Computing Canonical Polygonal Schemata with Generalized Maps

Guillaume Damiand, Sylvie Alayrangues

▶ **To cite this version:**

Guillaume Damiand, Sylvie Alayrangues. Computing Canonical Polygonal Schemata with Generalized Maps. Electronic Notes in Discrete Mathematics, 2008, 31, pp.287-292. 10.1016/j.endm.2008.06.058 . hal-00315869

HAL Id: hal-00315869

https://hal.science/hal-00315869v1

Submitted on 1 Sep 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Computing Canonical Polygonal Schemata with Generalized Maps

Guillaume Damiand

LaBRI, Université Bordeaux 1, UMR CNRS 5800, 33405 Talence cedex, France
damiand@labri.fr

Sylvie Alayrangues

SIC-XLIM, Université de Poitiers, UMR CNRS 6172, 86962 Chasseneuil, France
alayrangues@sic.univ-poitiers.fr

Abstract

This paper shows that a well-known algorithm proposed to compute the canonical polygonal schema of a surface can be transferred onto a 2-dimensional generalized map. We show that transformation rules on polygonal schemata can be achieved in O(1) with generalized maps, which can help optimizing existing algorithms.

Keywords: canonical polygonal schema, generalized map, topological invariant.

1 Introduction

The comparison of two topological surfaces [1] is a frequent question that may be solved by computing the canonical polygonal schema associated to each surface. In [3], authors propose an algorithm which computes this topological invariant by using particular transformation rules.

This paper shows that these transformations can be defined on a 2D generalized map encoding a surface. Firstly, we exhibit the links between polygonal

⋆ Paper published in Proceedings of International Conference on Topological & Geometric Graph Theory, Electronic Notes in Discrete Mathematics vol. 31, pp. 287-292, 2008. Thanks to Elsevier. The original publication is available at http://dx.doi.org/10.1016/j.endm.2008.06.058

[1] By surface, me mean compact connected 2-manifold orientable or not.

schemata and generalized maps. Then we show how the elementary *cut-and-paste* operations are related to edge shifting operations. We finally propose $O(1)$ algorithms to perform the high-level transforms proposed in [3], which proves that the algorithm of [3] can be transferred onto generalized maps.

2 Links Between Polygonal Schemata and Generalized Maps

A connected surface S can be represented by a polygon P, called *polygonal schema* [1], containing an even number of oriented edges. These edges are paired up, such that the topological space obtained after identification of paired edges is homeomorphic to S. Such a polygon can be conveniently represented by a word W, which is constructed by going along the boundary of the polygon, and adding a letter per edge. Two edges have the same letter if they have to be identified. This letter is written under an horizontal bar if the orientation of the edge is opposite to the direction of motion along the boundary (e.g. a and $\overline{a}$). Several polygons equivalently represent a same surface, but all can be simplified into the *canonical polygonal schema* form that is unique and fully characterizes the topology of the surface.

A generalized map [2] is a combinatorial structure used to represent space subdivisions of any dimension. We focus here on closed connected 2D generalized maps (or 2G-maps). Intuitively such a 2G-map is constructed by decompositions and splits of a surface into faces, edges and vertices. The basic element of a 2G-map is called a *dart*. Each dart is incident to a vertex, an edge and a face. Darts are glued together with three involutions $\alpha_i, i \in \{0, 1, 2\}$, to keep the structure of the subdivision: α_0 (resp. α_1, α_2) connects two vertices (resp. edges, faces) incident to a same edge and face (resp. vertex and face, vertex and edge) (Fig. 1A). We denote $\alpha_{ij}(d)$ for $\alpha_j(\alpha_i(d))$, and call

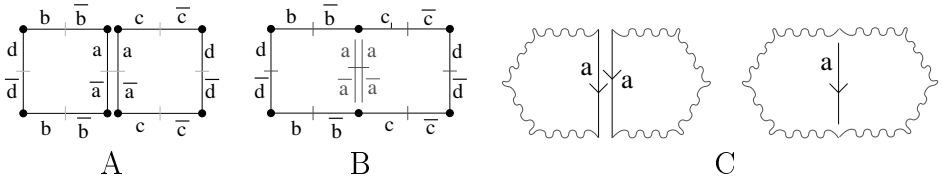

Fig. 1. (A) A 2G-map with labeled darts: α_0 is represented by small grey segments, α_1 by black dots and α_2 connects two darts with the same label. (B) Equivalent 2G-map where edge a has been removed. One associated word is $W = dbc\overline{d}\overline{c}\overline{b}$. (C) The corresponding operation to construct the polygonal schema is simply the glue of the 2 edges labeled a.

1-sew(d_1, d_2) the operation that links darts d_1 and d_2 by α_1.

Darts of a closed connected $2G$-map are labeled in the following way. Letter a is given to an arbitrary dart d, and is also associated to $d\alpha_2$. Letter $\overline{a}$ is associated to $d\alpha_0$ and to $d\alpha_2\alpha_0$. The edge associated to d is hence represented by the letter a (indeed, each edge of the $2G$-map is made of 4 darts, $\{d, d\alpha_0, d\alpha_2, d\alpha_2\alpha_0\}$ where d is one dart incident to this edge). Each dart is processed in the same way until all edges are labeled with different letters (Fig. 1A).

A polygonal schema can be constructed from this labeled $2G$-map by simplifying it into a $2G$-map containing only one polygonal face. This is simply achieved by removing each degree two edge (Fig. 1B). This operation is similar to the glue of two pieces of a polygonal schema (Fig. 1C).

The word representing the surface is computed as follows. Arbitrarily take a dart d and begin the word with its associated letter, then add the letter of the dart $d\alpha_0\alpha_1$ (which is the first dart encountered on the next edge). Go on adding letters in the same way until the first dart is found again. Note that the orientation of the edges of the polygon is straightforwardly obtained.

3 Basic Transformations

There are two "basic" transformations, dangling edge removal and edge shifting, which correspond to transforms A and B of [3].

Transform A. If the word is like $a\overline{a}X$, then edge a is a dangling edge which can be removed to obtain word X (see Fig. 2).

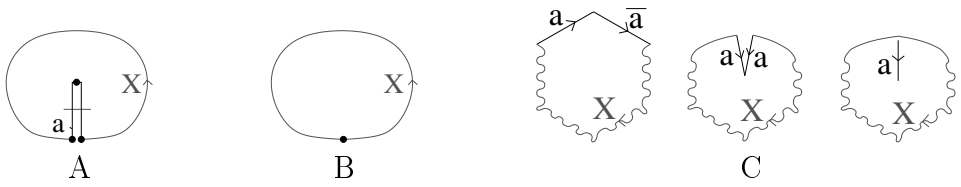

Fig. 2. Transform A. (A) A $2G$-map with a dangling edge a (B) $2G$-map after the removal of a. (C) Corresponding operation on the polygonal schema.

The edge shifting operation, $S_b(a)$, pushes an edge of a $2G$-map, a, along the direction of one of its neighbor edge, b. It does not depend on the orientability of edge a. However, both cases involve different modifications on the associated polygonal schema.

Transform B_1. If edge a is orientable (see Fig. 3), the corresponding word is $aX\overline{a}bY$ (with eventually X or Y empty). After $S_b(a)$, this word becomes

$baX\overline{a}Y$: the letter after $\overline{a}$ is put before a. The edge can also be shifted along the second direction, and roles of a, $\overline{a}$, b and $\overline{b}$ can be exchanged. This gives 8 possible transformations for the orientable case.

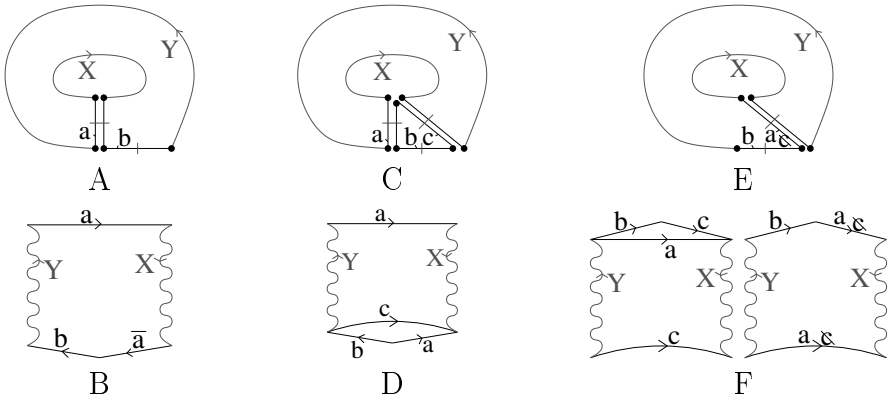

Fig. 3. Transform B_1. (A) A $2G$-map where edge a (which is orientable) will be shifted along edge b. (B) Corresponding polygonal schema. (C) and (D) first step: a new edge is added, c, which cuts the face. (E) and (F) second step: edge a is removed, in two steps in the polygonal schema, and edge c is re-labeled in a.

Transform B_2. If edge a is non-orientable (see Fig. 4), the corresponding word is $aXabY$ (with eventually X or Y empty). After $S_b(a)$, this word becomes $a\overline{b}XaY$: the letter after the second a is put after the first a and negated. The edge can also be shifted along the second direction, and we can exchange the role of both a. This gives 4 possible transformations for the non-orientable case.

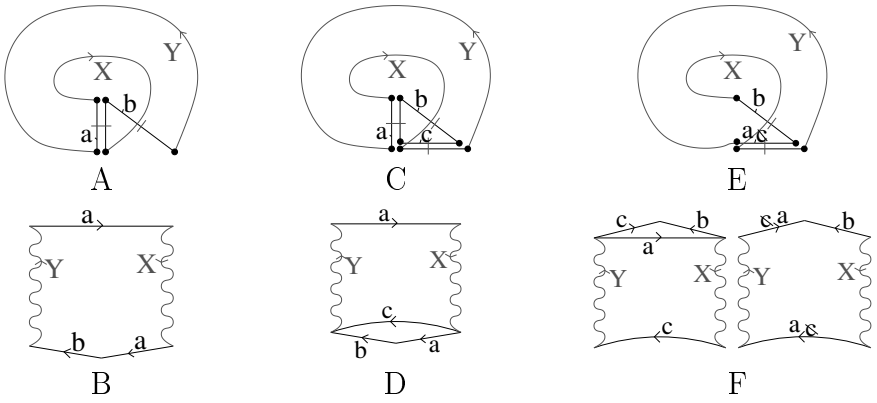

Fig. 4. Transform B_2. (A) A $2G$-map where edge a (which is non-orientable) will be shifted along edge b. (B),(C), (D), (E) and (F) have the same caption as in Fig. 3.

4 High-level Transformations

Several edge shifting operations can be combined in order to obtain the three other transformation rules given in [3]. However, by using such a technique, the complexity of each transformation C and D is in $O(n)$, with n the number of edges of the polygon. For each transformation, we design an optimized operation that modifies directly the given $2G$-map into its final form in $O(1)$.

Transform C. Starting from a word $aXaY = aXay_1y_2 \dots y_k$, where a is a non-orientable edge, we shift edge a several times to obtain word $a\overline{y_k} \dots \overline{y_2}\ \overline{y_1}Xa$ also denoted by $a\overline{Y^{-1}}Xa$.

Optimized Transform C is given in Algo. 1. The transformation of Y into $\overline{Y^{-1}}$ is achieved here in $O(1)$.

Algo 1: Transform C(a)

1-sew($\alpha_{01}(d_a)$,$\alpha_{201}(d_a)$);
1-sew($\alpha_1(d_a)$,$\alpha_0(d_a)$);
1-sew(d_a,$\alpha_{20}(d_a)$);

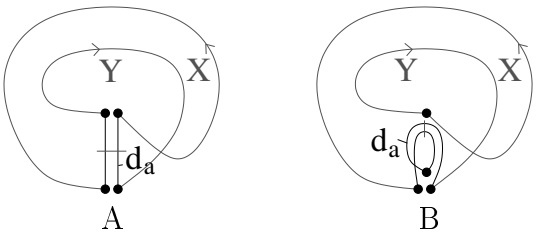

Fig. 5. Optimized Transform C. (A) $2G$-map corresponding to $aXaY$. (B) $2G$-map obtained after applying Algo. 1 which corresponds to $aa\overline{Y^{-1}}X$.

Transform D. Starting from a word $aXbY\overline{a}U\overline{b}V$, where a and b are orientable edges, we shift edge b to obtain word $abY\overline{a}U\overline{b}XV$. Then we shift edge b obtain word $ab\overline{a}UY\overline{b}XV$. Finally, we shift edge a to obtain word $UYab\overline{a}\overline{b}XV$ which is equal to $ab\overline{a}\overline{b}XVUY$.

Algo 2: Transform D(a,b)

1-sew($\alpha_1(d_a)$,$\alpha_{21}(d_a)$);
1-sew($\alpha_1(d_b)$,$\alpha_{21}(d_b)$);
1-sew(d_a,$\alpha_{201}(d_a)$);
1-sew($\alpha_{01}(d_b)$,$\alpha_{201}(d_b)$);
1-sew($\alpha_2(d_b)$,$\alpha_{01}(d_a)$);
1-sew($\alpha_2(d_a)$,$\alpha_{02}(d_b)$);
1-sew(d_b,$\alpha_0(d_a)$);
1-sew($\alpha_{20}(d_a)$,$\alpha_0(d_b)$);

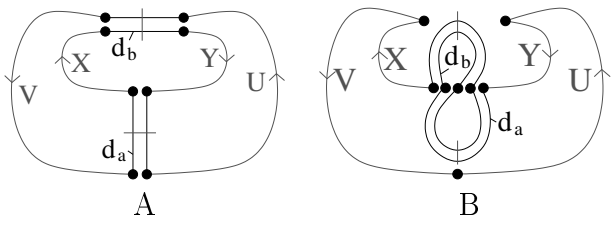

Fig. 6. Optimized Transform D. (A) $2G$-map corresponding to $aXbY\overline{a}U\overline{b}V$. (B) $2G$-map obtained after applying Algo. 2 which corresponds to $ab\overline{a}\overline{b}XVUY$.

Optimized Transform D is given in Algo. 2.

Transform E. Starting from a word $ab\bar{a}\bar{b}ccX$, we shift edge c to obtain word $abcbacX$. Then we shift edge a to obtain word $aa\bar{b}\bar{c}\bar{b}cX$. Finally we shift edge b to obtain word $aa\bar{b}\bar{b}ccX \simeq aabbccX$

Optimized Transform E is given in Algo. 3.

Algo 3: Transform E(a,b,c)

1-sew($\alpha_0(d_a)$,$\alpha_2(d_a)$);
1-sew($\alpha_{02}(d_a)$,d_b);
1-sew($\alpha_0(d_b)$,$\alpha_2(d_b)$);
1-sew($\alpha_{02}(d_b)$,d_c);

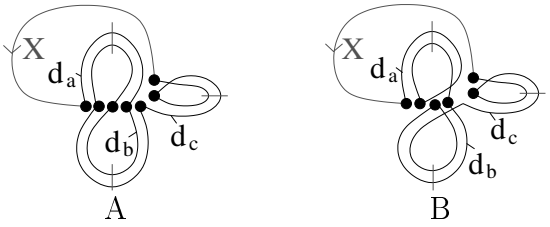

Fig. 7. Optimized Transform E. (A) $2G$-map corresponding to $ab\bar{a}\bar{b}ccX$. (B) $2G$-map obtained after applying Algo. 3 which corresponds to $aabbccX$.

5 Conclusion

This paper highlights the links between polygonal schemata and generalized maps. The canonical polygonal schema can now be directly computed from a $2G$-map, by simply adapting the algorithm described in [3].

The main advantage of our method is that transformations modify the original subdivision and not only a word representing it. It is thus possible to manage additional information carried by the $2G$-map like geometry, e.g. to compute the geometry of generators. Other future works will focus on improving this algorithm according to the specificities of generalized maps.

References

[1] Colin de Verdière, É., "Shortening of curves and decomposition of surfaces," Ph.D. thesis, Université Paris 7 (2003).

[2] Lienhardt, P., *Topological models for boundary representation: a comparison with n-dimensional generalized maps*, Computer-Aided Design **23** (1991), pp. 59–82.

[3] Vegter, G. and C. K. Yap, *Computational complexity of combinatorial surfaces*, in: *Proc. of the* 6^{th} *annual Symposium on Computational Geometry*, p. 102–111.