Computational Complexity of Combinatorial Surfaces

Vegter, Gert; Yap, Chee K.

Published in:
EPRINTS-BOOK-TITLE

Publication date:
1990

(Extended Abstract)

Computational Complexity of Combinatorial Surfaces

Gert Vegter*
Department of Computing Science
University of Groningen
P.O.Box 800, 9700 AV Groningen
The Netherlands

Chee K. Yap*†
Fachbereich Mathematik
Freie Universitaet Berlin
Arnimallee 2-6
1000 Berlin 33
West Germany

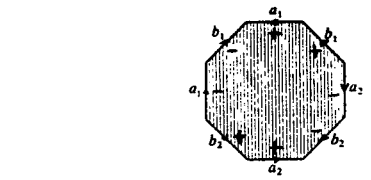
Abstract

We investigate the computational problems associated with combinatorial surfaces. Specifically, we present an algorithm (based on the Brahana-Dehn-Heegaard approach) for transforming the polygonal schema of a closed triangulated surface into its canonical form in $O(n \log n)$ time, where n is the total number of vertices, edges and faces. We also give an $O(n \log n + gn)$ algorithm for constructing canonical generators of the fundamental group of a surface of genus g. This is useful in constructing homeomorphisms between combinatorial surfaces.

1 Introduction

The principal problem in the topology of closed surfaces is the search for the topological invariants of closed surfaces so that we can tell if two arbitrarily given closed surfaces are or are not homeomorphic, see e.g. [2]. It follows then, that the principal *computational* problem is finding efficient methods to compute these invariants and for two closed homeomorphic surfaces, to *construct* homeomorphisms between them. In this paper we present such algorithms. We assume the reader has some familiarity with the topology of surfaces (see e.g., [2, 4]).

It is well-known that a closed surface can be represented as a simple polygon whose edges are labeled by symbols, each symbol occurring exactly twice, and each symbol being given a sign ($\pm$). The surface is obtained by identifying the pair of edges corresponding to a symbol, where the signs indicate whether the identification is with (equal signs) or without (opposite signs) a twist. See Figure 1 for an octogonal schema corresponding to the double-torus. Conversely, any simple polygon in which each edge is labeled by an arbitrarily signed symbol and each unsigned symbol appears exactly twice represents a closed surface. We call this a *polygonal schema* of the surface. Furthermore, each homeomorphic class of surface has a *canonical form* in the polygonal schema. There is a classic algorithm for converting a polygonal schema into its canonical form. The algorithm is essentially due to Brahana (see [1]), although the classification theorem is from Dehn and Heegaard. We call it the Brahana-Dehn-Heegaard (alphabetical order) or BDH algorithm.



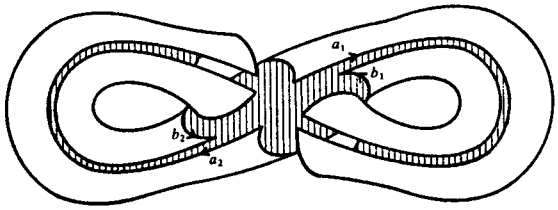


Figure 1: Octogonal schema.

It is not obvious how this algorithm can be implemented in $o(n^2)$ time, where n is the total number of vertices, edges and faces. We will present an algorithm that runs in time $O(n \log n)$.

Our second result shows that, given a triangulated closed surface, we can construct a 'canonical' set of generators for its fundamental group in time $O(n \log n + gn)$, where g is the genus of the surface. Using this, we can cut open the surface into a planar polygon such that its boundary must be identified in a certain canonical way.

*Supported by the ESPRIT II Basic Research Action of the E.C. under contract No.3075 (Project ALCOM).

†On leave from the Courant Institute, New York University. This author is also supported by the German Science Foundation (DFG) and NSF Grants #DCR-84-01898 and #CCR-87-03458.

Such a canonical triangulation can be used to construct a homeomorphism between two homeomorphic surfaces.

It should be realized that our goal in such algorithms is not simply to recognize the type of surface – for this purpose, a triangulation of the polygon followed by a computation of the Euler characteristic suffices to recognize the surface in linear time. Rather, the algorithm yields extra information that can be used to solve topological problems like e.g.

- constructing homeomorphisms, if they exist;
- deciding whether two curves on a surface are homotopic (or regularly homotopic, cf. [5, 8]), and if so, constructing a (regular) homotopy.

The approach via a polygonal schema seems most adequate for decision problems, while the approach via canonical generators seems more appropriate for topological constructions. We refer to the full paper for further details on applications of these methods.

Finally, we note that the present paper and also [5, 8] represent an attempt to study classical problems of combinatorial and algebraic topology from the complexity point of view. We believe that this represents a rich source of new problems for complexity theory and data-structures.

2 Representing Surfaces and Basic Algorithms

By a 'surface' M we shall mean a connected, compact topological surface. Normally, the surface is assumed to be closed (i.e., without boundary); we shall explicitly say so if the surface has boundary. The books [4, 7] would be general references. Computationally, we only deal with combinatorial representations of M. That is, we assume some triangulation S of M, but all manipulations of S can be done purely combinatorially. With this understanding, we will often conveniently lapse into descriptions that assume some embedding of the surface in a suitable Euclidean space. This is mainly for convenience, and the reader should be able to turn our intent into purely combinatorial terms. Let us now make these precise.

We recall that an *abstract simplicial complex* K is a collection of finite non-empty sets that is closed under taking subsets. A *subcomplex* K' of K is an abstract simplicial complex such that $K' \subseteq K$. Each set $A \in K$ is called a *face*. We also call A a *d-face* if the *dimension* of A is d, defined to be one less than the cardinality of A. The dimension of K is the largest dimension of a face of K. The *d-skeleton* of K is the set of all faces of K of dimension at most d. The 0-, 1- and 2- faces are called *vertices*, *edges* and *triangles* of K. By abuse of language, we also call v a vertex of K if $\{v\} \in K$. If $A \subseteq B \in K$ then we say that A is *incident* on B. We can *geometrically realize* K by embedding it in some Euclidean space E: that is, we map vertices v of K into points $f(v)$ in the Euclidean space such that for each d-face $A = \{v_0, \ldots, v_d\}$, the set $\{f(v_0), \ldots, f(v_d)\}$ spans a d-simplex $f(A)$ of E, and two such simplices have disjoint interior. It is known [7] that if K has dimension d then it can be geometrically realized in an Euclidean space of dimension $2d + 1$. In fact dimension 4 suffices for closed surfaces.

A *combinatorial surface* S is an abstract simplicial complex such that each edge is incident on one or two triangles, and for each vertex v, the set X_v of triangles that v is incident upon can be either linearly or circularly ordered so that two triangles of X_v share a common edge iff they are adjacent in this ordering. An edge of S incident on only one triangle is called a *boundary edge*. The surface is *closed* if it has no boundary edges. Note that vertex v is on the boundary of S iff this ordering of X_v above is linear. S is *connected* if its 1-skeleton is a connected graph. Clearly, a combinatorial surface is determined by its set of triangles.

Computationally, we are only interested in case S has a finite number of faces; for definiteness, this will be taken to be the size parameter n. We represent S by its incidence digraph $D(S)$: each face $f \in S$ is associated with a node $D(f)$ of $D(S)$, and there is an arc from $D(f)$ to $D(f')$ iff f is incident on f'. Our computational model will be pointer machines that manipulate such graphs. We assume that our algorithms only manipulate $D(S)$ that represent a *connected surface* S: checking if an arbitrary digraph is of the form $D(S)$ takes $O(n)$ time.

Polygonal Schema.

From a closed surface S we can get in linear time another classic representation of closed combinatorial surfaces, called here the *polygonal schema*. More precisely, a surface is represented by a (simple) polygon P with an even number $2m$ of edges, where $m \leq n$. The edges are labeled by *signed symbols*, $\pm a_1, \ldots, \pm a_m$ such that each *unsigned symbol* (i.e., ignoring the signs in a signed symbol) occurs exactly twice. In this way, we may speak of the *partner* of an edge (or, by abuse of language, the *partner* of a symbol), and refer to two edges or two symbols as being *partnered*. It is convenient to write an 'overbar' over a signed symbol to denote its *complement*: thus if σ denotes $+a_i$ (resp., $-a_i$) then $\bar{\sigma}$ denotes $-a_i$ (resp., $+a_i$). For definiteness (especially in figures), the positively signed edges are directed clockwise around the polygon P.

A polygonal schema P represents a closed surface after identification of each partnered pair of edges, where we take care to respect the orientation of each edge when doing the identification. Since the identity of the actual polygon P is unimportant for this representation, a polygonal schema can be represented by a sequence

$$P = \sigma_1 \sigma_2 \cdots \sigma_{2m}$$

where each σ_i is a signed symbol. We may assume in this representation that we can get from any signed symbol to its partner in constant time.

We will often, by abuse of language, say σ is a signed symbol of P when, strictly speaking, we should speak of an occurrence of σ in P (in any case, there is at most one other occurrence of σ). A partnered pair σ, σ' of signed symbols is *orientable* if the symbols have opposite signs;

otherwise they are *non-orientable*. We say (an occurrence of) a symbol σ is *orientable* if it forms an orientable pair with its partner; otherwise the symbol is *non-orientable*.

There are two well-known *canonical forms* for polygonal schemas. The canonical forms we use will be, for some $m \geq 1$, either (non-orientable case)

$$a_1 a_1 \cdots a_m a_m$$

or (orientable-case)

$$a_1 b_1 \bar{a}_1 \bar{b}_1 \cdots a_m b_m \bar{a}_m \bar{b}_m.$$

As usual, there is the exceptional canonical form $a\bar{a}$ representing the 2-sphere.

It is important to stress that we consider two polygonal schemas P, P' to be equivalent if they are related by the following operations:
1. *Rotation*: If $P = P_1 P_2$ and $P' = P_2 P_1$ for some substrings P_1, P_2.
2. *Complementation*: P' is obtained by negating the sign of each symbol in P.
3. *Reversal*: P' is the reverse of the string P, $P' = P^{rev}$.

Equivalent Vertices.

Suppose we start out with a polygonal schema with $2m$ signed symbols, $P = \sigma_1 \cdots \sigma_{2m}$. Let $v_1, \ldots, v_{2m}$ denote the $2m$ vertices in clockwise order around the polygon implicitly denoted by P, such that σ_i is the label of the edge $e_i = (v_i, v_{i+1})$ for $i = 1, \ldots, 2m$ ($v_{2m+1} = v_1$). We will orient the edge e_i in a clockwise direction if σ_i is positive and counterclockwise otherwise. Recall that two edges are partnered if they have the same unsigned symbol. Let Head(e_i) and Tail(e_i) denote the vertices at the head and tail (respectively) of the oriented edge e_i. So if σ_i is positive then Tail(e_i) $= v_i$ and Head(e_i) $= v_{i+1}$, and otherwise this correspondence is reversed. A basic computational problem is to decide which of these vertices are identified by the schema, and to represent the equivalence classes of these vertices.

For this purpose, it is quite easy to use the classic union-find algorithm, to compute a representation of the equivalence classes in $O(m\alpha(m))$ time, where $\alpha(m)$ is the inverse Ackermann's function. Note that two vertices v_i, v_j are equivalent if they are both heads or both tails of two edges that are partnered in P. The union-find algorithm simply processes the sequence of $2m$ such equivalences implied by the m partnered pairs.

Reduced Polygonal Schema.

We say a polygonal schema is *reduced* if it is either of the form $\sigma\bar{\sigma}$ or else all its vertices belong to one equivalence class. It is not hard to see that the canonical form is automatically reduced. We note the following useful fact:

Fact 2.1
1. A polygonal schema represents a non-orientable surface if and only if it contains at least one non-orientable pair.

Figure 2: Transform A.

2. Two reduced schemas that represent the same closed surface have the same number of symbols.

As a corollary, we can recognize the type of a surface once it is represented by a reduced polygonal schema.

We now introduce two (types of) *reductions* for polygonal schemas:

Transform A. If $P = X\sigma\bar{\sigma}$ (for some *non-empty* sequence of signed symbols X and σ is a signed symbol), then we may replace P by X. See Figure 2.

Transform B. There are two possibilities: (*Orientable case*) If $P = \sigma\tau X\bar{\tau}Y$ (where X, Y are sequences of signed symbols, and σ, τ are signed symbols) then we may replace P by $\rho X\bar{\rho}\sigma Y$ where ρ is a new symbol. See Figure 3. (*Non-orientable case*) If $P = \sigma X\sigma Y$ then P is replaced by $\rho\rho Y\bar{X}$.

It is clear that these transformations are valid. The classic description (see [2, 4]) of reducing an arbitrary polygonal schema (using the transforms A and B above) may, unfortunately, take a quadratic number of steps, not counting the time to find the appropriate partnered pairs for applying the reduction. In the following algorithm, we fix a non-empty equivalence class V_0 and in each iteration do one of the following: (1) discover that we have a 2-sphere already in reduced form, or (2) enlarge the equivalence class V_0 by one, or (3) eliminate the sole member of an equivalence class V (where V may in fact be equal to V_0).

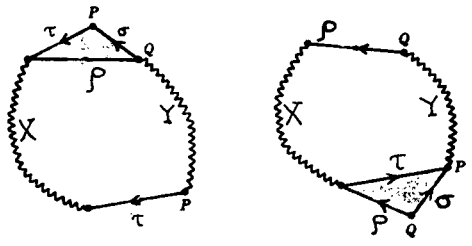


Figure 3: Transform B.

Algorithm for Reducing a Polygonal Schema

Step 1. (*Initialization*)
Pick any vertex v_0 of P and let V_0 be the equivalence class of v_0. In time $O(|V_0|)$, mark all the vertices that are in V_0. Now visit the vertices of P in a clockwise traversal starting with v_0. Let u be the first vertex in this traversal that is not equivalent to v_0.

Step 2. (*Loop*)
At this point, u is found not equivalent to v_0. Let the vertex visited just before u be v; so v is in V_0. There are three cases (steps 2.1, 2.2, 2.3). If $|V_0| = 1$ then go to step 2.1. Otherwise, let σ be the signed symbol that labels the edge e whose endpoints are u and v. Let e' be the next edge following e in the clockwise traversal. If e and e' are partnered, go to step 2.2; otherwise go to step 2.3.

Step 2.1
So v_0 is the sole member of its equivalence class. If we are in the situation $\sigma\bar{\sigma}$, then we return since P is the canonical form of the 2-sphere. Otherwise we may apply Transform A to eliminate v_0. We must then go back to the initialization (Step 1).

Step 2.2
Edges e and e' are partnered. Note that they are necessarily an orientable pair, since otherwise u would be equivalent to v. In this case, u is the only member of its equivalence class.
Since the labels of e, e' are $\sigma, \bar{\sigma}$ respectively, P has the form $X\sigma\bar{\sigma}$ and we may apply transform A. This has two effects: first, the equivalence class of u is completely eliminated. Second, the size of $|V_0|$ is reduced by one.

Step 2.3.
Suppose e and e' are not partnered. So let e' have label τ and let τ' be its partner. Then $P = \sigma\tau X\tau' Y$ and we may apply transform B. One checks that the number of vertices in the equivalent class of V_0 is increased by one at the expense of reducing the size of equivalence class of u by one.

Step 3.
Repeatedly examine the next clockwise vertex (starting from v) looking for one not equivalent to v_0. If we return to v_0, then we are done. Otherwise, we found such a vertex u and go to step 2.

The algorithm clearly halts with a reduced polygonal schema. To see that this algorithm runs in linear time, in particular, we note that the initialization (Step 1) (which may be executed several times) overall takes linear time.

3 An algorithm for canonical form

We assume that the input to the BDH algorithm is a reduced polygonal schema P. Our goal is to transform a sequence of such symbols by repeated reduction steps until we reach the canonical form. Each reduction step will be presented simply as a manipulation of a sequence of symbols.

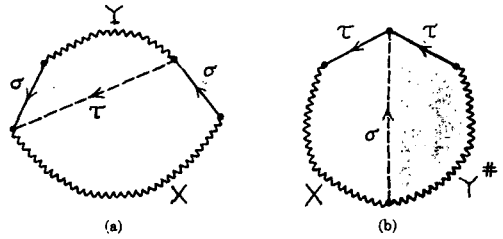


Figure 4: Transform C.

Definition 3.1 *(a) Let σ, τ be two signed symbols of P, and suppose they are not partnered. Then we say that σ, τ are* parallel *if P has the form*

$$P = \sigma \cdots \tau \cdots \tau' \cdots \sigma' \cdots$$

where σ' and τ' are the partners of σ and τ respectively. If σ, τ are not parallel then we say they are crossed.
(b) If σ is a non-orientable symbol of P, then we say σ is converted *if it and its partner are adjacent in P, i.e., $P = \sigma\sigma\cdots$. If σ is an orientable symbol of P, then we say σ is* converted *if for some other orientable pair $\tau, \bar{\tau}$, we have that $P = \sigma\tau\bar{\sigma}\bar{\tau}\cdots$ or $P = \bar{\sigma}\tau\sigma\bar{\tau}\cdots$. A signed symbol is* unconverted *if it is not converted.*
(c) A fan *F is a (not necessarily contiguous) subsequence of P,*

$$F = (\sigma_1, \ldots, \sigma_k)$$

($k \geq 0$) such that for each i, σ_i is an unconverted orientable symbol and the symbols of P counterclockwise from σ_i to σ_{i+1} are all converted, and for all $i \neq j$, σ_i, σ_j are parallel.

We make a simple observation.

Fact 3.2 *If an edge e of a polygonal schema P is labeled by a converted symbol then the two vertices incident on e are equivalent.*

It follows that if all the symbols of P are converted then the P is reduced. The following are well-known transformation steps [2, 4] to convert a non-orientable pair or two crossed orientable pairs:

Transform C. Let σ be a non-orientable symbol of P. If $P = \sigma X\sigma Y$, where X, Y are substrings of P, then we can 'convert' symbol σ into $P = \tau\tau Y^{\#}X$, where τ is a new signed symbol and $Y^{\#}$ denotes the reversal and complementation of Y. (So in fact we replace symbol σ with a converted symbol τ.) See Figure 4.

Transform D. Let σ, τ be a crossed pair of P, where both σ and τ are orientable. If $P = \sigma X\tau Y\bar{\sigma}U\bar{\tau}V$, where X, Y, U, V are substrings of signed symbols of P, then we can convert σ, τ into $P = \rho\xi\bar{\rho}\bar{\xi}UYXV$. See Figure 5.

We will naturally store P as a doubly-linked list L_0. However, we shall super-impose two additional data structures over L_0. First, we have another doubly-linked list

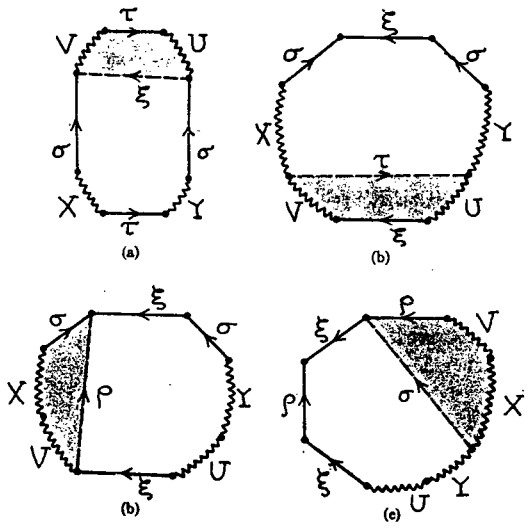


Figure 5: Transform D.

L_1 connecting all the unconverted symbols of P, but preserving their ordering in L_0. Second, we have a balanced binary tree B whose leaves are the members of L_1. Using the binary tree B, we can easily find in logarithmic time the *rank* of any unconverted symbol in logarithmic time. (The rank of any unconverted symbol is its position in the linked-list of unconverted symbols.) The tree B has another important use: in some of our applications, we need the ability to complement the signs of symbols in an entire (contiguous) substring of a polygonal schema. To do this efficiently, we will store the signs of symbols in a distributed fashion, that is to say, we store a sign (± 1) at each internal node of B and the sign of a symbol at a leaf of B is the product of all the signs from the root to that leaf.

In the following, we shall systematically suppress the display of converted symbols—in terms of our data structure for P, we only display the symbols in the list L_1.

Lemma 3.3 *Let $P = \sigma_1\sigma_2\cdots\sigma_k X\bar{\sigma}_k Y$ ($k \geq 1$), where $(\sigma_1,\ldots,\sigma_k)$ is a fan. If P is reduced then X is non-empty and contains an unconverted symbol.*

Proof: If X is empty then the symbols $\sigma_k, \bar{\sigma}_k$ are adjacent to each other. The vertex that these two edges have in common would be in a different equivalence class from the remaining vertices, contradicting the assumption that P is reduced. If X contains only converted symbols, then all the vertices of edges in X are equivalent (by basic properties of converted symbols). But this equivalence class of vertices cannot be equivalent to any other vertices of P, again contradiction. □

Lemma 3.4 *Let $P = \sigma_1\cdots\sigma_k\sigma_{k+1}Z$, $k > 1$, where $(\sigma_1,\ldots,\sigma_k)$ is a fan and the pair σ_k,σ_{k+1} is crossed. After conversion of this pair P is of the form $\sigma_1\cdots\sigma_{k-1}Z'$, where $(\sigma_1,\ldots,\sigma_{k-1})$ is a fan.*

Proof: Before conversion P is of the form (apply a rotation) $\sigma_k\sigma_{k+1}X\bar{\sigma}_kY\bar{\sigma}_{k+1}V\sigma_1\cdots\sigma_{k-1}$,
where YV contains a subsequence of the form $\bar{\sigma}_{k-1}\cdots\bar{\sigma}_1$. Applying transform D to convert the pair σ_k,σ_{k+1} we get (in view of our convention to suppress the display of converted symbols) the sequence $YXV\sigma_1\cdots\sigma_{k-1}$. Since YXV contains $\bar{\sigma}_{k-1}\cdots\bar{\sigma}_1$ as a subsequence, now P is of the form $\sigma_1\cdots\sigma_{k-1}Z'$, where $(\sigma_1,\ldots,\sigma_{k-1})$ is a fan. □

Using these lemmas, we can now present the algorithm for the orientable case.

Input: A reduced schema P for an orientable surface.
Output: A canonical form schema for the orientable surface (plus the transformation sequence).
While there exists unconverted symbols **do**
  Start a fan (σ_1).
  While current fan is non-empty **do**
    1. Let current fan be $(\sigma_1,\ldots,\sigma_k)$, and $P = \sigma_1\cdots\sigma_k X\bar{\sigma}_k Y$.
    2. Let σ_{k+1} be the next unconverted symbol in X.
    3. If σ_k,σ_{k+1} are parallel then we extend the fan by σ_{k+1}.
    4. Otherwise σ_k,σ_{k+1} are crossed and we convert σ_k,σ_{k+1} as given by Lemma 3.4.
  end {while current fan ...}
end {while there exists unconverted ...}

Data Structures. We assume that the beginning σ_1 of the fan $(\sigma_1,\ldots,\sigma_k)$ is the left most leaf of the binary search tree B. We can easily decide if a new unconverted element σ_{k+1} is going to extend the fan or not by computing the rank of the partner of σ_{k+1} and comparing it to the rank of the partner of σ_k (already computed). To convert σ_k,σ_{k+1}, we must reverse the order of two contiguous subsequences of L_1, which can be done in logarithmic time.

Non-orientable case. It is not hard to extend the algorithm to the non-orientable case. As usual, we grow a fan $F = (a_1,\ldots,a_k)$. Our initial goal is to convert all symbols of P. If the next unconverted symbol σ_{k+1} is orientable, we proceed as before. If it is non-orientable, then we convert the non-orientable pair as in the fact (part (a)) above. Note that in general, we have $P = \sigma_1\cdots\sigma_k\sigma_{k+1}X\bar{\sigma}_{k+1}Y$ and after conversion we have $P = \sigma_1\cdots\sigma_k X^{\#}Y$, where $X^{\#}$ denotes the complemented reversal of X.

Finally, we must repeatedly do the standard trick of converting a projective plane and a torus $(\sigma_1\sigma_2\bar{\sigma}_1\bar{\sigma}_1\sigma_3\sigma_3)$ into three projective planes $(\sigma_4\sigma_4\sigma_5\sigma_5\sigma_6\sigma_6)$. Specifically, a string can be converted as follows:

$$\sigma_1\sigma_1 X\sigma_2\sigma_3\bar{\sigma}_2\bar{\sigma}_3 Y \longrightarrow \sigma_4\sigma_4\sigma_5\sigma_5\sigma_6\sigma_6 XY.$$

We again refer to the full paper for details.

Bearing in mind that $m = O(n)$ the result of this section and the previous one is now:

Theorem 3.5 *A closed combinatorial surface with n triangles can be converted into a canonical polygonal schema*

for the surface in $O(n \log n)$ time. The algorithm also produces a list of the $O(n)$ transformations steps.

4 Construction of a Canonical Set of Generators

First we introduce some terminology. Let M be a surface. A *triangulation* of M is a set $\mathcal{T}$ of pairwise disjoint sets such that (i) their union is equal to M, (ii) each set in $\mathcal{T}$ is homeomorphic to a point (vertex), open linesegment (edge) or open disc (triangle), respectively, (iii) each edge is incident on two triangles and each triangle has exactly three edges incident on it. It is clear that if we view $\mathcal{T}$ abstractly, it defines some abstract simplicial complex. We call the pair $(M, \mathcal{T})$ a *triangulated surface*. The use of $M, \mathcal{T}$ is just convenient for discussion: computationally, we only manipulate $\mathcal{T}$ as a combinatorial surface without refering to M. A (simple) *piecewise linear curve* (PL-curve for short) on $(M, \mathcal{T})$ is a curve in M that intersects each triangle T of $\mathcal{T}$ in a finite collection of pairwise disjoint chords of T; we call these chords *segments* of the PL-curve. A triangulation $\mathcal{T}'$ is a *refinement* of $\mathcal{T}$ if each face (edge) of $\mathcal{T}$ is a union of faces, edges and vertices (resp. edges and vertices) of $\mathcal{T}'$, and each vertex of $\mathcal{T}$ also is a vertex of $\mathcal{T}'$. An *edge-path* in $(M, \mathcal{T})$ is a curve in M that is a union of vertices and edges of $\mathcal{T}$.

Definition 4.1 *A set of closed curves on a surface M of genus g is a* canonical set of generators *if the following conditions are satisfied:*
1. the curves have a common base-point;
2. the curves are pairwise disjoint, except for their common base-point;
3. the complement of the union of the curves is a disc D;
4. the curves appear on the boundary of the disc D in canonical order *(in the sense of polygonal schema). More precisely: we give each curve a symbol a_i and an arbitrary direction. Then each curve appears on the boundary of D exactly twice, with the induced direction. Clearly this gives us a polygonal schema, which we require to be in canonical form. In particular, if M is orientable, then the canonical set of generators has $2g$ curves giving rise to the schema*

$$a_1 a_2 \bar{a}_1 \bar{a}_2 \ldots a_{2g-1} a_{2g} \bar{a}_{2g-1} \bar{a}_{2g}.$$

If M is not orientable, then it has g curves giving rise to the schema

$$a_1 a_1 a_2 a_2 \cdots a_g a_g.$$

The main result of this section is:

Theorem 4.2 *There is a canonical set of generators for the triangulated surface $(M, \mathcal{T})$ which can be realized as edge-paths of a refinement of $\mathcal{T}$ of size $O(gn)$, where n is the size of the triangulation $\mathcal{T}$, and g is the genus of M. This set of generators can be computed in $O(gn)$ time.*

We note here that the method presented in the previous sections is not suitable for the construction of a canonical set of generators that can be realized as edge-paths of a refinement of $\mathcal{T}$. It turns out that straightforward application of the method may cause a prohibitive increase in the number of triangles.

In the rest of this extended abstract we restrict ourselves to the orientable case of this theorem. Henceforth, we assume the surface M orientable.

The key to the proof of this theorem is the following result, whose proof will be given in the remainder of this section.

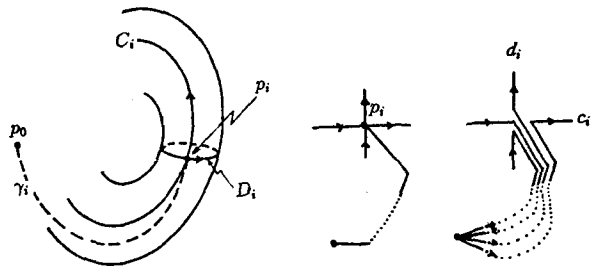


Figure 6: Construction of Pair of Generators

Lemma 4.3 *There is a refinement $\mathcal{T}'$ of $\mathcal{T}$ of size $O(gn)$ with the following properties.*
1. There are two families $\mathcal{C} = \{C_i | 1 \leq i \leq g\}$ and $\mathcal{D} = \{D_i | 1 \leq i \leq g\}$, consisting of simple closed edge-paths of $\mathcal{T}'$ with
(i) C_i and D_i intersect in a single vertex p_i of $\mathcal{T}'$;
(ii) The sets $C_i \cup D_i$, $1 \leq i \leq g$, are pairwise disjoint.
2. There is a vertex p_0 of $\mathcal{T}'$, not on any of the curves C_i and D_i, $1 \leq i \leq g$, which is connected to p_i by an edge-path γ_i, $1 \leq i \leq g$, in $\mathcal{T}'$. These so called approach-paths are pairwise disjoint, except for their common base-point p_0, and are disjoint from $\cup_{i=1}^{g}(C_i \cup D_i)$, except for their common end-points p_i, $1 \leq i \leq g$.
3. The complement of $\cup_{i=1}^{g}(\gamma_i \cup C_i \cup D_i)$ in the surface M is a disc.

Moreover, the refinement $\mathcal{T}'$, as well as the edge-paths C_i, D_i and γ_i, $1 \leq i \leq g$, can be constructed in $O(gn)$ time.

To prove Theorem 4.2 using lemma 4.3 assume the approach paths $\gamma_1, \cdots, \gamma_g$ appear in clockwise order around p_0. Now modify the curves C_i and D_i, $1 \leq i \leq g$, into c_i and d_i, respectively, as suggested by Figure 6. Conceptually speaking the point of intersection of C_i and D_i is moved along γ_i from p_i to p_0, without changing the topology of the complement. In fact curves c_i and d_i are homotopic to $\gamma_i C_i \overline{\gamma_i}$ and $\gamma_i D_i \overline{\gamma_i}$, respectively. □

Decomposition of a Triangulated Surface

Let the set of triangular faces $T_1, \ldots, T_f$ of $\mathcal{T}$ be ordered in such a way that T_i shares an edge with at least one

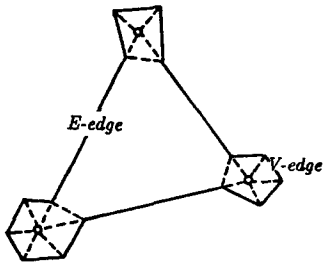


Figure 7: An Augmented Triangle.

triangle T_j, $1 \leq j < i$. The idea is to assemble the surface M starting with T_1, and successively attaching $T_2, \cdots, T_f$. However, the relative boundary of the set $M_i := \bigcup_{j=1}^{i} T_j$ may be 'pinched' at vertices whose star (in M) intersects M_i in non-adjacent triangles.

To get a *regular decomposition*, i.e. one for which each of the sets M_i is a regular topological manifold with boundary, we introduce a small 'substar' in the star of each vertex v. (Recall that the star of a vertex is the closure of the union of all triangles containing the vertex in their boundary.) This is achieved by introducing a new vertex on each edge incident upon v. Two new vertices are connected by a new edge if the edges containing them are incident upon the same face of $\mathcal{T}$. The subdivision thus obtained is denoted by $\mathcal{T}_0$. For a triangular face T_i of $\mathcal{T}$ the union of T_i and the three substars associated with the vertices of T_i is denoted by T_i', which will be referred to as an *augmented triangle*. See Figure 7. The substars will be called *augmented vertices*. The edges of T_i' belonging to a substar will be called *V-edges*, the other (three) edges will be called *E-edges*.

Now the chain

$$\mathcal{M} : \quad M_1' \subset M_2' \subset \ldots \subset M_f' = M, \qquad (1)$$

with $M_i' = \cup_{j=1}^{i} T_j'$, is a *regular* decomposition of the surface M.

The process of adding an augmented triangle T_{i+1}' to M_i' is called an *extension*. We distinguish four types of extensions.

Regular Extension. In this case T_{i+1}' and M_i' either share exactly one E-edge and two augmented vertices, or exactly two E-edges and three augmented vertices. In this case both the Euler characteristic and the number of boundary components of M_i' and M_{i+1}' are equal.

Closure. In this case T_{i+1}' and M_i' share exactly three E-edges and three augmented vertices. The Euler characteristic of M_i' is one smaller than that of M_{i+1}'. The boundary component containing the shared E-edges 'disappears'.

Splitting. Now T_{i+1}' and M_i' share exactly three augmented vertices and one E-edge, whilst the shared E-edge and the augmented vertex, not incident upon it, are in the same component of the boundary of M_i'. The Euler characteristic of M_i' is one larger than that of M_{i+1}', and the boundary component containing the shared E-edge is split into two parts.

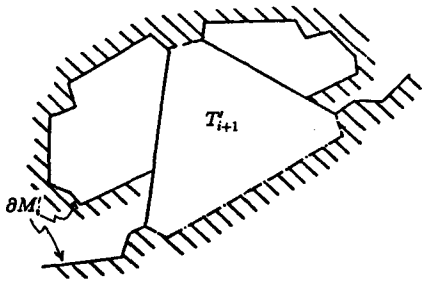


Figure 8: Extension of type Join

Join. In this case T_{i+1}' and M_i' share exactly three augmented vertices and one E-edge, whilst the shared E-edge and the augmented vertex, not incident upon it, are in different components of the boundary of M_i'. The Euler characteristic of M_i' is one larger than that of M_{i+1}', whilst the boundary component containing the shared E-edge is joined with the boundary component containing the augmented vertex not incident upon this E-edge, see Figure 8.

Lemma 4.4 *There are exactly g joins in the chain $\mathcal{M}$.*

The Decomposition Graph

With a decomposition $\mathcal{M}$ we now associate the so called *decomposition graph* $G(\mathcal{M})$. To avoid confusion *nodes* and *arcs* will refer to the *abstract* graph $G(\mathcal{M})$, while *vertices* and *edges* refer to the triangulation $\mathcal{T}$.

Consider boundary components B_i and B_j of M_i' and M_j', respectively, with $1 \leq i \leq j \leq f$. B_i and B_j are called *equivalent* if there is a sequence B_l, $i \leq l \leq j$, where B_l is a boundary component of M_l', such that for $i \leq l < j$ either $B_l = B_{l+1}$, or B_{l+1} is obtained from B_l by a Regular Extension.

The set of arcs of $G(\mathcal{M})$—which is a directed graph—is in one-one correspondence with the set of equivalence classes of boundary components. The arc associated with a boundary component B is denoted by $\overline{B}$. These arcs are connected according to the following rules.

1. If the boundary component B is split into B_1 and B_2, then the final node of arc $\overline{B}$ is identified with the initial nodes of arcs $\overline{B_1}$ and $\overline{B_2}$.
2. If the boundary components B_1 and B_2 are joined into B, then the final nodes of arcs $\overline{B_1}$ and $\overline{B_2}$ are identified with the initial node of $\overline{B}$.

It is not hard to see that the undirected version of the decomposition graph $G(\mathcal{M})$ is connected. Note that $G(\mathcal{M})$ is not unique, since there are many different ways in which to decompose the surface.

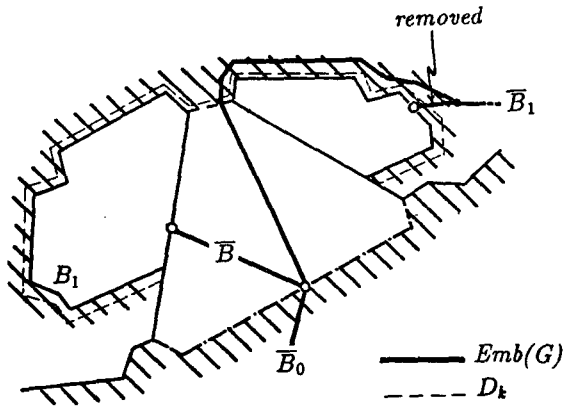


Figure 9: Extending the Embedding of G at a Join.

We will occasionally use the notation G as an abbreviation of $G(\mathcal{M})$.

Lemma 4.5 *The space of cycles of $G(\mathcal{M})$ has rank g.*

Embedding $G(\mathcal{M})$ in $\mathcal{M}$ as a one-dimensional subcomplex

We will construct an embedding of the decomposition graph $G = G(\mathcal{M})$ by extending its image in M upon each addition of a augmented triangle.

An *active boundary component* is a connected component of the boundary of M', where M' is the union of the augmented triangles processed so far. An *active arc* is any arc of G corresponding to an active boundary component. Note that the image of an active arc is 'under construction'. This construction will be such that once an arc has become 'inactive', it will never get active again. The part of G currently embedded in M' is denoted by $Emb(G, M')$, or $Emb(G)$ for short.

Our algorithm maintains the following invariant.
(*) The image of any active arc of G intersects the corresponding active boundary in exactly one point, which lies on an E-edge of T_0. Moreover, one of these E-edges that contain a point of $Emb(G)$ is incident on the next augmented triangle which extends M'. Call this E-edge the *glue edge*.

It is not hard to establish (*) initially: the boundary of T_1' corresponds to an arc, whose image is one of the E-edges of T_1' with some point on an E-edge (which is the next glue-edge). The full version of the paper explains how (*) can be restored rather trivially after a Regular Extension, a Closure or a Splitting. It turns out that $Emb(G)$ is extended with $O(1)$ edges in these cases.

So consider a Join of boundary components B_0 and B_1 into B. Figure 9 indicates how the image of arc $\overline{B_0}$ is connected to the image of arc $\overline{B_1}$ by a PL-curve of $O(n)$ segments running 'parallel' to part of B_1 (disregard D_k for now). The first segments of the curve lies in the hexagonal face of T_{i+1}'. Note that the last edge of the image of $\overline{B_1}$ (prior to extension) is removed in order to restore (*).

Note that this construction obtains $Emb(G)$ as a union of simple PL-curves which are disjoint except possibly at their endpoints.

Lemma 4.6 *There is a decomposition $\mathcal{M}$ of the surface (M, T) whose decomposition graph $G(\mathcal{M})$ can be embedded as a one-dimensional subcomplex of a refinement T' of T of size $O(gn)$. Moreover this refinement contains a subcomplex that constitutes a (tubular) neighborhood $\hat{G}(\mathcal{M})$ of $G(\mathcal{M})$. The refinement, the embedding and the (tubular) neighborhood can be constructed in $O(gn)$ time.*

The graph $G(\mathcal{M})$ contains a spanning tree T_G with the following properties:
1. the image of T_G in M consists of $O(n)$ edges;
2. the image of each non-tree arc of $G(\mathcal{M})$ consists of $O(n)$ edges.
Moreover, we can force $\hat{G}(\mathcal{M})$ to be planar by a suitable ordering of the sequence of triangles defining $\mathcal{M}$.

Proof: The construction of the embedding of $G = G(\mathcal{M})$ in the surface M has already been described. The PL-curves forming $Emb(G)$ gives us a natural refinement T'' of T.

To obtain a tubular neighborhood of G consider a vertex v of $Emb(G)$. On each edge of T'' incident upon v, but not contained in $Emb(G)$, we insert a new vertex near v. Connecting these vertices in the obvious way we obtain a refinement T' of T'' that contains a tubular neighborhood $\hat{G}(\mathcal{M})$ of $Emb(G)$ in M, see Figure 10.

Finally we focus on the construction of the spanning tree T_G. Each non-tree arc of $G(\mathcal{M})$ automatically has $O(n)$ edges since *a fortiori* every arc of G consists of $O(n)$ edges. It remains to ensure that T_G has a total number of $O(n)$ edges.

As noted earlier, in each extension which is not a Join, the number of edges added to the image of G is $O(1)$. Now consider a Join of boundary components B_0 and B_1 into B, cf. Figure 9. In this case the image of arc $\overline{B_1}$ is extended with $O(n)$ edges. However, removing arc $\overline{B_1}$ from G does not disconnect the subgraph of G corresponding to the part of $Emb(G)$ currently constructed.

Proceeding this way for each join we remove exactly g arcs from G, cf. Lemma 4.4. The remaining subgraph T_G is connected and contains all nodes of G. It is seen that the number of arcs of T_G is one less than the number of

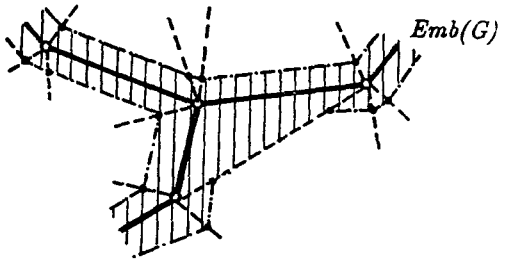


Figure 10: Construction of a Tubular Neighborhood.

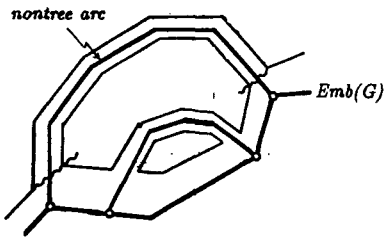


Figure 11: Strip associated with a nontree Arc.

nodes of T_G. Therefore T_G is a (spanning) tree, which obviously has the required properties.

The image $Emb(G)$ of G is constructed in $O(gn)$ time. The edges of $Emb(G)$ subdivide the faces of $\mathcal{T}_0$. In $O(gn)$ time we construct a refinement $\mathcal{T}'$ of $\mathcal{T}_0$ such that $Emb(G)$ is a union of edges and vertices of $\mathcal{T}'$.

To force $\hat{G}(\mathcal{M})$ to be planar, we basically choose the sequence of triangles that define $\mathcal{M}$ in such a way that the following invariant holds: *there does not exist a path in the complement of M that connects two boundary components of M'.* To ensure this, immediately after a Splitting, we will spend $O(n)$ time to find out if there is a sequence of Regular Extensions that ends with a Join that reconnects the two boundary components emerging from the latest Splitting. If so, we will choose such a sequence of Regular Extensions to restore our invariant. If not, we proceed as normal. One shows that we will have to do this at most g times. The full paper will show these in greater detail. □

Construction of the Family $\mathcal{D} = \{D_1, \cdots, D_g\}$

The tubular neighborhood $\hat{G} = \hat{G}(\mathcal{M})$, constructed in the previous section, is a topological disc with g strips (discs) attached to it. Each strip is attached along two disjoint segments in the boundary of the disc. To see this, imagine that $\hat{G}$ is cut along line segments near the endpoints of the image of each nontree arc of $G = G(\mathcal{M})$, see Figure 11, and remove the strips thus cut out. What remains is a tubular neighborhood of the spanning tree T_G, see Lemma 4.6, which is therefore homeomorphic to a disc. Note that the image of G is a deformation retract of $\hat{G}$, hence

$$\chi(\hat{G}) = \chi(G) = 1 - g. \qquad (2)$$

The family $\mathcal{D}$ (see Lemma 4.3) will be obtained by constructing a 'bridge' on each strip associated with a nontree arc.

Recall from the proof of lemma 4.6 that the final node of each nontree arc of G is a join. So consider an extension in which active boundary components B_0 and B_1 are joined into B, see Figure 9, where arc $\overline{B_1}$ is a nontree arc. In view of invariant $(*)$ the curve D_k, obtained by deforming B_1 slightly into the interior of $\cup_{j=1}^{i} T_j'$, intersects the image of arc $\overline{B_1}$ in exactly one point

Lemma 4.7 *Let $\epsilon_1, \cdots, \epsilon_g$ be the nontree arcs of the decomposition graph G. There is a refinement T'' of T' of size $O(gn)$, and a family $\mathcal{D} = \{D_1, \cdots, D_g\}$ of pairwise disjoint piecewise linear curves, each consisting of $O(n)$ edges such that:*

1. *Each D_i intersects $Emb(G)$ in a single point, which is a vertex of T'' lying on the image of ϵ_i;*
2. *$M \setminus (\hat{G} \cup (\cup_{i=1}^{g} D_i))$ is homeomorphic to a disc.*

(Here $\hat{G}$ is the tubular neighborhood of $G(\mathcal{M})$).
Both the refinement T'' and the family $\mathcal{D}$ can be constructed in $O(gn)$ time,

Proof: It remains to prove the second property. A closer look at the construction of $Emb(G)$ reveals that the set $M \setminus Emb(G)$, and hence $M \setminus \hat{G}$, is connected. It is not hard to see that $M \setminus (\hat{G} \cup (\cup_{i=1}^{g} D_i))$ is connected as well (see the full paper for details). Attaching D_i to $\hat{G} \cup (\cup_{j<i} D_j)$ corresponds to attaching a line segment, and therefore increases the Euler characteristic of the complement by one. (The line segment that is attached occurs once in the one-skeleton of $M \setminus (\hat{G} \cup (\cup_{j<i} D_j))$ and twice in the boundary of $M \setminus (\hat{G} \cup (\cup_{j \le i} D_j))$.) In view of (2) the Euler characteristic of the set $M \setminus \hat{G}$ is equal to

$$\chi(M) - \chi(\hat{G}) = (2 - 2g) - (1 - g) = 1 - g.$$

Hence $M \setminus (\hat{G} \cup (\cup_{i=1}^{g} D_i))$ (the result of attaching g line segments) has Euler characteristic equal to 1. Since this set is connected, it is homeomorphic to a disc. □

Construction of the Family $\mathcal{C} = \{C_1, \cdots, C_g\}$

In our construction of the family $\mathcal{C}$ we use small tubular neighborhoods $\hat{D}_j$ of D_j, $1 \le j \le g$, that are obtained by attaching a strip to s_j, see Figure 12. It is not hard to check that $M \setminus \hat{G} \cup (\cup_{j=1}^{g} \hat{D}_j)$ is also homeomorphic to a disc.

Lemma 4.8 *There is a family $\mathcal{C}$ of pairwise disjoint PL-curves $C_1, \cdots, C_g$ contained in $\hat{G} \cup (\cup_{i=1}^{g} \hat{D}_i)$ with the following properties.*

1. *Each D_i intersects C_i in a single point p_i, $1 \le i \le g$.*
2. *The sets $C_i \cup D_i$, $1 \le i \le g$, are pairwise disjoint.*
3. *The curve C_i, $1 \le i \le g$, consists of $O(n)$ segments, and is constructed in $O(n)$ time.*
4. *$M \setminus \cup_{i=1}^{g} (C_i \cup D_i)$ is connected.*
5. *The points p_i can be connected to a point q_i on the boundary of $\hat{D}_i$ by a single segment.*

Sketch of the Proof: Recall that each nontree arc ϵ_i of G is associated with a strip s_i, $1 \le i \le g$, attached to a disc D_G (viz. a tubular neighborhood of T_G). Curve C_i will intersect strip s_i along the image of ϵ_i. Note that this part of C_i intersects D_i in a single point p_i that can be connected to a point q_i on $\partial \hat{D}_i$ by a single edge.

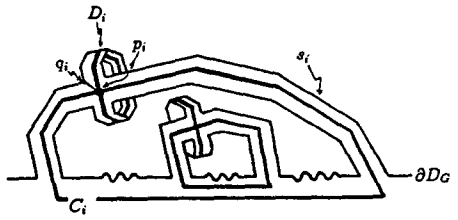


Figure 12: Construction of the family $\mathcal{C}$.

The problem now is to connect the two points of intersection of the partial curve $C_i \cap s_i$ and the boundary of the disc D_G, for $i = 1, \cdots, g$, by curves that are pairwise disjoint. See Figure 12. This is now easy in view of the planarity of $\hat{G}$. Clearly each C_i can be made to be PL-curve with $O(n)$ segments.

For the fact that $M \setminus \cup_{i=1}^{g}(C_i \cup D_i)$ is connected, we only have to note that the boundary of $\hat{G} \cup (\bigcup_{i=1}^{g} \hat{D}_i)$ is connected. □

Lemma 4.9 *$M \setminus \cup_{i=1}^{g}(C_i \cup D_i)$ is homeomorphic to a 2-sphere with g holes.*

Proof: Let l_i be the total number of edges of the curves C_i and D_i. Cutting M along $C_i \cup D_i$ changes the number of vertices from $l_i - 1$ into $2l_i$, and the number of edges from l_i into $2l_i$. Therefore cutting M along $C_i \cup D_i$ increases the Euler characteristic by 1. Hence $\chi(M \setminus \cup_{i=1}^{g}(C_i \cup D_i)) = 2 - g$. Since the set $M \setminus \cup_{i=1}^{g}(C_i \cup D_i)$ is connected, it is homeomorphic to a 2-sphere with g holes (apply the Classification Theorem for surfaces with boundary, cf. [4]). □

Construction of the Approach Paths $\gamma_1, \cdots, \gamma_g$

Let q_i be a point on the boundary of $\hat{G} \cup (\cup_{i=1}^{g} \hat{D}_i)$ near the point of intersection of C_i and D_i. Since $M \setminus (\hat{G} \cup (\cup_{i=1}^{g} \hat{D}_i))$ is a disc, we can connect some interior vertex p_0 to each of the points q_i by pairwise disjoint edge-paths $\gamma_1', \cdots, \gamma_g'$ of some refinement of the current triangulation of the set $M \setminus (\hat{G} \cup (\cup_{i=1}^{g} \hat{D}_i))$. It is not hard to check that these approach-paths can be constructed in such a way that each of them consists of $O(n)$ edges. Connecting q_i and $C_i \cap D_i$ we obtain edge-paths $\gamma_1, \cdots, \gamma_g$ as stated in Lemma 4.3.

Cutting $M \setminus \cup_{i=1}^{g}(C_i \cup D_i)$ along the approach-paths $\gamma_1, \cdots, \gamma_g$ increases the Euler characteristic by $g - 1$. Therefore
$\chi(M \setminus \cup_{i=1}^{g}(C_i \cup D_i \cup \gamma_i)) = (2 - g) + (g - 1) = 1$.

Since the set $M \setminus \cup_{i=1}^{g}(C_i \cup D_i \cup \gamma_i)$ obviously is connected, it is homeomorphic to a disc. This completes the proof of lemma 4.3.

5 Final Remarks

We have presented two basic algorithms in what could be called 'computational topology'. The algorithms can be used to decide upon the existence of homeomorphisms between surfaces, homotopies between closed curves on surfaces, or even to construct such objects. It is not hard to extend the algorithms to surfaces with boundary. In the full paper we will describe the details for the non-orientable case.

We can show that the $O(gn)$ bound cannot be avoided in the sense that there are surfaces such that any set of canonical generators represented as a set of PL-curves has $\Omega(gn)$ segments.

As stated in the introduction, there are many problems in combinatorial topology that deserve to be treated from the computational point of view. Among these are problems in surface topology, but also knot theory provides interesting questions (e.g. is it tractable to decide whether a given polygonal knot in three-space is trivial; this problem is decidable). Another interesting problem is the complexity of deciding isomorphism of two abstract simplicial complexes (it is in NP but is it harder than graph isomorphism?). We hope to deal with these problems in future work.

References

[1] H. R. Brahana. Systems of circuits on two-dimensional manifolds. *Ann. Math.*, 23:144–68, 1922.

[2] Maurice Fréchet and Ky Fan. *Initiation to Combinatorial Topology.* Prindle, Weber and Schmidt, Inc., Boston, Massachusetts, 1934. Translated from French, with notes by Howard E. Eves.

[3] F. Harary. *Graph Theory.* Addison-Wesley, Reading, Mass., 1969.

[4] William S. Massey. *Algebraic Topology: An Introduction.* Graduate Texts in Mathematics, Vol. 56. Springer-Verlag, 1977.

[5] Kurt Mehlhorn and Chee K. Yap. Constructive hopf's theorem: or how to untangle closed planar curves. *Proceedings ICALP, Springer Lecture Notes in Computer Science*, 317:410–423, 1988.

[6] J. R. Munkres. *Elements of Algebraic Topology.* Addison-Wesley, Menlo-Park, California, 1984.

[7] J. Stillwell. *Classical Topology and Combinatorial Group Theory.* Springer-Verlag, New York, 1980.

[8] Gert Vegter. Kinkfree deformations of polygons. *Proc. 5th Ann. Sympos. Comput. Geom.*, pages 61–68, 1989.