An approach to exhaustive generation of objects without testing on isomorphisms. Application of the method to the cell growth problem

Lyuba Alboul
Sheffield Hallam University, UK
e–mail: L.Alboul@shu.ac.uk

Alexandre Netchaev*
University of Twente, the Netherlands
e–mail: netchaev@math.utwente.nl

1 Introduction

Generating and enumerating a specific class of objects are fundamental problems in discrete mathematics and related fields. In this communication we discuss a new approach to the problem of generation that allows to avoid testing on isomorphisms without producing duplicates. We illustrate our approach by presenting several algorithms to generate exhaustively some classes of non–isomorphic combinatorial objects, such as dissectible polyhedra and polygons, and triangular animals.

The usual approach to generate a class of objects is *incremental*: one starts from an initial object and adds one generating block (unit) at a time. Depending on the objects to be generated, the generating unit might be a vertex, an edge, a triangle, and so on. In the generating process one encounters, in general, two types of procedures. One procedure aims at obtaining objects with $m+1$ generating blocks from an object with m generating blocks, and the other – at obtaining from an object with m generating blocks another object with the same number of generating blocks by exchanging two generating blocks at a time, or by replacing a fixed group of generating blocks by another fixed group. The first procedure occurs in various combinatorial problems, when all objects of some class must be enumerated up to a given number of generating blocks. The second procedure might be a sub–procedure of the first procedure. One encounters the second procedure in its 'pure' form, for example, in the triangulation problem, when one needs to transform a triangulation of n points into another triangulation. In a plane such a transformation often is *flipping an edge*, and in a space - *replacing two adjacent tetrahedra by the other three*, and vice versa. In both procedures only one operation is performed at each step: adding a generating block, or exchanging generating blocks.

One of the most difficult tasks, due to its high computational expenses, in the process of enumerating/generating combinatorial objects is to avoid duplicates, especially if one deals with the problem of generating unlabelled objects. In this case the problem of eliminating isomorphic objects, or controlling isomorphisms, becomes crucial. For this reason many algorithms are restricted to generation of so-called rooted objects, when one 'detail' of the object is fixed (for example, a 'node'/vertex, or a 'side'/edge) [2]. Choosing a root destroys most of the symmetries, which makes enumeration/generation easier. However, the number of rooted objects is larger then that of non-rooted ones, and some non-isomorphic rooted objects are still isomorphic in the usual setting. Therefore, if we want to derive from 'rooted' objects non–isomorphic non–rooted ones, an additional check on isomorphisms is needed. Indeed, if an object has no symmetries, we can

*The research of the author is partially supported by the NWO (STW) (Dutch Organisation for Scientific Research), project No. TWI4816

presume that it does not matter what edge or vertex we take as a 'root', however if an object has symmetries, then some roots will be equivalent and therefore isomorphic non–rooted objects will be produced in the process of generating. On the other hand, we cannot completely rule out the possibility of generating isomorphic objects at some step of the generating process even if the initial object has no symmetries.

In our method we use a heuristic idea to fix several 'roots' at each step of the generating process in order to avoid repetitions of the objects and the consequent testing on isomorphisms. This idea leads to the logical conclusion: at each step of generation to add not only one generating block at time, but, depending on the situation, a collection of these generating blocks.

We apply our method to generate explicitly and exhaustively several types of objects: non–isomorphic simplicial *dissectible* polyhedra, their two-dimensional counterparts, and so–called *triangular animals.* The generating block is a vertex (together with a corresponding star). Therefore, in the case of dissectible polyhedra we use tetrahedra as generating blocks, and in the case of 'dissectible' polygons - triangles. We refer to generating blocks in both cases as *generating vertices.* In our approach at each step of the generating process only a fixed number of generating vertices are simultaneously added with the advantage that no testing on isomorphisms is required. Our method is not incremental, but it allows for parallelisation. In such a setting the problem is similar to the problem of simultaneously edge flipping in triangulations, recently arisen in computational geometry [6]. In [1] our first algorithm was introduced. In this communication we shortly describe a theoretical background of constructive enumeration of dissectible polyhedra without testing on isomorphisms, and then present several new algorithms to generate 'dissectible' polygons and triangular animals.

2 Main definitions and concepts

A dissectible polyhedron is inductively defined as follows [3]:

Definition 1

1. *A triangle and a tetrahedron are both dissectible polyhedra.*
2. *A dissectible polyhedron with $(n+1)$ tetrahedra is obtainable from a dissectible polyhedron P with n tetrahedra by adding a new tetrahedron having precisely an exterior triangle in common with P.*

The concept of dissectible polyhedron is a natural generalisation of the concept of dissections of a polygon which dates back to Euler. Euler formulated this problem as enumeration of the ways of triangulating a convex polygon by means of nonintersecting diagonals [5]. This problem is equivalent to the problem of dissecting a disk into triangular regions, since the boundary of a polygon is homeomorphic to the boundary of a disc. There are many works dedicated to the problem of dissecting a polygon or a disc. In the later case not only triangulated regions are considered [4]. The papers are mostly dedicated to the problem of enumerating dissections combinatorially. The interested reader is referred to the bibliography in [3], where also a method for combinatorial enumeration of dissectible polyhedra is given.

Our method deals with constructive enumeration of dissectible polyhedra, or, in other words, with their explicit generation. By means of the method we generate exhaustively dissectible polyhedra and polygons without duplications and testing on isomorphism. Both objects, a dissectible polyhedron and a polygon, represent triangulations. For simplicity, we refer to them as *dissectible triangulations.* We generate all possible groups of automorphisms for a given dissectible triangulation (called a *preceding triangulation*) and based on this generation we reconstruct all non-isomorphic triangulations that are 'derived' from the given one. We call those triangulations *successive triangulations.* The processes of generating groups of automorphisms and reconstructing triangulations are superposed, so the number of operations in the algorithm is reduced. In order to avoid producing duplicates we add simultaneously a number of new generating vertices. The triangulations that are generated from different preceding ones, are then non–isomorphic by construction.

Essentially, our procedure consists of the following steps:

1. Take an initial (preceding) triangulation; define its groups of automorphisms.
2. Insert new vertices in such a way, that each group of automorphisms of the initial triangulation yields only one new (successive) triangulation (a 'derived' representative of the given automorphism group).
3. Repeat the above two steps for each new obtained triangulation.

Note. Only at the initial step a full group of automorphisms is generated. At each following step a restricted check on automorphisms suffices.

3 Results

On the base of our method several algorithms have been developed, each with a different number of generating vertices. In each subsequent algorithm we were able to reduce considerably this number. We denote the number of generating vertices by GV_i, where i corresponds to the number of the related algorithm.

In the first algorithm, developed for dissectible polyhedra, GV_1 is not less than k; k is the number of vertices of degree three in a triangulation with n vertices. Suppose, that GV_1 is equal to l, then the number l satisfies the following condition:

$$k \leqslant l \leqslant 2(n-2), \tag{1}$$

where k is a number of vertices of degree 3 of the initial/preceding triangulation, and $2(n-2)$ is the number of all faces (triangles) of a triangulation with n vertices. Indeed, we cannot add more than $2(n-2)$ vertices to a triangulation with n vertices. We add new vertices in a special way: to each star of a vertex of degree 3 one new vertex is always added. We define the vertices of degree three to which stars we add a new vertex as *active* vertices, and corresponding stars - as *active* stars. A new vertex is always added to each active star (to one of three possible faces) and the remaining $l-k$ points are added to some other faces of the same triangulation with n vertices. The latter faces do not need to belong to the stars of vertices of degree three. The remaining $l-k$ vertices are added not simultaneously, but one by one. We call our method *k-incremental*, since at the first step we always add k vertices simultaneously, but the further steps are incremental, because at each next step we add only one vertex from the remaining $l-k$ vertices that may be added. Therefore GV can be presented as $GVS + GVR$, where GVS is fixed and determined by the number of active vertices, and GVR varies. This algorithm allows a direct analogue for dissectible polygons. In this case k is the number of vertices of degree 2, GV_2 does not exceed n, where n is the number of boundary edges (equal to the number of vertices in the preceding triangulation). The following theorem is proved:

Theorem 2 *All generated dissectible triangulations are non-isomorphic and their generation is exhaustive.*

We have developed two more algorithms for dissectible polygons. The GVS has been significantly reduced. In the best algorithm is equal to 2 or 3, depending on the situation. We modify then this algorithm to generate *triangular animals*. This problem belongs to so-called *cell-growth problem* [9]. An animal is constructed from a collection of equivalent cells. A cell can be an equilateral triangle, a square or a hexahedron. The cells must be non–overlapping. Recently, the cell-growth problem has attracted attention of specialists in various fields (see, for example, [7]). Literature dedicated to generation of animals is scarce. One of the recent works is [8]. In this work a constructive enumeration of triangular animals up to 13 cells (triangles) is given, however, the algorithm requires testing on isomorphisms . We have developed algorithms to generate simply-connected and multiply-connected triangular animals without internal vertices. Simply–connected triangular animals have been easily generated up to 18 cells (20 vertices). The results of generation are given in Tab. 1.

Number of cells	13	14	15	16	17	18
Number of animals	7541	20525	55633	152181	416188	1143526

Table 1: Counts of simply–connected triangular animals.

4 Conclusion

We presented a new approach to the problem of constructive enumeration of some classes of the objects, that excludes testing on isomorphisms. By simultaneously adding several generating blocks and by treating this operation as a primitive operation, one can expect a considerable reduction of computational cost. Another advantage is that our approach divides the computation into mutually disjoint sub-computations. In another words, our computation process is a forest structure. Sub-computation processes (trees) are completely independent, and therefore can be generated on separate processors. The open problem is to generalise the method to more complex objects. The method might also be useful in simulation of growth of other structures, such as molecular structures, corals, fractals, where a similar growth pattern is repeated at each subsequent growth step.

References

[1] Alboul, L., Netchaev, A.: Isomorphic–free generation of some classes triangulations without repetitions. In Proc. of EWCG 2002 (European Workshop in Computational Geometry), April 10-12, 2002, Warsaw, pp. 116-117

[2] Avis, D.: Generating rooted triangulations without repetitions. Algorithmica, 16 (1996), 618–632.

[3] Beineke, L.W., Pippert, R.E.: Enumerating dissectible polyhedra by their automorphism groups. Can. J. Math., 26(1):50–67, 1974.

[4] Brown, W.G.: Enumeration of quadrilangular dissections of the disc. Can. J. Math., 17 (1965), 302–317.

[5] Euler, L.: Novi commentarii academiae scientiarium imperialis petropolitanae 7 (1758-1759), 13-14.

[6] Galtier, J., Hurtado, F., Noy, M., Perennes, S., Urrutia, J.: Parallel edge flipping. See: http://www-ma2.upc.es/ hurtado/flipcorner.html

[7] Ivanov, A.O, Tuzhilin, A.A.: Branched geodesics. Geometrical theory of local minimal networks. (in Russian) Russian Research in Mathematics and Science, Vol. 5. The Elwin Mellen Press 1999. ISBN 0-7734-3178-0.

[8] Konstantinova, E.: Constructive enumeration of triangular of triangular animals. See: http://com2mac.postech.ac.kr/papers/2000/00-22.ps

[9] Palmer, E.M.: Variations of the cell-growth problem. In: Graph theory and applications. Proc. Conf. western Michigan University, May 10-13 (1972), pp. 215-224, Berlin 1972.