $N_G(T)$. So we only have to test for the other points x in T whether any $g \in G$ maps $T \setminus \{x\}$ onto S and x onto y. This can be done by our requirements. The elements g found in this way form a set of coset representatives of $N_G(S) \cap N_G(\{y\})$ in $N_G(T)$. This is just the fusion version of the homomorphism principle applied to the present situation.

A simple remark may save a lot of computer time spended in isomorphism testing. Suppose there is an invariant for the objects we want to construct. This is just a function which will take the same values on all elements from the same orbit. Unfortunately, there may exist different orbits on which the invariant takes the same values. Therefore, an invariant is not sufficient to decide whether two objects lie in the same orbit, unless the invariant takes as many different values as there are orbits. An invariant is good, if the number of different values is near to the number of different isomorphism types. Usually, such an invariant can be computed much faster than a check for canonicity. Therefore one can prefer to first compute invariants and compare these before computing canonical forms and comparing those. In the generation problem one may compute for each candidate an invariant, and store the invariants in a hash table. Now, if candidates are generated in increasing order, the first time a **new** invariant is detected, the smallest element of a new orbit must have been found. Thus, this candidate is asserted to be a canonical representative without any test for canonicity.

Often the required solutions have to fulfill some constraints. Then a check if these constraints are fulfilled is usually much faster than a canonicity check and will be done before. One may even hope that after several recursion steps with increasing t only few candidates remain for a canonicity check. The corresponding generation strategy may lead to a larger number of candidates, since in the intermediate steps no restriction to extending canonical representatives only is made. Thus, if a candidate S is not minimal in its orbit then already its predecessor may not have been minimal also. In the light of lemma 2.4 above it is therefore useful to determine the first extension step where this non canonicity could have been detected. Then all further extensions of this candidate must also be rejected. Depending on the selectivity of the additional constraints a delicate balance of steps with constraint checking only and steps with canonicity check combined with tracing back to the earliest detection point is needed for the fastest strategy. This has been followed up by **MOLGEN** [26].

Several different strategies for solving the isomer generation problem have been pursued in the **MOLGEN** project. A first strategy followed the **DENDRAL** strategy [49]. There in a finite number of steps the isomers are constructed out of cyclic graphs. The present version uses orderly generation in filling characteristic classes of rows of the adjacency matrix of the graph representing an isomer [24]. For a future version we have implemented a proposal from [25] as a preliminary step. This version is presently only available for simple graphs [22]. The generation strategy of this version makes a more sophisticated use of homomorphisms and combines them with the orderly generation approach as discussed above, in the irreducible cases. This new strategy is explained in the next section.

6 A graph generator

The generator relies on a strategy of determining first how all graphs with a given degree partition can be built up recursively from regular graphs. The basic result for this approach is the following.

6.1 Theorem. *Let $a = (a_0, a_1, \cdots, a_m)$ be a degree partition of a graph, i. e. there exists a graph G having exactly a_i vertices of degree i for all a_i, and $a_j \neq 0$. If G is any graph with this degree partition then the a_j vertices of degree j span a subgraph T and the remaining vertices span a subgraph H, such that the degree partitions $b = (b_0, \cdots, b_j)$ of T and $c = (c_0, \cdots, c_m)$ of H fulfill the following conditions.*
For each $l \in \{1, \cdots, m\}, l \neq l$ there exists a partition

$$a_l = \sum_{i+k=l} c_{ik}$$

such that for all i

$$c_i = \sum_k c_{ik}.$$

There exists a matrix I with $|H|$ columns and $|T|$ rows such that all entries are 0 or 1 and $\sum_i c_{ik}$ rows of sum k and b_{j-l} columns of sum l.
If on the other hand these conditions are fulfilled for degree partitions a, b, c then for all subgraphs T with degree partition b and H with degree partition c there exists a graph G with degree partition a, having T and H as subgraphs.

There are well known criteria for a degree partition to be the degree partition of a graph [28]. Also, the existence condition for a 0/1-matrix with the required row and column sums can be expressed numerically without any explicit construction by the Gale-Ryser theorem [70, p. 148,149]. Thus, one can decide in advance whether a splitting of a given degree sequence a into two degree sequences of graphs b and c will allow to construct from two corresponding subgraphs T and H a graph G with the required degree sequence a.

It is also clear that the subgraphs T and H in such a case are uniquely determined in any resulting graph G. Also the incidence structure I with a row for each vertex $x \in H$ and a column for each vertex $y \in T$ and noting an egde connecting x to y by the entry 1 in the corresponding place of I is unique. Therefore we have the homomorphism σ mapping G onto (T, H, I). Thus, we may first find all degree sequences b and c and having constructed the corresponding subgraphs find the possible incidence structures I to form the required graphs G.

The strategy obviously reduces the construction problem of simple graphs with prescribed degree sequence to that of regular graphs and the problem of how to paste the subgraphs T and H together. Regular graphs are constructed by an implementation [55] of the method of G. Brinkmann [9]. This is the fastest method known to us presently. The problem of pasting T and H together breaks into two main steps.

Suppose H has c_i vertices of degree i and c_{ik} of them have just k neighbors in T. Then we have to find all partitions of the set of the c_i vertices into these subsets of c_{ik} subsets for all k up to equivalence under the automorphism group of H. This can be done by orderly generation or better by a combination of homomorphism steps and orderly generation. It is important to notice that we will often find only a few different isomorphism types of orbit problems in this step. Moreover, since very often the automorphism group of H will be trivial or act trivially on this set of partitions, the solutions of one case may be implicitely carried over to the isomorphic cases by just noting which bijections must be applied. Also all subgraphs H with the same edge degree sequence and trivial automorphism group can be considered as essential only one case.

Now we know the number of entries 1 in each row and each column of I. We have to find a set of representatives of the different ways to fill this matrix up to the action of the two automorphism groups $AutH$ and $AutT$. We can first partition I into blocks where the corresponding vertices of each row and those of each column are in the same orbit of the automorphism group of T or the stabilizer of the selected partition in the automorphism group of H. Then we have to assign to these blocks a number of 1's that we want to distribute there. We end up with the problem of selecting from the set of places in the block the subsets of those which should get an entry 1. This can be done by orderly generation. By the homomorphism principle only the stabilizer of any such solution has to be considered in its action on the next block to fill. We may even split that block further into the orbits of that stabilizer. Thus, again the acting groups and the blocks become smaller by some factor until no group action appears any more.
It should be clear that a lot of different choices can be made to follow up the general rule of first simplifying by homomorphisms and then using orderly generation in the irreducible cases. Our implementation allows to experiment at certain stages to find a

Table 2: Isomorphism types determined in 10 seconds

vertices	degree partition	number of graphs
20	(0,1,8,7,1,1,0,1,0,0,1)	175729
30	(0,0,4,2,4,0,2,0,10,0,6,0,2)	2900585207520000000
50	(0,2,10,8,11,5,8,1,2,1,0,1,1)	192382967718269922890569744384

good strategy. In the most successful combination strategies we obtain by the implicit handling of isomorphic cases run times of up to 10^{31} graphs per second on a PC, see the small table below. We remark that we used a labeled branching datastructure and a base change algorithm after [10] to deal with the various automorphism groups occuring during the generation process.
As in 6.1 the degree partitions give the numbers of vertices of the degrees $0, 1, \cdots, 12$. Each computation was interrupted after 10.15 seconds and is therefore incomplete. All computations were done on a PC 486DX2 with 8MB of memory.

Compared to generators using only orderly generation or only few reductions by homomorphisms as the present **MOLGEN** system the new approach needs much more time for small cases (up to 20 - 30 vertices). This is due to the overhead caused by determining the different decompositions of the given degree partition. So the methods will have to be chosen depending on the problem size. Still some optimization is needed to make the new generator useful. The most important point seems to be that we need powerful constraints and ways to exploit them as early as possible to reduce the solution space. According to the recursion steps this means to transform selection criteria for the result graphs to criteria applicable to the subgraphs which have to be combined in the recursion step. So one will have to study which properties are hereditary to the regular subgraphs which are the atoms in this approach.

7 Concluding Remarks

This note presents some techniques firstly for determining representatives from group orbits and secondly for finding the groups of which we need the actions. We confined our interest to the homomorphism principle and added some material on orderly generation. Both techniques are considered as general principles which lie beyond many different algorithms. The reader is encouraged to look for solutions of his favorite problem using these principles as a guideline. One might even implement a general toolbox using these techniques without knowing the actual type of objects and data structures of the problem. Thus, an object oriented approach could rely on some methods common to a great variety of object classes and build the generator of all objects up to isomorphism using only these abstract methods.
It should be noted that at least two of the construction strategies here rely on a principle

of determining structures up to isomorphism implicitely. One of these cases is the representation of solutions of the Kramer-Mesner diophantine system by the paths through a graph, [63]. Since the number of paths through a graph tends to grow exponentially with the number of vertices, such an implicit representation reduces the storage requirements logarithmically. The other situation is the graph generation algorithm presented in the last paragraph. There an unrank function can be used to compute a representative of the orbit with a given rank number. Thus, a large part of the isomorphism type listing can be done implicitely doing the computation only on demand. This principle ranges between the mere counting, which may be totally non-constructive as in the Polya-theory, and the construction by explicit listing. It should be remarked that an unrank function is not the tool needed for deciding isomorphism problems. For that purpose, a rank function would be a solution that determines, for each given structure, the rank of its isomorphism class.

References

[1] E. Arnold: Äquivalenzklassen linearer Codes, Zulassungsarbeit Bayreuth 1993.

[2] M.D. Atkinson, R.A. Hassan, M.P. Thorne: Group theory on a micro-computer. Computational Group Theory, M.D. Atkinson(e.d.),

Academic Press, London, 1984, 275-280.

[3] L. Babai, G. Cooperman, L. Finkelstein, A. Seress: Nearly linear time algorithms for permutation groups with a small base. Proc. ISSAC'91(Internat. Symp. on Symbolic and Algebraic Computation), Bonn 1991, 200-209.

[4] L. Babai, E. Luks, A. Seress: Fast management of permutation groups.

proc. 29 IEEE FOCS(1988),272-282.

[5] H. Bender: Entwicklungslinien in der Theorie endlicher Gruppen.

Jber. d. Dt. Math.-Verein. Jubiläumstagung 1990 (1992) Teubner Stuttgart, 77-123.

[6] A. Betten, A. Kerber, A. Kohnert, R. Laue, A. Wassermann: The discovery of simple 7-designs with automorphism group $P\Gamma L(2,32)$. Proc. AAECC-11(11th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes), Paris 1995, LN in Comp. Sc. 948, Springer Berlin (1995), 131-145.

[7] J. Biegholdt: Computerunterstützte Berechnung von Multigraphen mittels Homomorphieprinzip. Diplomarbeit Universität Bayreuth, may 28, 1995.

[8] E. F. Brickell: Solving low density knapsacks. *Advances in Cryptology, Proceedings of Crypto '83*, Plenum Press, New York (1984), 25–37.

[9] G. Brinkmann: Generating cubic graphs faster than isomorphism checking, preprint.

[10] C.A. Brown, C. Finkelstein, P.W. Purdom: A new base change algorithm for permutation groups.
SIAM J. Computing 18 (1989), 1037-1047.

[11] N.G. de Bruijn: Pólya's theory of counting.
Applied Combinatorial Mathematics (Beckenbach, ed.)
Wiley, New York, 1964.

[12] W. Burnside: Theory of groups of finite order.
Dover Publ. , New York, 1955, reprint of 2nd. edition 1911.

[13] G. Butler: Fundamental algorithms for permutation groups.
LN in Comp. Sc. 559, Springer Berlin (1991).

[14] J.J. Cannon: A language for group theory.
Department of Pure Mathematics, University of Sidney,1982.

[15] F. Celler, J. Neubüser, C.R.B. Wright: Some remarks on the computation of complements and normalizers in soluble groups.
Acta Appl. Math. 21(1990),57-76.

[16] F. Celler, A. Niemeyer, W. Nickel, M. Schoenert: Groups, algorithms and programming. Computer Algebra Program System of Lehrst. D für Mathematik, RWTH Aachen(1990).

[17] G. Cooperman, L. Finkelstein: Fast cyclic base change for permutation groups. Proc. ISSAC 92.

[18] M. J. Coster, B. A. LaMacchia, A. M. Odlyzko, C. P. Schnorr: An improved low-density subset sum algorithm. *Proceedings EUROCRYPT '91, Brighton, May 1991* in *Springer Lecture Notes in Computer Science* **547** (1991), 54–67.

[19] I.A. Faradzhev: Generation of nonisomorphic graphs with a given degree sequence (russian). In Algorithmic Studies in Combinatorics, Ed. Nauka, Moscow(1978), 11-19.

[20] M. R. Garey, D. S. Johnson: *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company (1979).

[21] L.A. Goldberg: Efficient algorithms for listing unlabeled graphs. J. Algorithms 13(1992), 128-143.

[22] Th. Grüner: Ein neuer Algorithmus zur rekursiven Erzeugung von Graphen. Diplomarbeit Universität Bayreuth, in preparation.

[23] Th. Grüner, R. Laue, M. Meringer: Algorithms for group actions applied to graph generation. In preparation for Proceedings of DIMACS95.

[24] R. Grund: Konstruktion molekularer Graphen mit gegebenen Hybridisierungen und überlappungsfreien Fragmenten. Bayreuther Math. Schr. 49(1995), 1-113.

[25] R. Grund, A. Kerber, R. Laue: Construction of discrete structures, especially of chemical isomers. to appear in Discrete Applied Mathematics.

[26] R. Grund, A. Kerber, R. Laue: MOLGEN, ein Computeralgebra-System für die Konstruktion molekularer Graphen. Communications in mathematical chemistry(Match)27(1992), 87-131.

[27] R. Hager, A. Kerber, R. Laue, D. Moser, W. Weber: Construction of orbit representatives.

Bayreuther Math. Schr. 35 (1991), 157-169.

[28] S.L. Hakimi: On realizability of a set of integers as degrees of the vertices of a linear graph I. SIAM J. Appl. Math. 10(1962), 496-506.

[29] M. Hall: The theory of groups. Macmillan, New York, 1959.

[30] E. Horowitz, S. Sahni: Data structures in Pascal.

Computer Science Press, Rockville, 1987.

[31] A.V. Ivanov: Constructive enumeration of incidence systems.

Annals of Discrete Mathematics 26 (1985), 227-246.

[32] M. Jerrum: A compact representation for permutation groups.

J. Algorithms 7 (1986), 60-78.

[33] A. Kerber: Algebraic combinatorics via finite group actions.

BI-Wissenschaftsverlag Mannheim, 1991.

[34] A. Korkine, G. Zolotareff, Sur les forms quadratiques. *Math. Ann.* **6** (1873), 366–389.

[35] E.S. Kramer, D.S. Mesner: t-Designs on hypergraphs.

Discrete Math.15(1976),263-296.

Discrete Math. **15** (1976), 263–296.

[36] D. L. Kreher, S. P. Radziszowski: Finding Simple t-Designs by Using Basis Reduction. *Congressus Numerantium***55**(1986), 235–244.

[37] D. L. Kreher, S. P. Radziszowski: Constructing 6-$(14, 7, 4)$ designs. *Contemporary Math.* **111**(1990),137–151.

[38] V. Kvasnicka, J. Pospichal: Canonical indexing and the constructive enumeration of molecular graphs. J. Chem. Computer Science 30(1990), 99-105.

[39] J.H. Kwak, J. Lee: Isomorphism classes of cycle permutation graphs.
Discrete Mathematics 105 (1992), 131-142.

[40] J. C. Lagarias, A. M. Odlyzko: Solving low-density subset sum problems. *J. Assoc. Comp. Mach.* **32** (1985), 229–246.

[41] E. Lang: Datenstrukturen und Algorithmen für Permutationsgruppen.
Diplomarbeit Univ. Bayreuth (1992).

[42] R. Laue: Zur Konstruktion und Klassifikation endlicher auflösbarer Gruppen.
Bayreuther Math. Schr. 9 (1982).

[43] R. Laue: Eine konstruktive Version des Lemmas von Burnside.
Bayreuther Math. Schr. 28 (1989), 111-125.

[44] R.Laue: Construction of combinatorial objects - a tutorial.
Bayreuther Math. Schr. 43 (1993), 53-96.

[45] R. Laue, J. Neubüser, U. Schoenwaelder: Algorithms for finite soluble groups and the SOGOS system. Computational Group Theory, M. D. Atkinson, (e.d.),
Academic Press, London, 1984, 105-135.

[46] R.Laue: Construction of groups and the constructive approach to group actions. Symmetry and Structural Properties of Condensed Matter, T. Lulek, W. Florek, S Walcerz, (e.d.), Zajaczkowo 1994, World Sci. Singapore, 1995, 404-416.

[47] A. K. Lenstra, H. W. Lenstra Jr., L. Lovász: Factoring Polynomials with Rational Coefficients, *Math. Ann.* **261** (1982), 515–534.

[48] J.S. Leon: Computing automorphism groups of combinatorial objects. Computational Group Theory, M.D. Atkinson, (e.d.),
Academic Press, London, 1984, 105-135.

[49] R.K. Lindsay, B.G. Buchanan, E.A. Feigenbaum, J. Lederberg: Applications of artificial intelligence for organic chemistry: The Dendral Project. McGraw-Hill, New York(1980).

[50] E. Luks: Isomorphism of graphs of bounded valence can be tested in polynomial time.

J. Comp. Sys. Sci. 25 (1982), 42-65.

[51] S. Magliveras, D. W. Leavitt: Simple $6-(33,8,36))$ designs from $P\Gamma L_2(32)$. *Computational Group Theory*, M.D. Atkinson ed., Academic Press 1984, 337–352.

[52] B.D. McKay: Computing automorphisms and canonical labelings of graphs. Proc. Internat. Conf. on Combinatorial Theory. LN in Math. 686 Springer, Berlin (1977),223-232.

[53] B.D. McKay: private communication.

[54] K. Mehlhorn: Datenstrukturen und effiziente Algorithmen.

Bd 1, Teubner Stuttgart 1988.

[55] M. Meringer: Erzeugung regulärer Graphen. Diplomarbeit Universität Bayreuth, in preparation.

[56] S.G. Molodtsov: Computer-Aided generation of molecular graphs. Commun. in Math. Chem. (MATCH) 30(1994), 213-224.

[57] M.F. Newman: Determination of groups of prime-power order. Proc. Miniconf. Theory of Groups (Canberra 1975), 17-50.

LN in Math. 573, Springer, Berlin (1977).

[58] G. Pólya: Kombinatorische Anzahlbestimmung für Gruppen, Graphen und chemische Verbindungen.

Acta Math. 68 (1937), 145-254.

[59] R.C. Read: Everyone a winner.

Ann. Discr. Math. 2 (1978), 107-120.

[60] G.-C. Rota, D.A. Smith: Enumeration under group action.

Annali Scuola Normale Superiore-Pica. Classe de Scienze(4)4, (1977), 637-646.

[61] B. Schmalz: Verwendung von Untergruppenleitern zur Bestimmung von Doppelnebenklassen.

Bayreuther Math. Schr. 31 (1990), 109-143.

[62] B. Schmalz: t-Designs zu vorgegebener Automorphismengruppe.

Bayreuther Math. Schr. 41 (1992).

[63] B. Schmalz: The t-Designs with prescribed automorphism group, new simple 6-designs.

J. Combinatorial Designs 1(1993),125-170.

[64] C. P. SCHNORR: A hierachy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science* **53** (1987), 201–224.

[65] C. P. SCHNORR: A More Efficient Algorithm for Lattice Basis Reduction. *J. Algorithms* **9** (1988), 47–62.

[66] C. P. SCHNORR, M. EUCHNER: Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Proceedings of Fundamentals of Computation Theory 91* in *Lecture Notes in Computer Science* **529** (1991), 68–85.

[67] C.C. Sims: Computation with permutation groups. Proc. Second Symp. on Symbolic and Algebraic Manip. (S.R. Petrick, ed.)

New York, 1971.

[68] D. SLEPIAN: Some further theory of group codes. In I. F. Blake: *Algebraic Coding Theory: History and Development (Benchmark papers in electrical engineering and computer science)*, Stroudsburg,Dowden, Hutchinson & Ross Inc. (1973), 118–151.

[69] P.K. Stockmeyer: Enumeration of graphs with prescribed automorphism group. Ph. D. Thesis, Univ of California San Diego, 1973.

[70] J.H. van Lint, R.M. Wilson: A course in combinatorics. Cambridge University Press, 1992.

[71] S. WEINRICH: Konstruktionsalgorithmen für diskrete Strukturen und ihre Implementierung, Diplomarbeit Bayreuth (1993), 274 pp.

[72] T. WIELAND, A. KERBER, R. LAUE: Principles of the generation of constitutional and configurational isomers. *J. Chem. Inf. Comput. Sci.*, **36**, 1996, 431-439.