Discrete Comput Geom 14:93–110 (1995)

Discrete & Computational
Geometry
© 1995 Springer-Verlag New York Inc.

A New Technique To Compute Polygonal Schema for 2-Manifolds with Application to Null-Homotopy Detection

T. K. Dey[1] and H. Schipper[2]

[1] Department of Computer Science and Engineering, I.I.T.,
Kharagpur, India 721302
dey@cse.iitkgp.ernet.in

[2] Computing Science, Groningen University,
9700 AV Groningen, Netherlands

Abstract. We provide a new technique for deriving optimal-sized polygonal schema for triangulated compact 2-manifolds without boundary in $O(n)$ time, where n is the size of the given triangulation T. We first derive a polygonal schema P embedded in T using Seifert–Van Kampen's theorem. A reduced polygonal schema Q of optimal size is computed from P, where a surjective map from the vertices of P is retained to the vertices of Q. This helps detecting null-homotopic (contractible to a point) cycles. Given a cycle of length k, we determine if it is null-homotopic in $O(n + k \log g)$ time and in $\Theta(n + k)$ space, where g is the genus of the given 2-manifold. The actual contraction for a null-homotopic cycle can be computed in $\Theta(nk)$ time and space. This is a considerable improvement over the previous best-known algorithm for this problem.

1. Introduction

In recent years a new focus has developed in studying the algorithmic aspects of topology [1], [3], [4], [5], [6], [8], [9], a well-developed branch of mathematics. This emergent field has been called "computational topology" [5], [9]. It is generally recognized that a vast repository of topological problems exists which have not been studied extensively from an algorithmic point of view. This paper deals with the problem of computing polygonal schema, an efficient representation of 2-manifolds, from their triangulations. An elegant solution to this problem provides an improved algorithm for detecting null-homotopic cycles on compact 2-manifolds.

A 2-manifold $\mathcal{M}$ can be represented by a polygon P (polygonal schema) with an even number of edges such that when edges are appropriately identified, we get back $\mathcal{M}$. Given a triangulation T of $\mathcal{M}$, Vegter and Yap [9] designed an algorithm that derives a polygonal schema (*canonical*) of optimal size whose edges belong to the edges of a refined triangulation of T. This method subdivides the edges of T into $O(g)$ subedges and increases the size of the triangulation to $O(gn)$. A lower bound of $\Omega(gn)$ is known for deriving such a canonical polygonal schema for 2-manifolds. In that respect the algorithm of Vegter and Yap is optimal. However, in applications we may not need such polygonal schema. As shown in this paper, we can derive a polygonal schema of optimal size in $O(n)$ time from T without refining it. This polygonal schema is not embedded in a refinement of T but retain a surjective mapping from the vertices of another polygonal schema embedded in T to its vertices. This fact is used to detect null-homotopic cycles on $\mathcal{M}$. Our method of computing the polygonal schema is also simpler than Vegter and Yap's method. We believe that this method will find more applications in other algorithms.

Given a cycle C on a 2-manifold, we say that C is null-homotopic if it can be contracted to a single point. Detecting the null-homotopy of a given cycle in a topological space is a century-old problem, known as the *contractibility problem* in topology [7]. A related problem involves determining if two given cycles can be continuously deformed to one another. It turns out that a solution to the contractibility problem also provides a solution to this problem.

The contractibility problem for 2-manifolds was first solved from a mathematical point of view during the 1880s. It was established that a cycle is null-homotopic if and only if its corresponding curve on the *universal covering space* is closed. Universal covering space is a collection of polygons (polygonal schema) appropriately attached to give a tessellation of the plane. In [5] Schipper used this result to give an $O(g^2k + gn)$ time and space algorithm to detect the null-homotopy of a given cycle. Here k is the length of the given cycle. In his algorithm Schipper derives the canonical polygonal schema using the algorithm of Vegter and Yap [9]. Using our polygonal schema computation and a clever modification of Schipper's algorithm, we design an algorithm that runs in $O(n + k \log g)$ time and in $\Theta(n + k)$ space. If the given cycle is null-homotopic, the actual contraction can be computed in $O(nk)$ time, which is worst-case optimal. This is a considerable improvement over Schipper's algorithm and is almost optimal. We should mention that Schipper's algorithm does not remain linear in k and takes $O(n + k^2)$ time when $g = 1$ for torus (orientable) and $g = 2$ for klein bottles (nonorientable). We treat these cases separately and give $\Theta(n + k)$ optimal algorithms for them.

We describe necessary concepts in homotopy and fundamental groups of topological spaces in Section 2. In the third section we explain the polygonal representation (polygonal schema) of 2-manifolds. In Section 4 we describe a technique to derive a reduced polygonal schema from the given triangulation of a 2-manifold using Seifert–Van Kampen's theorem. In the fifth section we detail the algorithm for null-homotopy detection, and we conclude in Section 6.

2. Homotopy and Fundamental Group

Let $f\colon T_1 \to T_2$ and $g\colon T_1 \to T_2$ be two maps between the two connected topological spaces T_1 and T_2. These two maps are called homotopic if there is a continuous function $h\colon [0,1] \times T_1 \to T_2$ such that $h(0,x) = f(x)$ and $h(1,x) = g(x)$. We can interpret h as a deformation process that transforms f to g in a continuous manner. We are interested in the case when $T_1 = S^1$, the unit circle. Then f and g are closed curves, also called cycles. Let C_1, C_2 be two cycles that begin at a common point, say p. The product operation, "$\cdot$", is defined on them as $C_3 = C_1 \cdot C_2$, where C_3 is the cycle beginning at p, then going around C_1 followed by a traversal around C_2 which finishes at p. The inverse C^{-1} of a cycle C is the cycle lying on top of C, but with opposite orientation.

Given a fixed origin p for the cycles on T_2, we call C, C' equivalent if there is a homotopy between them which keeps p fixed. The equivalence class of C is denoted by $[C]$. The product operation "$\cdot$" extends naturally to equivalence classes as $[C_1]\cdot[C_2] = [C_1 \cdot C_2]$.

These equivalence classes form a *group* under the product operation. The identity element 1 is represented by the equivalence class of the point cycle p, and $[C]\cdot[C^{-1}] = [C \cdot C^{-1}] = 1$ giving $[C]^{-1} = [C^{-1}]$. This group, denoted $\pi_1(T_2)$, is called the *fundamental group* of the topological space T_2. It turns out that this group is independent of the choice of the origin p and is an invariant property of the underlying space.

Fact 2.1. A cycle C is contractible or null-homotopic if and only if $[C] = 1$ in π_1 [7].

We use the following concepts of group theory in the next sections. Let G be a group with the product operation "$\cdot$". A set of elements $X = \{g_1, g_2, \ldots\}$ of G is called a *generator set* if any nonidentity $g \in G$ can be written as $g = g_1' \cdot g_2' \cdots g_k'$ for some $k \geq 1$ and $g_i' \in \{g_1, g_2, \ldots\}$. In other words all elements of G are *generated* by X. A *word* is a concatenation of elements of G under the product operation. For example $w = g_1 \cdot g_2 \cdot g_3 = g_1 g_2 g_3$ is a word. A *relation* r is a word which is set to 1. For example $r = g_1 g_2 g_3$ is a relation if $g_1 g_2 g_3 = 1$. Relations of the form $gg^{-1}, g^{-1}g$ are called *trivial* relations.

The structure $\mathcal{F} = \langle g_1, g_2, \ldots : r_1, r_2, \ldots \rangle$ is called a *group presentation* of G if $\{g_1, g_2, \ldots\}$ is a generator set that generates G with respect to the relations $\{r_1, r_2, \ldots\}$. We also write $G = \langle g_1, g_2, \ldots : r_1, r_2, \ldots \rangle$. Two relations are *equivalent* if one can be derived from the other. A group is *freely generated* by $X = \{g_1, g_2, \ldots\}$ if there are no nontrivial relations. In that case the group has a presentation of the form $\langle g_1, g_2, \ldots : -- \rangle$.

3. Polygonal Schema

A *2-manifold* is a connected topological space where each point has a neighborhood homeomorphic to an open disk. By this definition, we consider only 2-manifolds

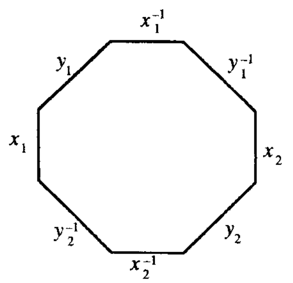

Fig. 3.1. A polygonal schema for double torus.

without boundary. A 2-manifold can be infinite or finite. Moreover, it can be closed or open depending on whether or not its closure coincides with itself. A closed and finite 2-manifold is also called a compact 2-manifold. A sphere and a klein bottle are two examples of compact 2-manifolds. A 2-manifold is called orientable if it has two distinct sides. Otherwise, it is nonorientable. For details see [7].

A 2-manifold is triangulable in the sense that it can be represented as the union of a set of triangles, edges, and vertices satisfying the following properties. Each pair of triangles either share a single vertex or a single edge, or are completely disjoint. Also, the triangles incident on a vertex can be ordered circularly so that two triangles share a common edge if and only if they are adjacent in this ordering. In this paper we consider only compact 2-manifolds without boundary.

Any orientable or nonorientable 2-manifold $\mathcal{M}$ can be represented by a simple polygon P with an even number of edges on $bd(P)$[1] which is also called a *polygonal schema* of $\mathcal{M}$. Each edge of P has a signed label such that each unsigned label occurs twice. See [2] and [7] for details. Two edges with the same unsigned labels are called *partnered edges*. Partnered edges can have labels with the same or opposite signs. Two partnered edges with the labels $+x$ and $-x$ represent the same edge on $\mathcal{M}$ but are oppositely directed on P. Figure 3.1 shows a polygonal schema for double torus ($g = 2$). We use x^{-1} to denote the complement of the label x. To reconstruct a surface homeomorphic to $\mathcal{M}$ from its polygonal representation, the oriented edges with the same labels are identified together in such a way that their orientations match. For simplicity, we say that $\mathcal{M}$ is obtained from P by identifying partnered edges appropriately.

An orientable 2-manifold $\mathcal{M}$ with genus $g > 0$ can be represented *canonically* using a $4g$-gon where all $4g$ vertices represent the same vertex on $\mathcal{M}$. The size of this polygon is *optimal* since no k-gon with $k < 4g$ can represent $\mathcal{M}$. The labels on the edges around the polygon are of the form

$$x_1 y_1 x_1^{-1} y_1^{-1} x_2 y_2 x_2^{-1} y_2^{-1} \cdots x_g y_g x_g^{-1} y_g^{-1}.$$

Similarly, a nonorientable 2-manifold with genus $g > 1$ can be represented *canonically* using an optimal sized $2g$-gon where the labels on the edges around the

[1] $bd(P)$ represents the boundary of the polygon P.

polygon are of the form

$$x_1x_1x_2x_2 \cdots x_gx_g.$$

For $g = 0$, the orientable 2-manifold is a sphere which can be represented canonically by two directed edges xx^{-1}. Similarly for $g = 1$, the nonorientable 2-manifold is the projective plane which can be represented by two directed edges xx. If we identify the partnered edges of the canonical polygon appropriately, they form a set of curves glued at a single point on the 2-manifold. These curves are called *canonical generators*.

Let T be a triangulation of a 2-manifold $\mathcal{M}$ with n edges. Following the notations of [9], we assume that T is represented by its incidence graph $D(T)$. Each face f (vertex, edge, or triangle) of T is represented by a node $D(f)$ in this graph, where there is an arc from $D(f)$ to $D(f')$ if and only if f is incident on f'. The computational model is the pointer machine that manipulates such graphs. A path on T is a sequence of alternating vertices and edges, $v_1e_1v_2e_2 \cdots e_{k-1}v_k$, where the edge e_i has vertices v_i, v_{i+1} as endpoints. A cycle is a path which is closed.

The following lemma provides a method that flattens out T to a triangulation T' of a planar polygon.

Lemma 3.1. *A polygonal schema P with triangulation T' can be constructed from T where there is a one-to-one correspondence between triangles of T' and T.*

Proof. We construct a sequence of closed disks $D_1, D_2, \ldots, D_n$ incrementally such that $P = D_n$ at the end. Initially, $D_1 = \sigma_1'$, a triangle that corresponds to an arbitrarily chosen triangle σ_1 on T. Let $D_i = \sigma_1' \cup \sigma_2' \cup \cdots \cup \sigma_i'$ after the ith step. At the $(i + 1)$th step we choose a triangle σ_{i+1} on T which has the following properties:

(i) No triangle corresponding to σ_{i+1} has been included in D_i.
(ii) A triangle σ_j adjacent to σ_{i+1} by an edge has a corresponding triangle in D_i.

These two conditions imply that there is an edge $e = \sigma_j \cap \sigma_{i+1}$ such that its corresponding edge e' on D_i appears on $bd(D_i)$. We attach a triangle σ_{i+1}' to $bd(D_i)$ such that $\sigma_{i+1}' \cap bd(D_i) = e'$. This gives the new disk $D_{i+1} = D_i \cup \sigma_{i+1}'$. It is clear that if D_i is a closed disk, so is D_{i+1}. Finally, when we exhaust all triangles on $\mathcal{M}$, we have D_n with the triangulation T' that has the following properties:

(i) Each triangle in T' corresponds to a single triangle in T and vice versa.
(ii) Each edge e' on $bd(D_n)$ has a partnered edge e'' such that they both correspond to a single edge e on T. This is because each edge e on T has two incident triangles and the edges on $bd(D_n)$ have a single triangle incident on them. Further, when partnered edges of $bd(D_n)$ are identified, we get back T. This is because, by our construction, the two triangles in T' incident on partnered edges e', e'' correspond to the two triangles incident on e in T.

(iii) Incidence relations of all other edges inside T' are isomorphic to their corresponding edges on T. So, $D_n = P$, a polygonal schema for $\mathcal{M}$ with a one-to-one correspondence between triangles in T and T'. □

4. Computing an Optimal Polygonal Schema

The size (number of edges on the boundary) of the polygonal schema as deduced in the last section can be $\Omega(n)$, which is too large for our purpose. Here, we show a technique to derive a polygonal schema Q of optimal size from P that retains enough information to preserve a surjective mapping from vertices of P to Q. Due to this mapping, we would be able to trace a path on Q that corresponds to a path on P and hence a path on T. Here we must mention that our method does not produce the reduced polygonal schema in canonical form as was done by Vegter and Yap in [9]. We already observed that the triangulation T' to P becomes isomorphic to the triangulation T of $\mathcal{M}$ when partnered edges are identified. From now on we do not distinguish between the triangles, edges of T and T'; we refer to them with the same name. With this set-up, two partnered edges on $bd(P)$ have the same name.

Let the 1-complex (graph) formed by the identified edges of $bd(P)$ be G. The space $\mathcal{M} - G$ is an open disk since $P - bd(P)$ is an open disk. Let us attach a thin layer of an open set on two sides of the edges of G to make it open, and denote this space G'. Let the attached open set be thin enough so that the space $D = \mathcal{M} - G'$ is an open disk; see Fig. 4.1. We can express $\pi_1(\mathcal{M})$ in terms of $\pi_1(G')$ and $\pi_1(D)$ using the following theorem:

Seifert–Van Kampen's Theorem. *Suppose S is a space which can be expressed as the union of path-connected open sets A, B such that $A \cap B$ is path connected and such that $\pi_1(A)$ and $\pi_1(B)$ have respective representations*

$$\pi_1(A) = \langle a_1, a_2, \ldots : r_1, \ldots, r_n \rangle,$$

$$\pi_1(B) = \langle b_1, b_2, \ldots : s_1, \ldots, s_q \rangle,$$

while $\pi_1(A \cap B)$ is finitely generated. Then $\pi_1(S) = \langle a_1, a_2, \ldots, b_1, b_2, \ldots : r_1, \ldots, r_n, s_1, \ldots, s_q, u_1 v_1^{-1}, \ldots, u_m v_m^{-1} \rangle$ where u_i, v_i are the expressions for the same generator of $\pi_1(A \cap B)$ in terms of the generators of $\pi_1(A)$ and $\pi_1(B)$, respectively.

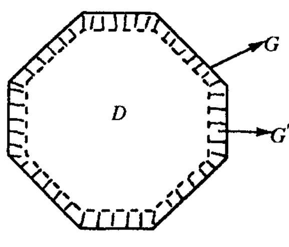

Fig. 4.1. The spaces D, G, and G'.

Since any cycle (curve) on G' can be continuously deformed to the edges of G, we have $\pi_1(G) = \pi_1(G')$. Let a set of generators of $\pi_1(G)$ be γ with the set of relations ρ_1. The fundamental group of D is the trivial group $\{1\}$ since D is an open disk. Let the set of relations obtained for the generators of $\pi_1(D \cap G')$ in terms of the generators of $\pi_1(D)$ and $\pi_1(G')$ be ρ_2. The spaces $\mathcal{M}$, G', and D satisfy all conditions of Seifert–Van Kampen's theorem. Then according to this theorem $\pi_1(\mathcal{M})$ has a presentation $\langle \gamma: \rho_1, \rho_2 \rangle$.

A presentation of $\pi_1(G)$ can be obtained as follows. Let Y be a spanning tree of G, and let p be a vertex of Y. Each edge e of G defines a cycle $c(e) = wew'$ originating at p where w, w' are the paths from p to the endpoints of e along the edges of Y. To simplify notation we write e for the cycle $c(e)$. Any cycle e where e is an edge of Y is contractible to p and hence belongs to the identity of $\pi_1(G)$. Let $B = \{b_1, b_2, \ldots, b_l\}$ be the set of edges of G that do not belong to Y. A presentation of $\pi_1(G)$ is $\langle b_1, b_2, \ldots, b_l: --\rangle$ implying $\rho_1 = \{\ \}$. Consider the sequence of edges around (clockwise) $bd(P)$. Some of these edges belong to Y which are set to 1 in $\pi_1(G)$. In terms of the generators of $\pi_1(G)$, let the sequence of edges around $bd(P)$ form the word $b'_1 11 \cdots b'_2 11 \cdots b'_3 11 \cdots b'_{2l} 11 \cdots 1$, where the b'_i's represent the edges in B or their inverses. This word is equivalent to $r = b'_1 b'_2 \cdots b'_{2l}$. The generator for the circle $D \cap G'$ is $b'_1 b'_2 \cdots b'_{2l}$ in terms of the generators of $\pi_1(G')$ and is 1 in terms of the generator of D. So we have $\rho_2 = b'_1 b'_2 \cdots b'_{2l} = 1$. Hence a presentation of $\pi_1(\mathcal{M})$ is $\langle b_1, b_2, \ldots, b_l: \rho_2 \rangle$.

Lemma 4.1. *$l = 2g$ if $\mathcal{M}$ is orientable and $l = g$ if $\mathcal{M}$ is nonorientable.*

Proof. Consider the polygon P with all edges in Y contracted to a single point. This results in a polygon Q with edges $b'_1 b'_2 \cdots b'_{2l}$ labeled clockwise around it (Fig. 4.2). The polygon Q is a $2l$-gon with partnered edges. Further, we can assume that all vertices of Q have same label, i.e., when partnered edges are identified, all

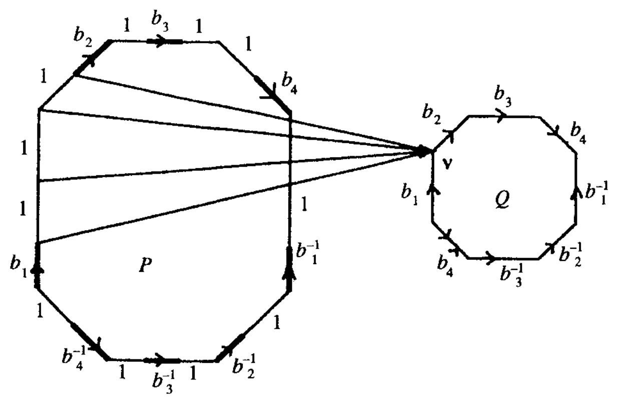

Fig. 4.2. Reduced polygonal schema.

vertices are identified to a single point. Then Q represents a polygonal schema for a manifold $\mathcal{M}'$ homeomorphic to $\mathcal{M}$. This is because:

(i) When partnered edges are identified Q forms a 2-manifold $\mathcal{M}'$ whose fundamental group is the same as $\pi_1(\mathcal{M})$ (apply Seifert–Van Kampen's theorem).
(ii) Any two compact 2-manifolds are homeomorphic if and only if their fundamental groups are isomorphic.

Euler's characteristic of $\mathcal{M}'$ is $2 - l$ since we get one vertex, l edges, and one face when the edges of Q are identified. It is known that Euler's characteristic of a surface $\mathcal{M}$ of genus g is $2 - 2g$ if $\mathcal{M}$ is orientable and is $2 - g$ if $\mathcal{M}$ is nonorientable. Since $\mathcal{M}$ and $\mathcal{M}'$ are homeomorphic, their Euler's characteristics must be equal, proving $l = 2g$ if $\mathcal{M}$ is orientable and $l = g$ if $\mathcal{M}$ is nonorientable. □

4.1. *The Algorithm*

We construct the polygonal schema P as described in Lemma 3.1. This takes only $O(n)$ time to traverse the incidence graph $D(T)$ and simultaneously build up the graph $D(T')$ and hence P.

While constructing P we maintain pointers from the edges and triangles of T to the edges and triangles of T'. We detect the edges of the 1-complex G through the edges of $bd(P)$ in $O(n)$ time. A spanning tree Y of G is computed in $O(n)$ time by a simple depth-first search in G. The edges not in Y are detected and the corresponding edges on $bd(P)$ are marked with the label other than 1. All other edges on $bd(P)$ are marked 1. All these take $O(n)$ time.

The reduced polygon Q of size $2l$ is constructed from P as follows. Let the sequence of edges around $bd(P)$ (clockwise) be $b'_1 b'_2 \cdots b'_{2l}$ ignoring the edges marked 1. We form the polygon Q to have $2l$ edges labeled $b'_1 b'_2 \cdots b'_{2l}$ around it. We maintain pointers from the vertices of P to the vertices of Q as follows. Let $b'_i b'_{i+1}$ be any two consecutive edges in the sequence (circular) $b'_1 b'_2 \cdots b'_{2l}$ and let $v_1 v_2 \cdots v_s$ be the vertices of $bd(P)$ between b'_i and b'_{i+1} where v_1 is an endpoint of b'_i and v_s is an endpoint of b'_{i+1}. All these vertices point to the same vertex v between b'_i and b'_{i+1} in Q. See Fig. 4.2. The polygon Q can be thought of as the polygon P with all edges in the spanning tree Y shrunk to a single vertex. By Lemma 4.1 Q has optimal size. The pointers from P to Q realize a surjective mapping between their vertices. Combining all costs together Q can be constructed from T in $O(n)$ time.

Theorem 4.2. *Let $\mathcal{M}$ be any compact 2-manifold of genus g with triangulation T of size n. A polygonal schema Q of optimal size can be constructed in $O(n)$ time, where a surjective mapping is retained from the vertices of a polygonal schema embedded in T to the vertices of Q.*

5. Detecting Null-Homotopy

Traversing a cycle C on T is equivalent to traversing an ordered set of paths $U = \{u_1, u_2, \dots, u_r\}$ that have endpoints on $bd(P)$ if C intersects G. Each of these paths u_i is homotopic to a path u'_i that has edges only on $bd(P)$ and have the same endpoints as u_i. Note that no such paths exist when C does not intersect G. In that case C is trivially null-homotopic. We consider the nontrivial case when C intersects the edges of G.

When edges on $bd(P)$ are identified, the sequence $u'_1, u'_2, \dots, u'_r$ form a cycle C' on T, which is homotopic to C. Each path u'_i can be expressed as a word w_i in terms of the generators $b_1, b_2, \dots, b_l$ by listing the sequence of edges on it that are not marked 1 on $bd(P)$. This is actually done on $bd(Q)$ to avoid unnecessary visits of edges that are marked 1 on $bd(P)$. We also write $w_i \sim u_i$ if the word w_i corresponds to the path u_i. The word $w = w_1, w_2, \dots, w_r$ represents the cycle C in $\pi_1(\mathcal{M})$. We determine if $w = 1$ to detect if C is null-homotopic (Fact 2.1).

The sequence of paths $\{u_1 u_2 \cdots u_r\}$ with endpoints on $bd(P)$ are detected through pointers as we traverse C on T. The homotopic path u'_i of u_i is identified by the endpoints v_1, v_2 of u_i. We do not traverse the path u'_i on $bd(P)$ which can have $\Omega(n)$ length. To form the word $w_i \sim u_i$ we detect the corresponding vertices of v_1, v_2 on $bd(Q)$ through pointers from P to Q. We can consider the labeled edges between these vertices around $bd(Q)$ in any direction (clockwise or counterclockwise) to form the word w_i. The length of w can be $\Omega(gk)$ since each word w_i can have $\Omega(l) = \Omega(g)$ length and there can be $r = \Omega(k)$ such words in the worst case, where k is the length of the given cycle. If we construct the word w explicitly, we may have to spend $\Omega(gk)$ time which is worse than our goal of $O(k \log g)$ time. For this we only concentrate on the *first* and *last* edges of each w_i, and process them in a partially built universal covering space as described below.

5.1. *The Universal Covering Space*

Informally, a topological space $\mathcal{U}(X)$ is a covering space of another topological space X if each point in $\mathcal{U}(X)$ has a neighborhood similar to a point in X. More precisely, a continuous surjective map $f\colon \mathcal{U}(X) \to X$ exists such that, for an ε-neighborhood N of any point in X, the inverse image $f^{-1}(N)$ is a union $\bigcup_i N_i$ of ε-neighborhoods in $\mathcal{U}(X)$ where each N_i is homeomorphic to N, which is realized by the restriction $f|_{N_i}$. If $\mathcal{U}(X)$ is simply connected, it is called the universal covering space of X.

We can construct a universal covering space $\mathcal{U}(\mathcal{M})$ of $\mathcal{M}$ as follows: Let $P_{\mathcal{M}}$ be a $4g$-gon (if $\mathcal{M}$ is orientable) or a $2g$-gon (if $\mathcal{M}$ is nonorientable) such that when its edges are identified we get a set of generators for $\mathcal{M}$ meeting at a single point. By taking infinitely many copies of $P_{\mathcal{M}}$ and gluing them together along the identified edges, a tessellation of the plane with either $4g$-gons or $2g$-gons, depending on whether $\mathcal{M}$ is orientable or not, is obtained. The polygon Q is a polygonal schema for $\mathcal{M}$ where all $2l$ vertices represent the same point on $\mathcal{M}$. Hence we can take $P_{\mathcal{M}} = Q$ to construct $\mathcal{U}(\mathcal{M})$. A general strategy to construct $\mathcal{U}(\mathcal{M})$ is by starting with

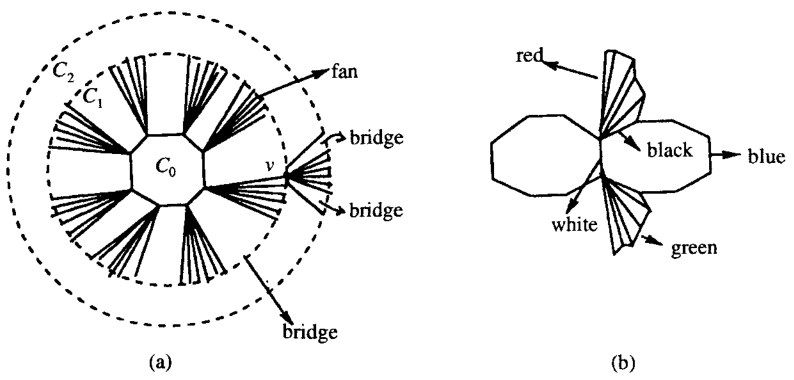

Fig. 5.1. Universal covering space of the double torus (a) and a typical structure (b).

a single copy of $P_{\mathcal{M}}$ (base polygon) whose vertices lie on a circle C_0. Now there is a unique topological way to complete the neighborhood of each vertex of the base polygon with copies of $P_{\mathcal{M}}$ since copies are glued along identified edges. The newly created vertices lie on a circle C_1. This process can be repeated *ad infinitum*. A part of the universal covering space of the double torus is illustrated in Fig. 5.1(a). In this figure the triangles that fan out from a vertex on C_i represent a set of squeezed copies of $P_{\mathcal{M}}$ with all other vertices on C_{i+1}. The following fact is well known. See [7] for details.

Fact 5.1. Any curve C on $\mathcal{M}$ can be mapped to a unique path u (lifted path) in $\mathcal{U}(\mathcal{M})$ such that C is null-homotopic if and only if u is closed.

Let the first vertex v_0 on w be mapped to a point p_0 on $\mathcal{U}(\mathcal{M})$. The lifted path u of w starting from p_0 can be constructed incrementally as follows. Let the path u be traced to the edge e_i', which corresponds to the edge e_i of w. The next edge e_{i+1} is mapped to a unique edge e_{e+1}' on $\mathcal{U}(\mathcal{M})$. To augment the lifted path to the edge e_{i+1}, we traverse the unique edge e_{i+1}' connected to e_i' on $\mathcal{U}(\mathcal{M})$. To check if the lifted path u of w is closed, it is sufficient to process the edges through which we enter or leave a polygon in $\mathcal{U}(\mathcal{M})$. We do not need to process explicitly other edges as they are traversed on the same polygon. In terms of a single component w_i, it means that it is sufficient to process the first and last edges of w_i since all edges of w_i are traversed on the same polygon. Let w' denote the sequence of first and last edges of w_i, $i = 1, \ldots, r$. Certainly, $|w'| \leq 2r = O(k)$. The edges of w' are processed in a partially built structure of $\mathcal{U}(\mathcal{M})$ that is augmented as and when necessary.

We use the notation as introduced in [5]. A vertex v has distance k from the base polygon if it lies on the circle C_k. A polygon has distance k if the maximum of the distances of its vertices is k. Edges between vertices of different distances are called *spokes*, other edges are called *arcs*. Polygons at distance k incident with two vertices at distance $k - 1$ are called *bridges*. The *fan* of a vertex v at distance k are the polygons incident on v at distance $k + 1$ which are not bridges.

5.2. The Structure

The algorithm maintains a part of the universal covering space, which we call the *structure*. In this structure the polygons represent copies of Q. However, not all polygons in the structure are real copies. Most polygons will be *under construction*; that is, chains of edges separated by vertices may be represented by a single edge (unfinished edge). The fans will also be under construction. Initially a fan at a vertex is realized by a single triangle which is expanded as and when necessary. If a polygon or a fan is not under construction, we call it *finished.* Edges of the structure have one of the following colors (Fig. 5.1(b)):

- White, if it is an edge between two finished polygons.
- Black, if it is an edge between two polygons in the structure, at least one of which is not finished.
- Red, if it is both a spoke and a boundary edge of the structure.
- Blue, if it is an arc and boundary edge of the structure.
- Green, if it is an arc, bounding the structure, and a so-called unfinished edge. It connects two vertices which are not connected by an edge of Q.

5.2.1. Enlarging the Structure. We enlarge the structure by one of the following operations.

- Add a polygon P to the structure. Initially P is a triangle or a rectangle. If P is a member of a fan, it will be a triangle. If P is a bridge, it will be a rectangle with two spokes, one arc at a smaller distance and an unfinished edge (green) at a larger distance.
- Continue constructing a polygon. This means we add at most two blue edges to a polygon. Adding blue edges means that a green edge e of a polygon is subdivided in a blue–blue sequence or a blue–green sequence depending on whether e clumps two or more edges in it.
- Add a fan to a vertex. If we do that, the fan will be under construction, that is, we add a triangle consisting of two extreme spokes of the fan and an unfinished green edge between them.

5.2.2. Invariants. During the algorithm we maintain the following invariants:

- The structure represents a part of the universal covering space and is homeomorphic to a disk.
- All the first and last edges of the lifted curve that have been processed are contained in the structure.

5.3. Algorithm

5.3.1. Data Structures. The polygons and the fans are maintained with data structures that allow fast search and insert operations. Each polygon is given a clockwise orientation which orients its edges. Thus each edge has two directed edges associated with it corresponding to the two polygons incident on it. Each directed edge, in turn, has the edge associated with it. A polygon is maintained as a linked list of directed edges which enlarges as the polygon expands. On top of this linked list, the data structure for fast search and insert operation is maintained. A similar fast data structure is also maintained for the spokes of each fan.

We use a balanced binary search tree that allows search and insert operation in $O(\log g)$ time. For this, the vertices around $bd(Q)$ are indexed by integers $1, 2, \dots, 4g$ ($2g$ if $\mathscr{M}$ is nonorientable), which in turn render the edges of w' as tuples of integers. Each copy of polygon Q maintains a balanced binary search tree $D(Q)$ on top of its linked list, where the indices of the vertices are used as search keys. Each vertex v maintains a similar binary search tree $F(v)$ for its fan, and a list $L(v)$ of at most five edges of bridges that are incident on v. If v is at a distance k, these five edges include four edges of two bridges at distance k and a spoke of a bridge at distance $k - 1$. See the vertex v in Fig. 5.1(a).

5.3.2. Processing the Edges. We process w in terms of the first and last edges of the w_i's. For each w_i, as well as its first and last edges we also need the edge after the first edge. Let e be any generic edge which we are visiting. We call the polygon *current* on which e is visited. Our strategy of moving through the structure is as follows.

For a first edge e of w_i we determine the current polygon Q_c with the help of e', where e' is the next edge after e on w_i. Then we search for the last edge of w_i in $D(Q_c)$. If it is not present, we insert it in $D(Q_c)$. If v is the vertex through which we exit Q_c, we search for the first edge of w_{i+1} in $F(v)$ and $L(v)$ after initializing them if necessary. Depending on the color and type of e we take appropriate actions as follows.

1. e is a first edge of w_i: e can only be black or white. This is guaranteed by the fact that when we exit a polygon through a vertex v, all necessary fans and bridges are attached to v. Of course, this is not true when e is the first edge of w_1. This initial condition can be handled as a special case without any difficulty. Let e' be the next edge after e on w_i. If $|w_i| = 1$, we take $e' = e$ and visit e as a last edge.
   - e is white. Out of the two polygons incident with e, only one has e' after e. Detect that polygon and make it current. This is possible since each edge is uniquely determined by its endpoints.
   - e is black. If necessary, insert the two polygons (under construction) incident on e. If e is a spoke, this involves inserting at most two edges in the fan of which e is a member. If e is an arc, the two polygons incident on e are already present. In both cases let Q_1, Q_2 be two polygons incident on e. Let $v = e \cap e'$. If any of the edges of Q_1, Q_2 incident on v is green, break it to have a blue edge incident on v. Detect the polygon $Q_c \in \{Q_1, Q_2\}$ on which e' will be traversed. Make Q_c current. See Fig. 5.2 for an illustration.

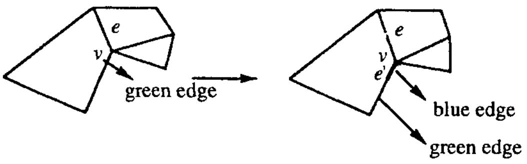

Fig. 5.2. Processing a black arc.

2. e is a last edge: Let the next edge e', a first edge on w_{i+1}, be incident on the vertex v of e.
   - e is red. Let v' be the vertex of e at the smaller distance. If v' is incident on a blue edge, add the missing fan and bridge (under construction) incident with v'. The edge e is now incident on two polygons. If v' is incident on a green edge, then replace the green edge by a blue–green sequence so that the blue edge is incident on v'. Treat this case by assuming v' to be incident on a blue edge. See Fig. 5.3 for an illustration. If v' is incident on another red edge e'', call the procedure recursively on e''. See Fig. 5.4. This will cause v' to be incident on a blue or green edge. Now apply the procedure described above.

     After this processing, proceed as follows. There are two cases: (i) v is the vertex at the smaller distance. The fan and the bridges incident on v are in the structure now. Select e' after the appropriate search. See Fig. 5.4(c). (ii) v is the vertex of e at the larger distance. Break the green edges (at most two) incident on v and then attach the fan and bridges to v (all under construction). Now select e'.
   - e is white. We have two cases. (i) e is a spoke. Both polygons incident on e are finished. Attach the fan at v and two bridges if they are not there (all under construction). If e' belongs to any of the two arcs incident on v, then visit e'. Otherwise, e' belongs to one of the edges of the fan at v. Search e' in $F(v)$. If it is not in $F(v)$, insert it with the necessary polygon(s) (under construction) incident on e'. Visit e'. (ii) e is an arc. If v is incident on any red edge, there must be one red edge with v at the smaller distance. Process the red edge as if it is visited as a last edge. This will put necessary fans and bridges incident on v. If v is not incident on any red edge, the necessary fans and bridges are already present. Select the edge e' from the fan and bridges incident on v as described above. See Fig. 5.5.

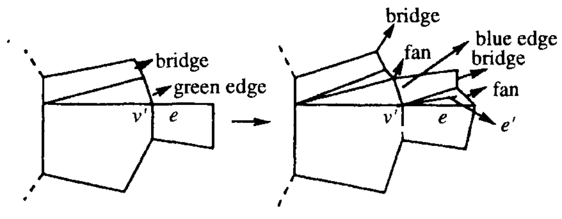

Fig. 5.3. Processing a red edge where v' is incident on a green edge.

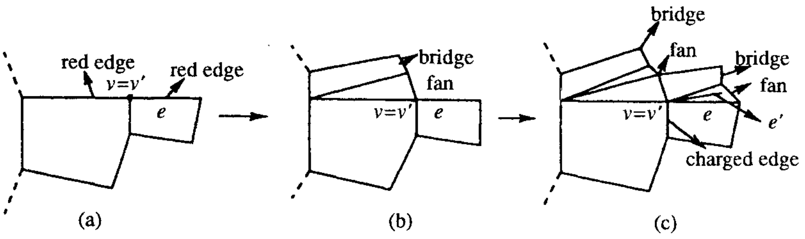

Fig. 5.4. Processing a red edge recursively.

- *e* is black. If necessary add new spokes with appropriate polygons to the fan under construction. Attach the necessary fan and bridges to v and select the edge e' as in the previous case.
- *e* is blue. (i) v is not incident on any spoke. Attach the fan and bridges to v and then select e'. (ii) v is incident on a spoke which is red. Process the red edge as if it is visited as a last edge. This attaches the fan and bridges to v. Select e' from them. (iii) v is incident on a spoke, which is not red. Attach the fan and bridges to v and select e'.

Time Complexity. Except for red edges, processing all other edges involves enlarging the structure by $O(1)$ size. This implies that processing each nonred edge increases space by a constant amount. Processing edges involves search and insert operations in the data structure for fans and polygons. Hence each nonred edge takes $O(\log g)$ time.

A red edge can incur an $\Omega(k)$ cost since it may involve $\Omega(k)$ other red edges in a recursive call. However, we can charge this cost to some other edges processed in the past. Each red edge r involved in a recursive call takes $O(1)$ time. This includes the time to attach an $O(1)$ size fan and bridge to the vertex v between r and the next red edge. Also only $O(1)$ time is needed to check if any of the edges in $L(v)$ other than r is a red edge. Each red edge r in a recursive call is incident on a bridge because every time we attach a fan, we also attach the adjacent bridges. The $O(1)$ time spent for it can be charged to the black or white edge incident on v between the two bridges. See Fig. 5.4(c). Observe that a red edge can participate in a recursive call only once since it is turned white or black by the recursive call. Also a

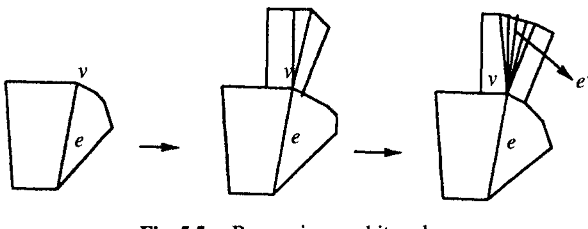

Fig. 5.5. Processing a white edge.

crucial observation is that the black or white edges created by the recursive calls are never charged in the future. This implies that any black or white edge that is charged must be created by an edge that is not involved in a recursive call. All these imply that we spend $O(1)$ amortized time per red edge for recursive calls. Including the possible search and insert time for the first red edge we conclude that each red edge processing takes $O(\log g)$ amortized time. The same argument shows that each red edge adds $O(1)$ amortized space.

So, processing each first and last edge takes $O(\log g)$ amortized time and $O(1)$ amortized space. Since there are k such edges, we have $O(k \log g)$ total time and $O(k)$ total space. However, when $g \leq 1$ ($g \leq 2$ if $\mathscr{M}$ is nonorientable), the above charging scheme fails since there is no white or black edge that is guaranteed to be created by a nonrecursive step, which can be charged for red edges involved in recursive calls. Indeed, for $g = 1$ ($g = 2$ if $\mathscr{M}$ is nonorientable), the above procedure takes $\Omega(k^2)$ time in total. We consider these cases separately.

Combining the costs of all steps we obtain the following theorem.

Theorem 5.1. *Let $\mathscr{M}$ be any compact orientable (nonorientable) 2-manifold of genus $g > 1$ ($g > 2$) with a triangulation T of size n. We can detect if a given cycle C of length k is null-homotopic in $O(n + k \log g)$ time and in $\Theta(n + k)$ space.*

5.3.3. Special Cases. Any cycle on a sphere ($g = 0$) is contractible to a single point. So, the contractability problem is trivial for spheres. The fundamental group of any torus ($g = 1$) has the presentation $\langle a, b\colon aba^{-1}b^{-1}\rangle$ which means $ab = ba$. Thus any word w on the canonical generators can be expressed as $w = a^m b^n$. This implies $w = 1$ if and only if $m = 0$ and $n = 0$. Let an edge e on Q (a rectangle) be traversed c_1 times in the clockwise direction and c_2 times in the counterclockwise direction on the paths/words $w_1, w_2, \ldots, w_r$. Let $c^e = c_1 - c_2$. We have $m = 0$ and $n = 0$ if and only if $c^e + c^{e'} = 0$ for each pair of partnered edges e, e'. Detecting this fact takes $O(\Sigma_{i=1}^r |w_i|) = O(gk) = O(k)$ time. With the $O(n)$ preprocessing time we have a $\Theta(n + k)$ time optimal algorithm for the torus.

Similar to the orientable 2-manifolds, the cases when $g = 1$ (projective planes) and $g = 2$ (klein bottle) for nonorientable 2-manifolds are treated separately. Although the case for $g = 1$ can be solved trivially, the case for $g = 2$ (polygon Q is a rectangle) cannot be solved by the method used for the torus. Instead we use a different method for klein bottles.

For klein bottles, the polygon Q is a rectangle and the sequence of edges around $bd(Q)$ reads either $abab^{-1}$ or $aabb$; see Fig. 5.6. The universal covering space with these two patterns are shown in Fig. 5.6. We observe that vertices in the universal covering space have two different configurations in both cases. We can determine in an $O(1)$ step the edges that take us left, right, up, and down for each configuration. For example, from configuration F_1 (Fig. 5.6) we go left, right, up, and down by the edges b^{-1}, b, a^{-1}, and a, respectively. We can also decide in $O(1)$ time which configuration we end up with when we move from a particular configuration through a particular edge. For example, we go to F_2' from F_1' through the edge b^{-1} (Fig. 5.6).

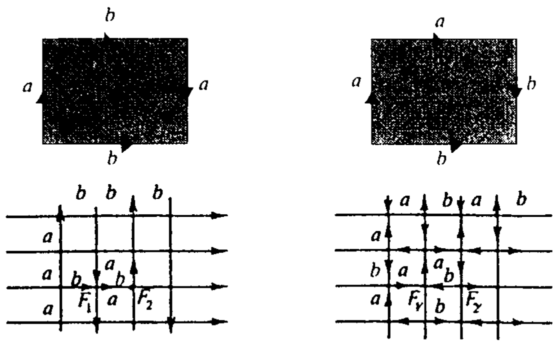

Fig. 5.6. Universal covering spaces for klein bottles.

Once we have this information available through a preprocessing step, we can visit the edges of w on $bd(Q)$ and for each such edge we know the direction of our movement in the universal covering space without constructing it explicitly. The path on the universal covering space will be closed if and only if we move an equal number of times left and right and an equal number of times up and down. This procedure takes $O(|w|) = O(gk) = O(k)$ time. With preprocessing times added, we have a $\Theta(n + k)$ time and space algorithm.

5.3.4. Related Problems.

Detecting Homotopic Cycles. Two cycles C_1, C_2 of lengths k_1, k_2 respectively are homotopic if and only if $[C_1] = [C_2]$ giving $[C_1] \cdot [C_2]^{-1} = [C_1 \cdot C_2^{-1}] = 1$. Let w_1, w_2 be the words constructed corresponding to C_1 and C_2^{-1}, respectively. We have $[C_1 \cdot C_2^{-1}] = 1$ if and only if $w_1 \cdot w_2^{-1} = 1$. This can be detected in $O(g(k_1 + k_2))$ time giving an $O(n + g(k_1 + k_2))$ time algorithm for this problem.

Computing Actual Contractions. The path u corresponding to w on $\mathscr{U}(\mathscr{M})$ is closed if and only if the original cycle C is null-homotopic. In case u is closed we can compute the actual contraction as follows. First, we finish all unfinished polygons in the partial structure. This takes $O(gk)$ time since there are at most $O(k)$ copies of Q (finished or unfinished) which are expanded by $O(g)$ edges. The path u is recovered by concatenating $w_1, w_2, \ldots, w_r$, where the first and last edges of each w_i are available in the structure. This also takes $O(gk)$ time. Next, each polygon Q is expanded to the polygon P with a subdivision of its edges. The triangulation T' of P is superimposed on these expanded polygons. The edges of the closed curve u also get subdivided by this process. This refined cycle represents the cycle $C' = u'_1 u'_2 \cdots u'_r$ on $bd(P)$. A contraction of the refined cycle C' over the expanded polygons gives a contraction of C'. Also the deformations of the paths $u_1, u_2, \ldots, u_r$ to $u'_1, u'_2, \ldots, u'_r$ constitute the deformation of C to C'. Since each edge of C can be deformed over

$O(n)$ triangles, deformation of C to C' takes $O(nk)$ time and space. Since there are $O(k)$ polygons, C' can be contracted over $O(k)$ triangulations of size $O(n)$ each. This gives an $O(nk + gk) = O(nk)$ time and space actual contraction for C. This is worst-case optimal since there is an example, where actual contraction necessarily takes $\Omega(nk)$ time and space [5].

Theorem 5.2. *The actual contraction of a given cycle on a triangulated compact 2-manifold can be computed in* $\Theta(nk)$ *time and space.*

6. Conclusions

We have presented a new technique to derive a reduced polygonal schema from a triangulation of a given 2-manifold. This has produced an improved algorithm for detecting null-homotopic cycles on compact 2-manifolds. We believe that this technique will find further applications in other algorithms on 2-manifolds. The method can be adapted to solve the contractibility problem on compact 2-manifolds with boundary. An open question is: can the time bound $O(n + k \log g)$ achieved for the null-homotopy problem be improved to $\Theta(n + k)$?

An extension of the null-homotopy problem is to consider it on arbitrary complexes. The fundamental group of any ($d > 1$)-dimensional complex is determined only by its two-dimensional subcomplex. So, to solve the contractibility problem we need to consider only the 2-skeleton of any complex. However, it is known that the contractibility problem is unsolvable for arbitrary 2-complexes [7]. For any arbitrary 2-complex a 4-manifold that contains it as a subcomplex exists. Hence the contractibility problem for four- and higher-dimensional manifolds is unsolvable. The case for 3-manifolds remains to be answered.

Acknowledgments

We thank the referee whose suggestions improved the expositions of the paper. The first author wishes to thank Herbert Edelsbrunner for sharing an idea on the special case of klein bottles.

References

1. H. Edelsbrunner and C. Delfinado. An incremental algorithm for Betti numbers of simplicial complexes. *Proc. 9th. Ann. Symp. Computational Geometry*, 1993, pp. 232–239.
2. P. J. Giblin. *Graphs, Surfaces and Homology*, Halsted Press, New York, 1977.
3. J. Hersberger and J. Snoeyink. Around and around: computing the shortest loop. *Proc. 3rd Canad. Conf. on Computational Geometry*, 1991, pp. 157–161.
4. K. Melhorn and C. K. Yap. Constructive Whitney-Graustein theorem: or how to untangle closed planar curves. *SIAM J. Comput.* **20**(4) (1991), 603–621.

5. H. Schipper, Determining contractibility of curves. *Proc. 8th ACM Symp. on Computational Geometry*, 1992, pp. 358–367.
6. J. S. Snoeyink. Topological Approaches in Computational Geometry. Tech. Report STAN-CS-90-1331, 1990.
7. J. Stillwell. *Classical Topology and Combinatorial Group Theory*. Springer-Verlag, New York, 1980.
8. G. Vegter. Kink-free deformations of polygons, *Proc. 5th ACM Symp. on Computational Geometry*, 1989, pp. 61–68.
9. G. Vegter and C. K. Yap. Computational complexity of combinatorial surfaces. *Proc. 6th ACM Symp. on Computational Geometry*, 1990, pp. 102–111.

Received April 15, 1993, and in revised form January 20, 1994, and September 19, 1994.