communications in mathematical
match
and in computer chemistry

no. 41, March 2000

ISSN 0340-6253

MATCDY (41) 137-144 (2000)

Fast Generation of Polycyclic Chains with Arbitrary Ring Sizes

G. Brinkmann[*]
Fakultät für Mathematik
Universität Bielefeld
D 33501 Bielefeld, Germany

A.A. Dobrynin [†]
Sobolev Institute of Mathematics
Siberian Branch of the
Russian Academy of Sciences
Novosibirsk 630090, Russia

A. Krause[‡]
Fakultät für Mathematik
Universität Bielefeld
D 33501 Bielefeld, Germany

Abstract

In this article we give an algorithm for the constructive enumeration of polycyclic chains with arbitrary ring sizes and present sample results and running times of the algorithm.

Introduction

The counting and enumeration of molecular graphs of chemical compounds is a well-established trend in organic and computational chemistry. Isomer enumeration is an important topic in chemical information processing, studies of structure similarity and molecular modelling. A great number of articles are devoted to the analytical or constructive enumeration of isomers of polycyclic conjugated compounds, especially, of isomers of benzenoid hydrocarbons (see [2][16] and references cited therein). The enumeration of isomers

[*]gunnar@mathematik.uni-bielefeld.de
[†]dobr@math.nsc.ru
[‡]akrause@mathematik.uni-bielefeld.de

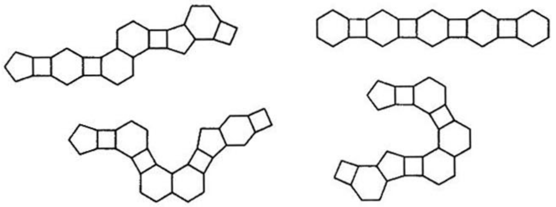
Figure 1: Examples of polycyclic chains with 4,5,6-membered rings.

with other rings than hexagons is also of interest in chemistry. A number of papers deal with the counting of graphs consisting of triangular, tetragonal, pentagonal and q-gonal rings, see e.g. [5][7][12][13][15]. Detailed results on the enumeration of fluoranthenoids, fluorenoids and indacenoids have been reported by Dias and Cyvin *et al.* [8][10][11]. The corresponding molecular graphs contain pentagonal and hexagonal rings. Combinatorial expressions in explicit forms as well as results obtained by computer generation have been presented for the numbers of isomers of other polycyclic systems having rings of two different types (di-4-catafusenes, α-q-catafusenes, mono-q-polyhexes, etc.) [4][6]. The numbers of small polycyclic systems with arbitrary ring sizes have been mainly counted by analytical methods [3][9].

The present paper deals with the constructive enumeration of polycyclic chains with rings of arbitrary sizes. The main goal is to describe the basic ideas of the algorithm and present results of a fast computer program based on this algorithm.

Polycyclic chains

We consider graphs, called polycyclic chains, composed of rings of arbitrary size. Two rings in a polycyclic chain have either one common edge (and are then said to be adjacent) or have no common vertices. No three rings share a common vertex. Each ring is adjacent to two other rings, with the exception of the terminal rings to which a single ring is adjacent. The defined chains include various families of molecular graphs, for instance, unbranched catacondensed benzenoids [2]. Examples of polycyclic chains are presented in Figure 1. It is clear that every polycylic chain is a planar graph and that rings of size 3 can occur only as terminal rings. However, if all rings of a chain are drawn as regular polygons, then this set includes chains which would possess overlapping edges if drawn in the plain.

The generation algorithm

For polycyclic chains the very powerful *homomorphism principle* (see [1][14]) can be applied. To this end we need an underlying coarser structure with the property that isomorphic polycyclic chains have the same underlying coarser structure and that any isomorphism of two polycyclic chains induces an automorphism of the underlying structure. In order to have a strictly coarser partition of the set of polycyclic chains than just isomorphism classes, it is of course necessary that there are also non-isomorphic polycyclic chains with the same underlying coarser structure.

We have chosen the *labelled inner dual* of the polycyclic chains as the coarser structure. The inner dual of a polycyclic chain with n rings is a path of length n. We label each of the vertices of this inner dual with the size of the face that it represents. We call this the *first label.* It is obvious that an isomorphism of two polycyclic chains induces an isomorphism of the corresponding labelled inner duals. Figure 2 shows an example of two non-isomorphic polycyclic chains with the same labelled inner dual.

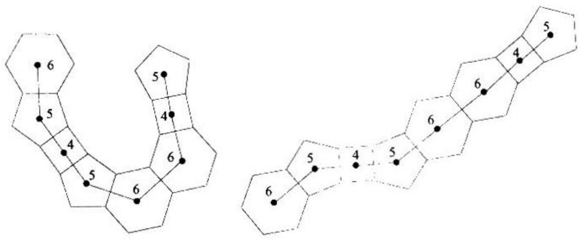

Figure 2: Two non-isomorphic polycyclic chains with the same labelled inner dual.

In order to determine a polycyclic chain uniquely, we have to add some more information. We fix one of the two boundary paths going from one of the ends of the chain to the other and for every face that corresponds to a vertex of degree 2 in the inner dual we add the number of edges of this face contained in the boundary path as a *second label*. An example of such a labelling is given in figure 3.

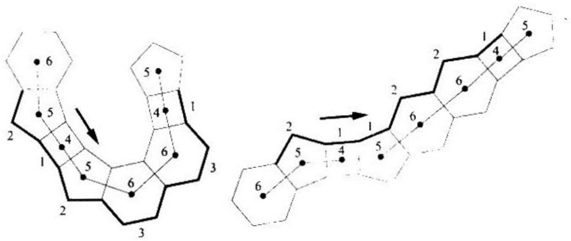

Figure 3: Two different second labellings of the labelled inner dual. Choosing the two boundary paths on the other side, the labellings would be 1,1,1,1,1,1 resp. 1,1,2,2,2,1.

So by fixing a direction in which the chain shall be evaluated and a boundary path, we can encode a polycyclic chain of length n by a sequence of $2n - 2$ integers: First we list the n first labels of the faces in the order given by the direction and then we list the $n-2$ second labels given by the boundary path in the same order. So there are 4 ways to encode a polycyclic chain, corresponding to the different choices for the direction and the boundary path. In order to fix one of them as the *canonical code*, we choose the code that is the lexicographically minimal word among the 4 possibilities. The canonical codes for the structures in figure 2 are $5, 4, 6, 6, 5, 4, 5, 6, 1, 1, 1, 1, 1, 1$, resp. $5, 4, 6, 6, 5, 4, 5, 6, 1, 2, 2, 1, 1, 2$. Obviously there is a unique canonical code for every isomorphism class of polycyclic chains and two different canonical codes correspond to different isomorphism classes. Our aim is to generate the canonical code for every isomorphism class of polycyclic chains.

We do this with a 2-step strategy: First we generate all labelled inner duals and then we assign the second labels to them. We will first describe the basic strategy and then present some optimizations.

Assume that a set of face sizes that may occur in a chain is given. In order to generate all isomorphism classes of labelled inner duals, we assign face sizes that may occur in every possible combination. Having completed the labelling, we check whether reading our code backwards we get a smaller word. If this is the case, we reject our structure, otherwise we accept it. Note that in case the code in reverse order is **strictly** larger than that in the usual order, the direction for the canonical code is fixed. If the reverse code is the same, there is an automorphism of the labelled inner dual and we cannot fix the direction yet.

In the second step we assign second labels to the (valency 2) vertices of the inner dual. If the first label is, say, x, then the second label can be $1 \dots x-3$. Having done so in every possible way, we check whether choosing a different direction or boundary path we get a smaller code. If yes, we reject the chain, otherwise we accept it.

But we can do better:

In cases where the direction is already fixed by the labelled inner dual (which are by far most of the cases), we have to check only the fixed direction of course. For this direction we know that if we would choose the other boundary path, a label y of a face of size x would become $x-2-y$, so the first face in the fixed direction with $y \neq x-2-y$ must fullfill $y < x-2-y$ and it fixes the boundary path for the canonical code, since the other path would give a lexicographically larger word. From then on, we can assign labels in the range $1 \dots x-3$ in every possible way. Since the direction and the boundary path for the smallest code are fixed, we know in advance that every code from then on will be canonical. We do not have to do any checking at the end. Furthermore: If we are not interested in the structures, but just in their number, we can simply multiply up all the numbers of possibilities for the remaining second labels without really assigning the labels.

As soon as the direction and the boundary path are fixed, things become easy and very fast. To this end we can modify our strategy a bit: In the case where the labelled inner dual has not yet fixed the direction, that is where we have a nontrivial automorphism of the labelled inner dual (e.g. in all cases with just one face size allowed), we do not label the faces in the order in which they occur in the path, but we first label face number 2, then number $n-1$, then number 3, number $n-2$, etc. The first case with the second label of face i different from that of face $n+1-i$ fixes the direction again and as soon as the boundary path has also been fixed (by the same arguments as above), we can again go on labelling without checking at the end.

This strategy of non-consecutive labelling can already be applied to force canonicity of the twice labelled inner dual very early. Note that in fact it is again an application of the homomorphism principle with the underlying coarser structures given by the second labelling restricted to segments of equal length on both ends of the inner dual.

Results

The algorithm described in this paper has been implemented as a computer program *chains.c*. It accepts as parameters the sizes of the allowed rings and upper and lower bounds for the length of the chains. It is also possible to fix the number of rings of a certain size that may occur. The program is free for scientific use and the source code of the program can be obtained from any of the authors. Though it is possible to write the structures to a file, it is faster to generate them from the spot whenever they are needed for tests. Sample numbers of polycyclic chains for several parameters and computation times are presented in Table 1. The times without brackets are those needed to determine the structures without really forming, coding and writing them. The times in brackets are the times needed when the structures are also formed, coded in *planarcode* format (a code for planar graphs described in the source code) and written to *stdout*.

References

[1] G. Brinkmann. Isomorphism rejection in structure generation programs. to appear in the Proceedings of the DIMACS Workshop on Discrete Mathematical Chemistry 1998.

[2] J. Brunvoll, B.N. Cyvin, and S.J. Cyvin. Enumeration of benzenoid systems and other polyhexes. *Top. in Curr Chem.*, 162:65–180, 1992.

[3] J. Brunvoll, B.N. Cyvin, and S.J. Cyvin. Isomers of polycyclic conjugated hydrocarbons with arbitrary ring sizes: generation and enumeration. *Comput. Chem.*, 17:291–296, 1993.

[4] J. Brunvoll, B.N. Cyvin, and S.J. Cyvin. Enumeration of poly-5-catafusenes representing a class of catacondensed polycyclic conjugated hydrocarbons. *J. Chem. Inf. Comput. Sci.*, 36:91–99, 1996.

[5] B.N. Cyvin, S.J. Cyvin, J. Brunvoll, and A.A. Dobrynin. Enumeration of unbranched catacondensed systems of congruent polygons. *Vychislitel'nye Sistemy*, 155:3–14, 1996.

[6] S.J. Cyvin, B.N. Cyvin, and J. Brunvoll. Unbranched catacondensed polygonal systems containing hexagons and tetragons. *Croat.Chem.Acta.*, 69:757–774, 1996.

[7] S.J Cyvin, B.N. Cyvin, J. Brunvoll, E. Brendsdal, F. Zhang, X. Guo, and R. Tošić. Theory of polypentagons. *J. Chem. Inf. Comput. Sci.*, 33:466–474, 1993.

[8] S.J. Cyvin, B.N. Cyvin, J. Brunvoll. F. Zhang, X. Guo, and R. Tošić. Graph-theoretical studies on fluoranthenoids and fluorenoids: enumeration of some cata-condensed systems. *J. Mol. Struct. (THEOCHEM)*, 285:179–185, 1993.

[9] J. R. Dias. A periodic table for polycyclic aromatic hydrocarbons. 2. polycyclic aromatic hydrocarbons containing tetragonal, pentagonal, heptagonal, and octagonal rings. *J. Chem. Inf. Comput. Sci.*, 22:139–152, 1982.

[10] J.R. Dias. Deciphering the information content of chemical formulas: chemical and structural characteristics and enumeration of indacenes. *J. Chem. Inf. Comput. Sci.*, 32:203–209, 1992.

[11] J.R. Dias. Studies in deciphering the information content of chemical formulas: a comprehensive study of fluorenes and fluoranthenes. *J. Chem. Inf. Comput. Sci.*, 32:2–11, 1992.

[12] R. Doroslovački, I. Stojmenović, and R. Tošić. Generating and counting triangular systems. *BIT*, 27:18–24, 1987.

[13] S.B. Elk. An algorithm to identify and count coplanar isomeric molecules formed by the linear fusion of cyclopentane modules. *J. Chem. Inf. Comput. Sci.*, 27:67–69, 1987.

[14] R. Grund, A. Kerber, and R. Laue. MOLGEN – ein Computeralgebrasystem für die Konstruktion molekularer Graphen. *match 27*, pages 87–131, 1992.

[15] R. Tošić, R. Doroslovački, and I. Stojmenović. Generating and counting square systems. *Proc. VIII Yugoslav Seminar on Graph Theory, Novi Sad, April 17–18, 1987*, pages 127–136, 1989.

[16] N. Trinajstić, S. Nikolić, J.V. Knop, W.R. Müller, and K. Szymanski. *Computational chemical graph theory: characterization, enumeration and generation of chemical structures by computer methods.* Simon & Schuster: Horwood, 1991.

Ring sizes (number)	Total number of rings	Number of graphs	time (sec)
6	5	10	0
6	10	1681	0 (0.02)
6	15	399310	0 (2.37)
6	20	96864964	0.03 (758.3)
6	25	23535971854	0.34
6	30	5719200505225	6.2
6	35	1389765184685602	82.8
6	40	337712929999378756	1514.5
10	12	70627216	0.04
15	10	107505792	0.05
5 6	10	391251	0.01 (1.58)
5 6	15	1220750001	0.24
5 6	20	3814699218751	11.25
5 6	25	11920929101562501	615.4
5(10) 6(10)	20	483853268016	2.4
5(15) 6(15)	30	3161168846625669120	1491.9
4 6	16	268468224	0.69
4(8) 6(8)	16	9230967	0.15
4 5 6	10	3781656	0.14
4 5 6	12	136062864	1.6
4(4) 5(4) 6(4)	12	4111740	0.2
8 9 10	9	1377582840	0.3
8 9 10	10	24795069336	1.0
8(3) 9(3) 10(3)	9	111933150	0.02
4 5 6 7 8	10	16018233975	34.2
4(2) 5(2) 6(2) 7(2) 8(2)	10	81321168	2.0

Table 1: The numbers of polycyclic chains and the time to determine them on a processor Pentium II 350 Mhz.