Algorithms for group actions: homomorphism principle and orderly generation applied to graphs

Thomas Grüner, Reinhard Laue, and Markus Meringer
Universität Bayreuth,
Lehrstuhl II für Mathematik
D-95440 Bayreuth

Abstract

The generation of discrete structures up to isomorphism is interesting as well for theoretical as for practical purposes. Mathematicians want to look at and analyse structures and for example chemical industry uses mathematical generators of isomers for structure elucidation. The example chosen in this paper for explaining general generation methods is a relatively far reaching and fast graph generator which should serve as a basis for the next more powerful version of MOLGEN, our generator of chemical isomers.

1 Introduction

Generally the problem of generating graphs 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. Read [17] and I. Faradzhev [4], and an important step forward was made by L.A. Goldberg [5], where it was shown that by generating graphs by adding vertices of maximal degree a polynomial delay can be achieved. 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 [13], [8]. There for a "planning step" the vertices of degree 1 and 2 are removed and cyclic components are formed by removing bridges. Then an analysis of the given brutto formula 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. 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 makes use of both kinds of ideas. 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

which is slow compared to existing generators as for example NAUTY [14] or the present MOLGEN generator [9] 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.

2 Basic Principles

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. Thus, we have to consider induced group actions. In such situations there is often enough structure to apply algebraic decomposition techniques. The basis of this approach is to make algorithmic use of homomorphisms.

2.1 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 will be used to describe large sets of solutions implicitely in our generator while still allowing an explicit listing on demand.

Suppose an epimorphism $\sigma : (\Omega_1, G_1) \to (\Omega_2, G_2)$ is given. Then we may use σ in two ways. Firstly, if a solution of the orbit problem is known in (Ω_2, G_2) then we only have to look at the preimage sets $\sigma_\Omega^{-1}(\omega_2)$ for representatives ω_2 and let the preimage groups $\sigma_G^{-1}(N_{G_2}(\omega_2))$ act on these sets respectively. We call this usage of σ a **split**.

Secondly, if a solution of the orbit problem is known in (Ω_1, G_1) then we may use the homomorphic images of the representatives and their stabilizers to find a solution in the image domain. There is only one complication that different orbits may be mapped onto the same orbit. So, if the image $\omega_2 = \omega_1^{\sigma_\Omega}$ of one representative ω_1 is used one has to avoid the other image representatives for the elements from $\Delta = \sigma_\Omega^{-1}(\omega_2)$. It should be

noted that those points γ in Δ that are mapped onto ω_1 by some $g(\gamma) \in G_1$ correspond bijectively to the cosets of $N_{G_1}(\omega_1)$ in the full preimage group of $N_{G_2}(\omega_2)$. The elements $g(\gamma)^{-1}$ are just a set of coset representatives. Thus, one can also obtain the stabilizer of ω_2 in this way. We call this usage of σ a **fusion**.

To point out the importance of these two interpretations we shortly mention a typical application in construction problems. Suppose a set of representatives for the isomorphism types of a certain class of graphs is given. Then an extension step adds new subgraphs into each of these graphs. This step can be broken into two parts by the homomorphism principle. In the first part the new subgraph is added as a coloured object in all possible ways up to equivalence under the automorphism group of the old graph. This is the split part of the extension. It is related to the simplification which forgets the new added subgraph. In the next half of the extension step we forget the special colour of the subgraph. This is a simplification σ where we use fusion.

For another simple example using some aspects of homomorphisms consider the generation of multigraphs. A multigraph may be obtained by gradually increasing the highest edge multiplicity by 1. In each step only the automorphism group of the chosen representative needs to be considered with its action on the set of mappings from the set of all edges of maximal degree m to the set $\{m, m+1\}$. This is of course a first application of a split. It is well known that the percentage of simple graphs with nontrivial automorphism group tends to 0 with increasing number of vertices. Thus, in most cases we don't have any group action at all in these steps (a recent implementation [1] demonstrates that already for a small number of vertices the automorphism groups of the multigraphs tend to become trivial very soon). Nevertheless the cases with nontrivial group action have to be considered. So suppose a multigraph G has k egdes of highest multiplicity forming the egde set E_k. Then $AutG$ acts on the set of mappings from E_k to $\{m, m+1\}$. The orbits correspond to the multigraphs with highest multiplicity $m+1$ which can be deduced from G. To solve this orbit problem we may map E_k onto $\{1, \cdots, k\}$, $\{m, m+1\}$ onto $\{0, 1\}$, and $AutG$ onto some corresponding subgroup U of S_k. This is an isomorphism of group actions which maps onto an orbit problem which is independent of k. If we solve these orbit problems as in the case of simple graphs once and for all we only have to note the relation of the solutions there to the desired ones.

One can break up this orbit problem even further by reducing the action of U from $\{0, 1\}^{\{1,\cdots,k\}}$ to $\{0, 1\}^B$ where B is an orbit of U on $\{1, \cdots, k\}$. This may be repeated until the stabilizer of a representative mapping is trivial or the mappings are fully determined.

As usual in algebra, simplifications by homomorphisms stop at some stage because no non trivial homomorphisms exist any more. These situations are called irreducible cases. For such cases we need another tool, i.e. orderly generation[17],[4]. 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[10].

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

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.

2.3 Lemma *Let $T = \{x_1, \cdots, x_t\}$, 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 $canon_<(2^X, G)$.*

The candidates obtained after removing the cases of the preceding lemma are often called semicanonical in the special case of graph generation [12],[7],[16].

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 [7].

2.4 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.

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 [9].

Several different strategies for solving the isomer generation problem have been pursued in the MOLGEN project. A first strategy followed the DENDRAL strategy [13]. 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 [7]. For a future version we have implemented a proposal from [8] as a preliminary step. This version is presently only available for simple graphs[6]. 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.

3 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.

3.1 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 [11]. 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[18, 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[15] of the method of G. Brinkmann [2]. 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 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 [3] to deal with the various automorphism groups occuring

during the generation process.

Isomorphism types determined in 10 seconds

no. of 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)	19238296771826992289056974438 4

As in 3.1 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.

References

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

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

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

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

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

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

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

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

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

[10] R. Hager, A. Kerber, R. Laue, D. Moser, W. Weber: Construction of orbit representatives. Bayreuther Math. Schr. 35(1991), 157-169.

[11] 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.

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

[13] 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).

[14] B.D. McKay: Practical graph isomorphism. Congressus Numerantium 30(1981), 45-87.

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

[16] S.G. Molodtsov: Computer-Aided generation of molecular graphs. preprint.

[17] R.C. Read: Every one a winner. Annals of Discrete Math. 2(1978), 107-120.

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