An Incremental Algorithm for Betti Numbers of Simplicial Complexes*

Cecil Jose A. Delfinado and Herbert Edelsbrunner
Department of Computer Science, University of Illinois, Urbana, Illinois 61801, USA.

Abstract

A general and direct method for computing the betti numbers of the homology groups of a finite simplicial complex is given. For subcomplexes of a triangulation of $\mathbb{S}^3$ this method has implementations that run in time $O(n\alpha(n))$ and $O(n)$, where n is the number of simplices in the triangulation. If applied to the family of α-shapes of a finite point set in $\mathbb{R}^3$ it takes time $O(n\alpha(n))$ to compute the betti numbers of all α-shapes.

1 Introduction

Computing the homology groups of a topological space is one of the main interests in the field of algebraic topology, see e.g. [1, 9, 11, 12]. Although current applications of homology groups are not numerous, homology groups of complexes imbeddable in three-dimensional Euclidean space, $\mathbb{R}^3$, attract attention because of their intuitive appeal. Given an object in $\mathbb{R}^3$, it is natural to ask how many connected components, how many tunnels, and how many holes there are. The number of components, tunnels, and holes have concrete interpretations. Examples are the number of heavenly bodies in a galaxy, the number of independent closed routes that go around obstacles, and the number of portions of a cell occupied by fluid. The nice thing about these objects is that they are physically realizable. That is, it is possible to imbed them in $\mathbb{R}^3$. It is here where our algorithm will probably find most of its applications.

In homology theory, if the simplicial complex is small, then the homology group computations can be done by hand. To solve these problems in general, a classic algorithm exists and is discussed in length in [11]. It forms matrices and reduces them to a canonical form, known as the Smith normal form [13], from which one can read off the homology groups of the complex. The reduction to Smith normal form is the bottleneck of this algorithm. Starting with [10], several methods have been proposed to speed up this part of the computation. The only upper bound known on the worst-case running time of the classic reduction algorithm is double-exponential in the size of the input. However, [4] have observed that for simplicial complexes that arise in geometric design the matrices are sparse, and they argue that in a probabilistic sense the algorithm then runs in time at most quadratic in the size of the complex.

We describe a more direct method for computing the betti numbers of the homology groups of simplicial complexes in finite dimensions. Its correctness is established using Mayer-Vietoris sequences. In particular, this method is applied to complexes imbeddable in $\mathbb{R}^3$. Assuming the complex is given as a subcomplex of a triangulation of $\mathbb{S}^3$, the algorithm runs in time $O(n\alpha(n))$, where n is the number of simplices in the triangulation and $\alpha(n)$ is the extremely slowly growing inverse of the Ackermann function. If the complex is represented so that the simplices incident to a given simplex can be accessed in constant time each then this can be improved to time $O(n)$. Section 2 reviews the necessary concepts from algebraic topology, sections 3, 4, and 5 develop the algorithm, complete with proofs of correctness and analysis, and section 6 applies the algorithm to three-dimensional α-shapes.

*This work is supported by the National Science Foundation under grant ASC-9200301 and the Alan T. Waterman award, grant CCR-9118874. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and do not necessarily reflect the view of the National Science Foundation.

2 Algebraic Topology Concepts

This paper requires some prerequisites from homology theory, at least if one wants to be sure about the correctness of the general approach. The algorithms themselves are reasonably intuitive so that a good understanding can be reached without the prerequisites reviewed in this section. Our terminology follows the one in [11].

Simplices, complexes, and triangulations. For $0 \leq k \leq d$, a *k-simplex* σ in $\mathbb{R}^d$ is the convex hull of a set T of $k+1$ affinely independent points. The *dimension* of σ is $\dim \sigma = |T| - 1 = k$. For every $U \subseteq T$, the simplex σ' defined by U is a *face* of σ, and if $U \neq T$ then σ' is called a *proper face* of σ. We say that σ' and σ are *incident* if σ' is a face of σ. In three dimensions we use the terms *vertex* for 0-simplex, *edge* for 1-simplex, *triangle* for 2-simplex, and *tetrahedron* for 3-simplex.

A collection of simplices, $\mathcal{K}$, is a *(simplicial) complex* if it satisfies two properties, namely (i) if σ' is a face of σ and $\sigma \in \mathcal{K}$ then $\sigma' \in \mathcal{K}$, and (ii) if $\sigma_1, \sigma_2 \in \mathcal{K}$ then $\sigma_1 \cap \sigma_2$ is either empty or a face of both. The largest dimension of any simplex in $\mathcal{K}$ is the *dimension* of $\mathcal{K}$. All simplices in this paper have finite dimension, and all complexes are finite collections of simplices. A subset $\mathcal{L} \subseteq \mathcal{K}$ is a *subcomplex* of $\mathcal{K}$ if it is a complex itself. It is *proper* if $\mathcal{L} \neq \mathcal{K}$. Note that $\mathcal{L}$ inherits (ii) from $\mathcal{K}$, so it is a subcomplex iff $\mathcal{L}$ has property (i). The *underlying space* of $\mathcal{L}$, denoted $|\mathcal{L}|$, is the set of all points in $\mathbb{R}^d$ contained in at least one simplex of $\mathcal{L}$. A particular subcomplex of $\mathcal{K}$ is its *k-skeleton* $\mathcal{K}^{(k)} = \{\sigma \in \mathcal{K} \mid \dim \sigma \leq k\}$. For example, the 1-skeleton of $\mathcal{K}$ is a simple graph in $\mathbb{R}^d$. The *components* of $\mathcal{K}$ are the equivalence classes of the transitive closure of the incidence relation. If $\mathcal{K}$ has only one component then it is *connected*. Since $\mathcal{K}$ is a complex it is connected iff $\mathcal{K}^{(1)}$ is connected.

An *imbedding* of a topological space $\mathbb{A}$ in another such space $\mathbb{B}$ is a continuous one-to-one map from $\mathbb{A}$ to $\mathbb{B}$. It is a *homeomorphism* if it is also onto and its inverse is continuous. $\mathbb{A}$ and $\mathbb{B}$ are *homeomorphic* if there is a homeomorphism between $\mathbb{A}$ and $\mathbb{B}$. A *triangulation* of $\mathbb{B}$ is a simplicial complex, $\mathcal{T}$, whose underlying space, $|\mathcal{T}|$, is homeomorphic to $\mathbb{B}$. Clearly, the underlying space of every subcomplex of $\mathcal{T}$ has an imbedding in $\mathbb{B}$. We say the subcomplex is *imbeddable* in $\mathbb{B}$. We are particularly interested in triangulations of the d-sphere, $\mathbb{S}^d$, and in subcomplexes of such triangulations. The usual model of $\mathbb{S}^d$ is the set of points $x \in \mathbb{R}^{d+1}$ with unit distance from the origin. If $\mathcal{K}$ is a proper subcomplex of a triangulation of $\mathbb{S}^d$ then it is imbeddable in $\mathbb{R}^d$, e.g. by stereographic projection.

Boundary, cycles, and homology. Each k-simplex of a complex $\mathcal{K}$ can be *oriented* by assigning a linear ordering on its vertices, denoted $\sigma = [u_0, u_1, \ldots, u_k]$. Two orientations are the *same* if one sequence differs from the other by an even number of transpositions. The *boundary* of σ is

$$\partial_k \sigma = \sum_{i=0}^{k} (-1)^i [u_0, u_1, \ldots, \hat{u_i}, \ldots, u_k],$$

where the hat means that u_i is omitted. If σ is a vertex then $\partial_0 \sigma = 0$. So ∂_k maps each oriented k-simplex to a formal sum of oriented $(k-1)$-simplices. A formal sum of integer multiples of oriented k-simplices is called a *k-chain*. If the coefficient of a k-simplex in some k-chain is non-zero we say the simplex *belongs* to the chain. The free abelian group of k-chains from $\mathcal{K}$ is denoted as $\mathrm{C}_k = \mathrm{C}_k(\mathcal{K})$. The map ∂_k naturally extends to the *boundary homomorphism* $\partial_k : \mathrm{C}_k \rightarrow \mathrm{C}_{k-1}$ defined by

$$\partial_k(\sum_j a_j \sigma_j) = \sum_j a_j \partial_k \sigma_j,$$

where the a_j are integers and the σ_j are k-simplices of $\mathcal{K}$. Note that a $(k-1)$-simplex can occur in more than one term of this sum. The coefficients of these terms are added in the usual way.

A number of interesting groups can be defined using the boundary homomorphism. The group of *k-cycles*, $\mathrm{Z}_k = \mathrm{Z}_k(\mathcal{K})$, is the kernel of ∂_k, that is, the subgroup of k-chains $z \in \mathrm{C}_k$ with $\partial_k z = 0$. The group of *k-boundaries*, $\mathrm{B}_k = \mathrm{B}_k(\mathcal{K})$, is the image of ∂_{k+1}, that is, the subgroup of k-chains $z \in \mathrm{C}_k$ for which there exists a $(k+1)$-chain $z' \in \mathrm{C}_{k+1}$ with $z = \partial_{k+1} z'$. It can be shown that $\partial_k \partial_{k+1} z' = 0$ for every $(k+1)$-chain z', so B_k is a subgroup of Z_k. Finally, the quotient group $\mathrm{H}_k = \mathrm{H}_k(\mathcal{K}) = \mathrm{Z}_k | \mathrm{B}_k$ is the *k-th homology group* of $\mathcal{K}$.

The groups C_k, Z_k, B_k, and H_k are abelian and finitely generated, and with the possible exception of H_k they are free. By the fundamental theorem for finitely generated abelian groups, see e.g. [11], such a group, G, is isomorphic to a direct sum

$$\mathbb{Z}^{\beta} \oplus \mathbb{Z}|t_1 \oplus \mathbb{Z}|t_2 \oplus \ldots \oplus \mathbb{Z}|t_\ell,$$

where β is a non-negative integer, and the *torsion coefficients*, t_i, are integers greater than or equal to 2 so that t_i divides t_{i+1}, for $1 \leq i \leq \ell - 1$. The subgroup of G isomorphic to $\mathbb{Z}|t_1 \oplus \ldots \oplus \mathbb{Z}|t_\ell$ is its *torsion subgroup*, T. The rank of the quotient group G|T is the *betti number* of G, $\beta(\mathrm{G}) = \beta$. If G is free then T is trivial, that is, G is isomorphic to $\mathbb{Z}^{\beta}$, denoted $\mathrm{G} \cong \mathbb{Z}^{\beta}$. In this case G is *torsion-free.*

Set $\mathrm{G} = \mathrm{H}_k$ and let T_k be its torsion subgroup. Then $\beta_k = \beta_k(\mathcal{K}) = \beta(\mathrm{H}_k)$ is called the *k-th betti number* of $\mathcal{K}$. Each element of H_k is a *coset* of *homologous k-cycles*. Particularly, two k-cycles, z_1 and z_2, belong to the same coset iff $z_1 - z_2$ is the boundary of some $(k+1)$-chain.

For simplicial complexes imbeddable in $\mathbb{S}^3$, the homology groups are free, so the torsion subgroups are trivial, see e.g. [1, chapter X, §1]. Their betti numbers lend themselves to clear geometric interpretations. These are based on the important result that the homology groups of two complexes, $\mathcal{K}$ and $\mathcal{L}$, are the same if $|\mathcal{K}|$ is homeomorphic to $|\mathcal{L}|$. Hence, it is possible to talk about the homology groups and betti numbers of a topological space rather than of its triangulations. The 0-th betti number, β_0, is the number of connected pieces, the 1-st betti number, β_1, is the number of independent "tunnels", and the 2-nd betti number, β_2, is the number of "holes" of $|\mathcal{K}|$.

Mayer-Vietoris sequences. Let $A_1, A_2, \ldots, A_{m+1}$ be abelian groups and $\eta_i : A_i \rightarrow A_{i+1}$ homomorphisms between them. The sequence

$$A_1 \xrightarrow{\eta_1} A_2 \xrightarrow{\eta_2} \ldots \xrightarrow{\eta_m} A_{m+1}$$

is *exact* if the image of η_i, $\text{im}\eta_i$, is the same as the kernel of η_{i+1}, $\ker\eta_{i+1}$, for $1 \le i \le m-1$. If $m = 4$ and A_1 and A_5 are trivial then this is a *short exact sequence*:

$$0 \rightarrow A_2 \xrightarrow{\eta_2} A_3 \xrightarrow{\eta_3} A_4 \rightarrow 0.$$

Call $\text{cok}\eta_i = A_{i+1}|\text{im}\eta_i$ the *cokernel* of η_i. An important property of short exact sequences is that $A_2 \cong \text{im}\eta_2$ and $A_4 = \text{im}\eta_3$. Since $\text{im}\eta_2 = \ker\eta_3$ by definition, we have $\text{cok}\eta_2 = A_3|\text{im}\eta_2 \cong A_4$, and therefore $\beta(A_3) = \beta(A_2) + \beta(A_4)$. Furthermore, if

$$A_1 \xrightarrow{\eta_1} A_2 \rightarrow A_3 \rightarrow A_4 \xrightarrow{\eta_4} A_5$$

is exact then so is

$$0 \rightarrow \text{cok}\eta_1 \rightarrow A_3 \rightarrow \ker\eta_4 \rightarrow 0.$$

Let now $\mathcal{K}$, $\mathcal{K}'$, and $\mathcal{K}''$ be complexes so that $\mathcal{K} = \mathcal{K}' \cup \mathcal{K}''$, and define $\mathcal{L} = \mathcal{K}' \cap \mathcal{K}''$. The Mayer-Vietoris sequence relates the homology groups of $\mathcal{K}$, $\mathcal{K}'$, $\mathcal{K}''$, and $\mathcal{L}$, see e.g. [11, chapter 3, §25]:

2.1 There is an exact sequence
$$\ldots \rightarrow \text{H}_k(\mathcal{L}) \xrightarrow{\lambda_k} \text{H}_k(\mathcal{K}') \oplus \text{H}_k(\mathcal{K}'') \rightarrow \text{H}_k(\mathcal{K}) \rightarrow \text{H}_{k-1}(\mathcal{L}) \xrightarrow{\lambda_{k-1}} \text{H}_{k-1}(\mathcal{K}') \oplus \text{H}_{k-1}(\mathcal{K}'') \rightarrow \ldots$$

As mentioned above, it follows that

$$0 \rightarrow \text{cok}\lambda_k \rightarrow \text{H}_k(\mathcal{K}) \rightarrow \ker\lambda_{k-1} \rightarrow 0,$$

is a short exact sequence. Let $\text{N}_{k-1} = \ker\lambda_{k-1}$; it is the subgroup of $\text{H}_{k-1}(\mathcal{L})$ defined by the $(k-1)$-cycles that bound in $\mathcal{K}'$ and also in $\mathcal{K}''$. Now, $\text{cok}\lambda_k = \text{H}_k(\mathcal{K}') \oplus \text{H}_k(\mathcal{K}'')|\text{im}\lambda_k$, and $\text{im}\lambda_k \cong \text{H}_k(\mathcal{L})|\ker\lambda_k$. Hence we get the following corollary of the Mayer-Vietoris sequences that can also be found in [1, chapter VII, §2].

2.2 For all k, $\beta_k(\mathcal{K}) = \beta_k(\mathcal{K}') + \beta_k(\mathcal{K}'') - \beta_k(\mathcal{L}) + \beta(\text{N}_k) + \beta(\text{N}_{k-1})$.

3 The Incremental Method

The basic idea of the algorithm is to build up a complex by adding one simplex at a time. At each step the betti numbers of the current complex are updated to reflect the changes in the homology groups. We first make this abstract view of the algorithm more precise and then prove its correctness using Mayer-Vietoris sequences. The supporting data structures and the analysis will be given in sections 4 and 5.

The abstract incremental method. The input to the algorithm is a simplicial complex, $\mathcal{K} = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$, that is imbeddable in $\mathbb{S}^d$. If $\mathcal{K}$ is not a complete triangulation of $\mathbb{S}^d$, it is imbeddable in $\mathbb{R}^d$. For each $0 \le i \le m$ define $\mathcal{K}_i = \{\sigma_1, \sigma_2, \ldots, \sigma_i\}$. We say the ordering (indexing) of the simplices has the *prefix property* if every $\mathcal{K}_i$ is a genuine complex. The largest dimension of any simplex can be at most d, so except possibly for dimensions between 0 and d inclusive, all betti numbers of $\mathcal{K}$ vanish. To compute $\beta_\ell(\mathcal{K})$, for $0 \le \ell \le d$, we use the following incremental method.

```
for $\ell := 0$ to $d$ do $b_\ell := 0$ endfor;
for $i := 0$ to $m$ do
    $k := \dim\sigma_i$;
    if $\sigma_i$ belongs to a $k$-cycle of $\mathcal{K}_i$
      then $b_k := b_k + 1$
      else $b_{k-1} := b_{k-1} - 1$
    endif
endfor.
```

Note that if σ_i is a vertex, so $\dim\sigma_i = 0$, then σ_i belongs to a 0-cycle by definition. Hence there will be no access to an undefined variable b_{-1}. The above algorithm is complete if we can give a concrete method for deciding whether or not σ_i belongs to a k-cycle of $\mathcal{K}_i$. We will sloppily refer to this operation as "detecting a k-cycle." This will be discussed in section 4.

Correctness proof. All algorithms in this paper are derivatives of the incremental method. Its correctness is based on the prefix property of the simplex sequence and on the corollary of the Mayer-Vietoris sequences theorem stated as 2.2.

Recall from section 2 that the Mayer-Vietoris sequence is defined for a complex, $\mathcal{K}$, and three subcomplexes, $\mathcal{K}'$, $\mathcal{K}''$, and $\mathcal{L}$, so that $\mathcal{K} = \mathcal{K}' \cup \mathcal{K}''$ and $\mathcal{L} = \mathcal{K}' \cap \mathcal{K}''$. In our application we have $\mathcal{K} = \mathcal{K}_i$, $\mathcal{K}' = \mathcal{K}_{i-1}$, and $\mathcal{K}''$ the complex consisting of σ_i and its faces. We prove the correctness of the incremental method inductively. The assumption is that for all ℓ, $0 \le \ell \le d$, the algorithm correctly computes the ℓ-th betti number of $\mathcal{K}_{i-1}$. Locally within this proof we use index pairs and define $H_{\ell,i} = H_\ell(\mathcal{K}_i)$ and $\beta_{\ell,i} = \beta_\ell(\mathcal{K}_i)$. So after processing σ_1 through σ_{i-1} the variable b_ℓ stores the correct ℓ-th betti number of $\mathcal{K}_{i-1}$, that is, $b_\ell = \beta_{\ell,i-1}$. Assume that σ_i is a k-simplex and note that the only betti numbers of $\mathcal{K}_i$ that can possibly be different from the corresponding ones of $\mathcal{K}_{i-1}$ are the k-th and the $(k-1)$-st. The case $k = 0$ is handled correctly because $\mathcal{K}_i$ differs from $\mathcal{K}_{i-1}$ only by one vertex, which forms its own component in $\mathcal{K}_i$. So assume $k \geq 1$. As mentioned before, $\mathcal{K}''$ consists of σ_i and its faces. Hence

$$\beta_j(\mathcal{K}'') = \begin{cases} 1 & \text{if } j = 0, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

Furthermore, $\mathcal{L} = \mathcal{K}_{i-1} \cap \mathcal{K}'' = \mathcal{K}'' - \{\sigma_i\}$ is the boundary complex of σ_i. Hence

$$\beta_j(\mathcal{L}) = \begin{cases} 2 & \text{if } k = 1 \text{ and } j = 0, \\ 1 & \text{if } k \geq 2 \text{ and } j = k-1, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

All groups N_j, for $j \leq k-2$, are trivial because every j-cycle bounds in $\mathcal{K}_{i-1}$ and also in $\mathcal{K}''$. The betti number of N_{k-1} depends on whether or not $\partial_k \sigma_i$ bounds in $\mathcal{K}_{i-1}$; it always bounds in $\mathcal{K}''$.

Case 1. σ_i belongs to a k-cycle in $\mathcal{K}_i$, so $\partial_k \sigma_i$ bounds in $\mathcal{K}_{i-1}$. Then

$$\beta(N_j) = \begin{cases} 1 & \text{if } j = k-1, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

We can now use 2.2 to compute the ℓ-th betti number, for $\ell = k$ and $\ell = k-1$. In both cases $\beta_{\ell,i} = \beta_{\ell,i-1} + \beta_\ell(\mathcal{K}'') - \beta_\ell(\mathcal{L}) + \beta(N_\ell) + \beta(N_{\ell-1})$. For $\ell = k$ we get $\beta_k(\mathcal{K}'') = \beta_k(\mathcal{L}) = \beta(N_k) = 0$ and $\beta(N_{k-1}) = 1$, hence $\beta_{k,i} = \beta_{k,i-1} + 1$. For $\ell = k-1$ we get $\beta_{k-1}(\mathcal{K}'') - \beta_{k-1}(\mathcal{L}) = -1$, $\beta(N_{k-1}) = 1$, and $\beta(N_{k-2}) = 0$, so $\beta_{k-1,i} = \beta_{k-1,i-1}$. Both results are consistent with the actions of the incremental method.

Case 2. σ_i does not belong to any k-cycle of $\mathcal{K}_i$, so $\partial_k \sigma_i$ does not bound in $\mathcal{K}_{i-1}$. Then $\beta(N_j) = 0$ for all j, and hence $\beta(N_\ell) + \beta(N_{\ell-1}) = 0$ for $\ell = k, k-1$. The contribution of N_ℓ and $N_{\ell-1}$ is thus one less than in case 1. Therefore, $\beta_{k,i} = \beta_{k,i-1}$ and $\beta_{k-1,i} = \beta_{k-1,i-1} - 1$, which is again consistent with the actions of the incremental method. This completes its correctness proof.

Remark. It is interesting to note that the corollary of the Mayer-Vietoris sequences expressed by the incremental method is sufficient to imply a classic theorem on the Euler number of a complex. Let ν_k denote the number of k-simplices of $\mathcal{K}$. The *Euler number* of $\mathcal{K}$ is defined as $\chi = \sum_{k=0}^{d} (-1)^k \nu_k$. The theorem asserts that

3.1 $\chi = \sum_{k=0}^{d} (-1)^k \beta_k$,

see e.g. [11, chapter 2, §22]. It follows from the correctness of the algorithm as follows. Let ν'_k be the number of k-simplices $\sigma_i \in \mathcal{K}$ so that σ_i belongs to a k-cycle of $\mathcal{K}_i$. Hence $\nu_k = \nu'_k + \nu''_k$, where ν''_k is the number of k-simplices σ_i that do not belong to any k-cycle of $\mathcal{K}_i$. Following the computations we get $\beta_k = \nu'_k - \nu''_{k+1}$. So

$$\begin{aligned} \chi &= \sum_{k=0}^{d} (-1)^k \nu_k = \sum_{k=0}^{d} (-1)^k (\nu'_k + \nu''_k) \\ &= \sum_{k=0}^{d} (-1)^k (\nu'_k - \nu''_{k+1}) = \sum_{k=0}^{d} (-1)^k \beta_k \end{aligned}$$

because $\nu''_0 = \nu''_{d+1} = 0$.

4 Supporting Data Structures

Vertices trivially belong to 0-cycles, so they do not need any data structure support to distinguish between cases. We have good data structures for detecting and 1-cycles, and for detecting $(d-1)$-cycles provided the complex of interest is a subcomplex of a triangulation of $\mathbb{S}^d$. For $d \leq 3$ we thus can cover all cases and get an efficient algorithm. We first discuss 1-cycles. The solution for $(d-1)$-cycles is similar.

Detecting 1-cycles. Let σ_i be a 1-simplex. It belongs to a 1-cycle of $\mathcal{K}_i$ iff it belongs to a 1-cycle of $\mathcal{K}_i^{(1)}$. $\mathcal{K}_i^{(1)}$ is a graph, and various efficient methods for detecting 1-cycles (cycles) in graphs are known, see e.g. [3]. For completeness we describe the method that fits best into our framework. It is based on a data structure for the so-called union-find problem.

A *union-find* data structure represents a collection of *elements* partitioned into a system of pairwise disjoint *sets*. It supports the following types of operations.

ADD(u):	Add u as the only element of a singleton set, $\{u\}$, to the system.
FIND(u):	Determine and return (the name of) the set that contains u.
UNION(A, B):	Replace the sets A and B by their union, $A \cup B$.

In our application, the elements are the vertices of the 1-skeleton and the sets correspond to its components. Initially, the system is empty. The union-find structure needs to be updated only if σ_i is a vertex or an edge; two- and higher-dimensional simplices can be ignored. Assume the union-find structure represents $\mathcal{K}_{i-1}^{(1)}$, and consider the next simplex, σ_i. If σ_i is a vertex then ADD(σ_i) adds it to the system. If σ_i is an edge connecting vertices u and v then we find the corresponding sets, $A :=$ FIND(u) and $B :=$ FIND(v). If $A = B$ then u and v belong to the same component of $\mathcal{K}_{i-1}^{(1)}$ and thus σ_i belongs to a 1-cycle. Otherwise, σ_i does not belong to any 1-cycle in $\mathcal{K}_i^{(1)}$. Rather it connects two components which must now be merged. This is done by calling UNION(A, B).

Detecting $(d-1)$-cycles. To detect $(d-1)$-cycles we assume that $\mathcal{K}$ is a subcomplex of a triangulation $\mathcal{T} = \{\sigma_1, \sigma_2, \ldots, \sigma_m, \ldots, \sigma_n\}$ of $\mathbb{S}^d$, and that the ordering of the simplices has the prefix property. For $0 \leq i \leq n$ define $\mathcal{K}_i = \{\sigma_1, \sigma_2, \ldots, \sigma_i\}$ and $\bar{\mathcal{K}}_i = \mathcal{T} - \mathcal{K}_i$. Note that $\bar{\mathcal{K}}_i$ is in general not a complex. However, it satisfies the reverse of property (i), namely (i') if σ' is a face of σ and $\sigma' \in \mathcal{K}$ then $\sigma \in \mathcal{K}$. This can be used to characterize the $(d-1)$-cycles of $\mathcal{K}_i$ in terms of the components of a graph. Let V be the set of d-simplices of $\bar{\mathcal{K}}_i$, and let E be the set of pairs of d-simplices, $\{a, b\}$, so that $a \cap b$ is a $(d-1)$-simplex in $\bar{\mathcal{K}}_i$. The graph $\mathcal{G}_i$ with node set V and arc set E is termed the *dual graph* of $\bar{\mathcal{K}}_i$.

Let σ_i be a $(d-1)$-simplex. It is fairly intuitive that σ_i belongs to a $(d-1)$-cycle in $\mathcal{K}_i$ iff $\mathcal{G}_i$ has one more component than $\mathcal{G}_{i-1}$. This can formally be proved using cohomology groups and duality theorems in algebraic topology. We omit the formal argument and refer to [11] for a good introduction of these concepts. So this means that σ_i belongs to a $(d-1)$-cycle in $\mathcal{K}_i$ iff it does not belong to a 1-cycle in $\mathcal{G}_{i-1}$. Adding a simplex to $\mathcal{K}_{i-1}$ means removing the same simplex from $\bar{\mathcal{K}}_{i-1}$. Hence it appears that the dual graph must be maintained through a sequence of node and arc removals, which is computationally more expensive than a similar sequence of node and arc insertions. For this reason we reverse the processing order of the simplices and obtain the empty complex by starting with $\mathcal{T}$ and removing a simplex at a time. This is done only for detecting $(d-1)$-cycles and does not affect other computations.

The data structure used to represent $\mathcal{G}_i$, and thus $\bar{\mathcal{K}}_i$, is again a union-find structure. Its elements are the nodes of $\mathcal{G}_i$ (the d-simplices of $\bar{\mathcal{K}}_i$), and the sets in the system represent the components of $\mathcal{G}_i$. Initially, $i = n$, $\mathcal{G}_n = (\emptyset, \emptyset)$, and the system that represents $\mathcal{G}_n$ is empty. The representation of $\mathcal{G}_0$ (the dual graph of $\mathcal{T} = \mathcal{T} - \emptyset$) is built by processing the simplices σ_n down to σ_1. Of course, only d- and $(d-1)$-simplices have any effect on the data structure.

To go from $\mathcal{G}_i$ to $\mathcal{G}_{i-1}$ we add the simplex σ_i to $\bar{\mathcal{K}}_i$. If σ_i is a d-simplex then ADD(σ_i) adds it to the system. In the forward direction this corresponds to removing an isolated node of $\mathcal{G}_{i-1}$ to obtain $\mathcal{G}_i$. If σ_i is a $(d-1)$-simplex then an arc connecting the two incident d-simplices is added to $\mathcal{G}_i$, resulting in $\mathcal{G}_{i-1}$. Using two FIND operations, we can test whether or not the two d-simplices belong to the same component of $\mathcal{G}_i$. If they do then no further action is required. Otherwise, the two d-simplices belong to two different components, represented by two sets $A \neq B$ in the system. These two sets are merged by calling UNION(A, B). In the forward direction this corresponds to splitting a component. The communication between the main algorithm, which runs forward, and the $(d-1)$-cycle detection mechanism, which runs backward, is based on marks left with $(d-1)$-simplices σ_i that belong to a $(d-1)$-cycle of $\mathcal{K}_i$.

5 The Algorithm

After establishing the ingredients in sections 3 and 4, we put things together to obtain the algorithm in complete detail. We have good data structures only for 1- and for $(d-1)$-cycles, so we get a good algorithm only for $d \leq 3$. For $d \geq 4$ we cannot even compute the 1-st homology group because this requires detecting 1- *and* 2-cycles. The algorithm is described for $d = 3$. It assumes that the input consists of a triangulation of $\mathbb{S}^3$ of which the complex of interest is a subcomplex. If only the complex is given it is necessary to first construct a compatible triangulation of the complement of its underlying space. We refer to [2] for an algorithm that constructs such a triangulation.

The incremental algorithm. Let $\mathcal{T}$ be a triangulation of $\mathbb{S}^3$ with simplices $\sigma_1, \sigma_2, \ldots, \sigma_n$, and define $\mathcal{K}_i = \{\sigma_1, \sigma_2, \ldots, \sigma_i\}$, for $0 \leq i \leq n$, as before. We assume that each $\mathcal{K}_i$ is a complex. The complex of interest is $\mathcal{K} = \mathcal{K}_m$, with $m \leq n$.

The first step of the algorithm marks every simplex, σ_i, that belongs to a cycle of the same dimension in $\mathcal{K}_i$. Each vertex belongs to a 0-cycle, so all vertices get marked. To mark the appropriate edges we process the simplices in forward direction and maintain a union-find structure for $\mathcal{K}_i^{(1)}$. An edge is marked iff it does *not* cause a UNION operation. For marking the appropriate triangles we process the simplices backwards, from σ_n down to σ_1. A union-find structure representing the dual graph, $\mathcal{G}_i$ of $\bar{\mathcal{K}}_i = \mathcal{T} - \mathcal{K}_i$, is maintained, and a triangle is marked iff it causes a UNION operation. Finally, the only tetrahedron that belongs to a 3-cycle at the time it is processed is σ_n. This is the only tetrahedron that gets marked.

The second step counts the marked and unmarked simplices and derives the betti numbers as simple sums of these numbers. This is done by scanning the simplices once more, in forward direction.

```
$b_0 := b_1 := b_2 := b_3 := 0$;
for $i := 1$ to $m$ do
    $k := \dim \sigma_i$;
    if $\sigma_i$ is marked then $b_k := b_k + 1$
                        else $b_{k-1} := b_{k-1} - 1$
    endif
endfor.
```

The only case where we get $b_3 \neq 0$ is when $\mathcal{K}$ contains all tetrahedra of $\mathcal{T}$ and thus is a triangulation of $\mathbb{S}^3$. If $\mathcal{K}$ is imbeddable in $\mathbb{R}^3$ then $\mathcal{K} \neq \mathcal{T}$ and we can drop b_3 from the algorithm.

The analysis. The vertices, edges, triangles, and tetrahedra in $\mathcal{T}$ cause different actions in the algorithm. Let ν_k be the number of k-simplices in $\mathcal{T}$, for $0 \leq k \leq 3$. Observe that $2\nu_3 = \nu_2 \geq \nu_1 \geq 2\nu_0$. Since $n = \nu_0 + \nu_1 + \nu_2 + \nu_3$ we have $n \leq 3\nu_2$ and $n \leq 6\nu_3$, that is, at least one third of all simplices are triangles and at least one sixth of them are tetrahedra.

It is clear that step 2 of the algorithm takes only $O(n)$ time. Similarly, the vertices and the last tetrahedron can be marked in time $O(n)$. The forward process that marks the edges executes a sequence of ν_0 ADD operations, $2\nu_1$ FIND operations, and at most $\nu_0 - 1$ UNION operations. Using a standard implementation of the union-find structure, see e.g. [3], this takes time $O((\nu_0 + \nu_1)\alpha(\nu_0, \nu_1))$, where $\alpha(x, y)$ is the extremely slowly growing inverse of Ackermann's function. Similarly, the backwards process that marks the triangles executes ν_3 ADD operations, $2\nu_2$ FIND operations, and at most $\nu_3 - 1$ UNION operations. This takes time $O((\nu_3 + \nu_2)\alpha(\nu_3, \nu_2))$. Recall the customary notation $\alpha(x) = \alpha(x, x)$.

5.1 Let $\mathcal{K}$ be a subcomplex of a triangulation $\mathcal{T}$ of $\mathbb{S}^3$, with $|\mathcal{T}| = n$. The possibly non-vanishing betti numbers of $\mathcal{K}$, $\beta_0, \beta_1, \beta_2, \beta_3$, can be computed in time $O(n\alpha(n))$ and storage $O(n)$.

Remark. If $\mathcal{K}$ is a subcomplex of a triangulation of $\mathbb{S}^2$ then no backwards computation is necessary. Hence, there is no need to consider any of the simplices that do not belong to $\mathcal{K}$. The result can then be improved to time $O(m\alpha(m))$ and storage $O(m)$.

Using depth-first search. Consider the case where the complex is represented by a data structure so that for a given σ the simplices incident to σ can be accessed in constant time. An example of such a data structure is the *adjacency-list representation* which is common for graphs. The nodes are elements of a linear array. An arc is given as an index pair, so the incident nodes can be found in constant time by array look-up. The arcs incident to a node are represented by a linear list whose address is stored with the node. Given a node it is thus possible to access the incident arcs in constant time per arc.

For our purpose it will be sufficient to have an adjacency-list representation for $\mathcal{T}^{(1)}$, the 1-skeleton of $\mathcal{T}$, and for $\mathcal{G}_0$, the dual graph of $\bar{\mathcal{K}}_0 = \mathcal{T}$. Depth-first search is a standard graph search method that takes constant time per arc (edge or triangle) and can distinguish between arcs that complete a cycle and arcs that connect to a new node, see e.g. [3]. Using the data structure for $\mathcal{T}^{(1)}$ we can use depth-first search to properly mark the edges of $\mathcal{T}$. Using the data structures for $\mathcal{G}_0$ we can use depth-first search to properly mark its triangles. It is important to notice that the two depth-first searches are not coordinated with each other. Indeed, to achieve $O(n)$ running time the search of $\mathcal{T}^{(1)}$ needs the freedom to visit the edges in any order it pleases. Similar for $\mathcal{G}_0$. Fortunately, every sequence in which vertices precede edges, edges precede triangles, and triangles precede tetrahedra has the prefix property. In particular such a sequence in which the edges are ordered how they are visited by the search of $\mathcal{T}^{(1)}$ and the triangles are ordered in reverse how they are visited by the search of $\mathcal{G}_0$ has the prefix property. The betti numbers can now be computed by traversing this sequence and counting marked and unmarked simplices as before. This leads to the following improvement of 5.1.

5.2 Let $\mathcal{K}$ be a subcomplex of a triangulation $\mathcal{T}$ of $\mathbb{S}^3$, with $|\mathcal{T}| = n$, and assume the 1-skeleton and the dual graph of $\mathcal{T}$ are given by their adjacency-list representations. Then the betti numbers of $\mathcal{K}$ can be computed in time and storage $O(n)$.

Remark. The improvement with depth-first search sacrifices the ability to prescribe the order of the simplices. This ability is crucial for our application to α-shapes discussed in the next section.

6 Signatures for Alpha Shapes

A comprehensive discussion of the family of α-shapes of a finite point set is beyond the scope of this paper. As a substitute we refer the reader to papers on two-dimensional [7], three-dimensional [8], and weighted as well as higher-dimensional α-shapes [6]. Computing betti numbers for α-shapes is what really motivates us to develop the algorithm in this paper.

A brief description of α-shapes. Let S be a finite point set in $\mathbb{R}^d$, with $|S| \geq d + 1$, and let $\mathcal{D} = \mathcal{D}(S)$ be its Delaunay triangulation, see e.g. [5]. Provided the points are in general position, $\mathcal{D}$ is indeed a simplicial complex. Commonly, it is defined so that its underlying space, $|\mathcal{D}|$, is the convex hull of S. It is more convenient for us to add a point "at infinity" and connect it to all simplices on the boundary of the convex hull of S. The resulting complex is a triangulation of $\mathbb{S}^d$, which we denote by $\mathcal{T}$.

For each non-negative real α, the *α-complex* of S is a subcomplex of $\mathcal{D}$ and therefore also of $\mathcal{T}$. The *α-shape* of S is the underlying space of the α-complex. It is defined so that for $\alpha = 0$ we get the point set S itself and for $\alpha = +\infty$ we get the convex hull of S, see [6, 8].

Although an α-shape is defined for every non-negative real α, there are only finitely many different subcomplexes of $\mathcal{D}$ and therefore only finitely many different α-shapes. It is convenient to index the α-shapes and α-complexes by position. Let s be the number of different α-complexes, denoted $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_s$. For $1 \leq i \leq s$, the α-shape that corresponds to the i-th α-complex is $\mathcal{S}_i = |\mathcal{C}_i|$. With increasing index the corresponding α-value also grows. For convenience, define $\mathcal{C}_0 = \emptyset$ and $\mathcal{C}_{s+1} = \mathcal{T}$. An important property of the resulting sequence of complexes is that each is a proper subcomplex of the next one, that is,

$$\mathcal{C}_0 \subset \mathcal{C}_1 \subset \mathcal{C}_2 \subset \ldots \subset \mathcal{C}_s \subset \mathcal{C}_{s+1},$$

see [6, 8]. It follows that $s + 1 \leq n$, the number of simplices in $\mathcal{T}$.

For each simplex $\sigma_i \in \mathcal{T}$, let $\lambda(i)$ be the smallest index ℓ so that $\sigma_i \in \mathcal{C}_\ell$. We order and reindex the simplices of $\mathcal{T}$ as $\sigma_1, \sigma_2, \ldots, \sigma_n$ so that if $i < j$ then

(i) $\lambda(i) < \lambda(j)$ or

(ii) $\lambda(i) = \lambda(j)$ and $\dim \sigma_i \leq \dim \sigma_j$.

As before we define $\mathcal{K}_i = \{\sigma_1, \sigma_2, \ldots, \sigma_i\}$ for $0 \leq i \leq n$. Clearly, $\mathcal{K}_0 = \mathcal{C}_0$ and $\mathcal{K}_n = \mathcal{C}_{s+1}$. Property (i) implies that the sequence of α-complexes is a scattered subsequence of the sequence of $\mathcal{K}_i$. Indeed, $\mathcal{C}_{\lambda(i)}$ is the smallest α-complex that contains $\mathcal{K}_i$, and $\mathcal{C}_{\lambda(i)} = \mathcal{K}_i$ iff $\lambda(i) \neq \lambda(i+1)$. Property (ii) and the fact that each $\mathcal{C}_{\lambda(i)}$ is a complex imply that each $\mathcal{K}_i$ is a complex. In other words, the ordering of the simplices has the prefix property.

Computing Signatures. The implementation of three-dimensional α-shapes reported in [8] includes a small number of signature functions that follow the evolution of the α-shape as α increases from 0 to $+\infty$. Let $[s]$ denote the set $\{1, 2, \ldots, s\}$. By a *signature function* we mean a function $f : [s] \rightarrow R$ that maps each index $\ell \in [s]$ to a value $f(\ell)$ in some range R. For reasons of usefulness the function should be defined so that $f(\ell)$ expresses some property of the α-shape $\mathcal{S}_\ell$. For example, $f(\ell)$ could express a combinatorial property, such as the number of triangles bounding $\mathcal{S}_\ell$, or a metric property, such as the surface area of $\mathcal{S}_\ell$.

In this section we are interested in three topological signature functions that count the number of components, independent tunnels, and holes of $\mathcal{S}_\ell$. For $0 \leq k \leq 2$ define

$$\beta_k : [s] \longrightarrow \mathbb{Z}$$

so that $\beta_k(\ell)$ is the k-th betti number of $\mathcal{S}_\ell$. The homology groups of $\mathcal{S}_\ell$ and $\mathcal{C}_\ell$ are the same, so $\beta_k(\ell) = \beta(\mathrm{H}_k)$ where H_k is the k-th homology group of $\mathcal{C}_\ell$. Each signature function, β_k, is represented by a linear array, $b_k[1..s]$.

We can now modify the algorithm of section 5 to compute the signature functions β_0, β_1, and β_2 of all α-shapes of S. Step 1, which marks the simplices, is exactly the same as in section 5. The only change in step 2 is that for some values of i the computed betti numbers need to be stored in the appropriate elements of the three arrays.

```
$b_0[1] := b_1[1] := b_2[1] := 0$; $\ell := 1$;
for $i := 1$ to $n$ do
    $k := \dim \sigma_i$;
    if $\sigma_i$ is marked then $b_k[\ell] := b_k[\ell] + 1$
                            else $b_{k-1}[\ell] := b_{k-1}[\ell] - 1$
    endif;
    if $\ell < s$ and $\lambda(i) \neq \lambda(i+1)$ then
        $b_0[\ell+1] := b_0[\ell]$; $b_1[\ell+1] := b_1[\ell]$;
        $b_2[\ell+1] := b_2[\ell]$; $\ell := \ell + 1$
    endif
endfor.
```

Clearly the asymptotic complexity of this algorithm is the same as of the algorithm in section 5. We thus obtain the main result of this section.

6.1 The signature functions that map $\ell \in [s]$ to the 0-th, 1-st, and 2-nd betti numbers of $\mathcal{S}_\ell$ can be computed in time $\mathrm{O}(n\alpha(n))$ and storage $\mathrm{O}(n)$.

We note that everything said about three-dimensional α-shapes also applies to weighted three-dimensional α-shapes [6].

7 Discussion

This paper presents an incremental algorithm for computing the betti numbers of a topological space represented by a simplicial complex. It has an efficient implementation for complexes imbeddable in $\mathbb{R}^3$ or $\mathbb{S}^3$. The algorithm is an example of how algorithmic techniques developed for graphs can be applied and extended to complexes of dimension higher than one. It is to be hoped that this is a step towards a revived interest in algorithmic problems in algebraic topology. As demonstrated in this paper, these algorithms do not necessarily have algebraic flavor. Indeed, we see our algorithm as evidence that combinatorial algorithms can outperform algebraic methods designed to solve the same problems.

The most interesting unanswered question concerns data structures that give an efficient implementation of our incremental method for complexes not imbeddable in $\mathbb{S}^3$.

References

[1] P. Alexandroff and H. Hopf. *Topologie I.* Julius Springer, Berlin, 1935.

[2] M. Bern. Compatible tetrahedralizations. "Proc. 9th Ann. Sympos. Comput. Geom., 1993", to appear.

[3] T. H. Cormen, Ch. E. Leiserson and R. L. Rivest. *Introduction to Algorithms.* The MIT Press, Cambridge, Mass., 1990.

[4] B. R. Donald and D. R. Chang. On the complexity of computing the homology type of a triangulation. *In* "Proc. 32nd Ann. IEEE Sympos. Found. Comput. Sci., 1991", 650–661.

[5] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag, Heidelberg, Germany, 1987.

[6] H. Edelsbrunner. Weighted alpha shapes. Rept. UIUCDCS-R-92-1760, Comput. Sci. Dept., Univ. Illinois, Urbana, Illinois, 1992.

[7] H. Edelsbrunner, D. G. Kirkpatrick and R. Seidel. On the shape of a set of points in the plane. *IEEE Trans. Inform. Theory* **IT-29** (1983), 551–559.

[8] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Trans. Graphics*, to appear.

[9] P. J. Giblin. *Graphs, Surfaces, and Homology.* Second edition, Chapman and Hall, London, 1981.

[10] R. Kannan and A. Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM J. Comput.* **8** (1979), 499–507.

[11] J. R. Munkres. *Elements of Algebraic Topology.* Addison-Wesley, Redwood City, California, 1984.

[12] J. J. Rotman. *An Introduction to Algebraic Topology.* Springer-Verlag, New York, 1988.

[13] H. J. Smith. On systems of indeterminate equations and congruences. *Philos. Trans.* **151** (1861), 293–326.