On Computing the Homology Type of a Triangulation

Bruce Randall Donald*
David Renpang Chang†

TR 91-1183
December 1990
(revised August 1991)

Program of Computer Graphics
Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*Supported in part by the National Science Foundation under grant No. IRI-8802390 and by a Presidential Young Investigator award, and in part by the Mathematical Science Institute.
†Supported by the National Science Foundation under the REU program.

On Computing the Homology Type of a Triangulation

Bruce Randall Donald[*] **David Renpan Chang**[†]
Computer Science Department, Cornell University

Abstract: We analyze an algorithm for computing the homology type of a triangulation. By *triangulation* we mean a finite simplicial complex; its *homology type* is given by its homology groups (with integer coefficients). The algorithm could be used in computer-aided design to tell whether two finite-element meshes or Bézier-spline surfaces are of the same "topological type," and whether they can be embedded in $\Re^3$. Homology computation is a purely combinatorial problem of considerable intrinsic interest. While the worst-case bounds we obtain for this algorithm are poor, we argue that many triangulations (in general) and virtually all triangulations in design are very "sparse," in a sense we make precise. We formalize this sparseness measure, and perform a probabilistic analysis of the sparse case to show that the expected running time of the algorithm is roughly quadratic in the geometric complexity (number of simplices) and linear in the dimension.

[*]Supported in part by the National Science Foundation under grant No. IRI-8802390 and by a Presidential Young Investigator award, and in part by the Mathematical Science Institute.

[†]Supported by the National Science Foundation under the REU program.

Contents

1 Introduction

In this paper we study the complexity of computing all the homology groups of a triangulation. In computer-aided design (CAD), many geometric designs are triangulations. For example, finite-element methods use 2D and 3D triangulations of solids, called finite-element meshes. Triangular parametric surfaces such as rational Bézier surfaces are widely used in geometric modelling systems [War]. A collection of these objects is, topologically, a triangulation (a 3D finite-element tetrahedralization is simply a 3D triangulation). Furthermore, algorithms exist for triangulating polygons and tetrahedra (eg, [Che]); the problem of triangulating algebraic varieties has been studied by Hironaka, McCallum and others. In this paper we will adopt the following

Definition 1 *A* triangulation *is a finite simplicial complex.*

Thus, a triangular parametric surface will be regarded as a homeomorph of a triangulation. Since homology type is a topological invariant, it is the same for all homeomorphs of a triangulation, and for all other triangulations of the same geometric object.

In geometric design, we often construct triangulations (either by machine or by hand), and then modify them incrementally in the design process. These modifications can involve adding new triangles and tetrahedra, identifying edges and vertices, etc. We wish to ask, when are two triangulations "topologically equivalent?" For example, we may wish to compute whether two designs have the same "topological type." Alternatively, after modifying a design, we wish to know whether its "topology" has been altered. Indeed, the topological type of a design may be taken as a design specification, or as an invariant, which no modelling operation should violate. Finally, we are interested in the physical realizability of a geometric design. One basic question we may ask is, "Can the design be embedded in Euclidean Space ($\Re^3$)?" In general these questions are very hard. One way to attack this problem is to search for computable topological invariants. For example, one topological invariant we might consider is the fundamental group. As is well known, given a simplicial complex it is possible to "effectively compute" its fundamental group, in the sense that one may compute a presentation for that group. However, deciding the isomorphism of two groups from their presentations is uncomputable. Indeed, by showing that any finitely-presented group can be the fundamental group of a compact 4-manifold [ST], it has been shown that there exists no algorithm for deciding the topological equivalence (homeomorphism) of two compact, orientable, triangulable 4-manifolds [Mar].[1] One may wonder why we seek faster algorithms for computing homology groups. One motivation is that this fast algorithm is a vital subroutine called in the construction of Postnikov complexes. We are working on obtaining a fast algorithm for computing the higher homotopy groups using this method. This is an important open problem in algebraic topology and the results in this paper are perhaps the most important step to obtaining a fast algorithm for its solution. Specifically, to compute higher homotopy, we must quickly compute the cohomology groups of Postnikov complexes

[1]See [Mas] for a nice review of this history.

derived from sparse triangulations. It is key that this computation be fast, because it is done often: this computational bottleneck is perhaps the fundamental barrier to a practical solution. See [DC] for work in this direction. We show in section 7 how a slight tweak on our algorithm can compute the cohomology groups of a sparse triangulation in the same time bounds.

1.1 Statement of Approach and Results

This paper develops the basic concepts of sparse triangulations, and the basic algorithmic structure of the resulting sparse matrices representing the boundary homomorphisms. We develop tools that use sparseness to bound the expected growth in density and algebraic complexity. This paper concentrates on the geometric and algebraic consequences of sparseness, and the reduction of sparse homology computation to a "sparse" computational algebraic problem which is amenable to probablistic analysis. We introduce tools for analyzing the resulting "sparse" algebraic problem in a probabilistic setting. These tools yield probabilistic bounds on the growth of algebraic complexity, which is a key obstacle to obtaining fast algorithms.

1.1.1 Homology Type is Effectively Computable

One topological invariant is the *homology type* by which we mean all the homology groups of a triangulation. Henceforth, all homology and chain groups will be taken to have integer coefficients. A finite triangulation's homology groups are all finitely generated abelian groups. These groups have a structure theorem, which tells us that each can be expressed in normal form as the direct sum of a free abelian group, and zero or more several torsion groups (quotients of **Z**),

$$\mathbf{Z}^b \oplus \mathbf{Z}/d_1 \oplus \cdots \oplus \mathbf{Z}/d_l \tag{1}$$

where the each d_i divides the next, i.e., $d_1|d_2|\cdots|d_l$. Furthermore, classical algebraic topology tells us that these homology groups are effectively computable.[2] Homology computation has been considered in a complexity-theoretic setting (for regular CW-complexes and for semi-algebraic sets) by [SS], among others.

The homology computation for a simplicial complex K is effected as follows:

First, we construct the "chain groups" $\{ C_p \}$ which are free abelian groups generated by the (oriented) simplices of each dimension. A boundary homomorphism ∂_p maps from each C_p to C_{p-1}. The p^{th} homology group is defined by

$$H_p(K) = (\ker \partial_p)/(\mathrm{im}\ \partial_{p+1}). \tag{2}$$

We calculate the matrix of the boundary homomorphism in "simplex basis coordinates." This matrix is then "diagonalized" over the ring of integers; the diagonal matrix is its

[2]For a finite triangulation or regular cell complex. See textbooks such as [Mun, Spa] for a good modern treatment.

"(Smith) normal form:" The diagonal entries are $\overbrace{1, \ldots, 1}^{r\text{ 1's}}, d_1, d_2, \ldots, d_l$, $(r \geq 0)$. From this matrix we may immediately calculate the *betti number* b and *torsion coefficients* $d_1, \ldots, d_l$ that give the cannonical form of the group (1).

1.1.2 Complexity: What Is Known

Our goal here is to study how efficiently the homology type can be computed. Suppose the input triangulation K has n_p simplices in dimension p, and dim $K = D$ for some dimension bound D. Let $n = \max_p n_p$. The matrix of each boundary homomorphism has size $O(n^2)$. The computational bottleneck in determinining simplicial homology is the conversion of these matrices to (Smith) normal form.

In 1861, Smith [Smi] gave what is now considered the classical "reduction" algorithm showing that an integer matrix can be diagonalized using elementary row and column operations (see section 4 below). It was noted in the 1970's that neither the classical algorithm nor several well-known subsequent algorithms for reduction were known to be polynomial (see [KB] for a review of these observations). Kannan and Bachem then gave a polynomial-time algorithm for reduction to Smith normal form. This bound has been subsequently improved: [CC] provided an $O(s^{11})$ algorithm, where for an $n \times m$ matrix with maximum coefficient $\wp$ we define $s = m+n+\log|\wp|$. A paper by [Ili] gives the fastest worst-case bounds for computing Smith normal form, and hence the canonical structure of abelian groups. The worst-case bounds given by Iliopoulos are roughly $O(s^5)$ The algorithms in [Ili]; [CC] are different from the classical algorithm. Either of these results could be used to compute the homology type in (deterministic) times $O(Dn^{11})$ or $O(Dn^5)$.

In this paper we exploit the sparseness of the matrices and use a probabilistic analysis to derive a faster algorithm in the sparse, probabilistic case.

This paper reviews the classical algorithm for computing all the homology groups of a triangulation. While the worst-case analysis of the algorithm indicates that it could run in doubly-exponential time [Dom], we identify a class of common simplicial complexes of the kind most frequently encountered in geometric design which have a "sparseness" property that enables the algorithm to run quickly. We formalize this sparseness measure, and then perform a probabilistic analysis of the sparse case to show that the expected running time of the algorithm is roughly quadratic in the geometric complexity n (number of simplices) and linear in the dimension D of the triangulation. The bound obtained in the sparse, probabilistic case is $O(Dn^2)$.

1.1.3 Sparseness

Because diagonalizing over a ring can be expensive, the *a priori* worst-case bounds we obtain are doubly-exponential in $n = \max_p n_p$ [Dom]. However, our problem has a much better expected (probabilistic) running time because of two properties: (1) the non-zero coefficients in the boundary matrix are all initially units, and (2) we argue that many triangulations (in general) and virtually all triangulations in design are very "sparse," in a sense we now make precise. The rank n_p of C_p is the number of p-simplices in K. Hence, ∂_p may be represented by an $n_p \times n_{p-1}$ matrix A_p. Now, each p-simplex has a boundary which is a $(p-1)$-chain. For example, consider a triangular patch as an oriented 2-simplex $[v_0, v_1, v_2]$. Now, in general, for a p-simplex σ we have

$$\partial_p \sigma = \partial_p[v_0, \ldots, v_p] = \sum_{i=0}^{p}(-1)^i[v_0, \ldots, \hat{v}_i, \ldots, v_p] \tag{3}$$

where the "hat" $\hat{v}_i$ means that the vertex v_i is to be deleted to obtain a $(p-1)$-simplex. So, for our triangular patch we have

$$\partial_2[v_0, v_1, v_2] = [v_1, v_2] - [v_0, v_2] + [v_0, v_1]. \tag{4}$$

Hence, "encoding" eq. (4) into the matrix A_p requires $p+1$ entries (which are ± 1's) into one column of A_p (here, $p = 2$). These are the only non-zero entries in that column. This suggests a good definition of "sparse". Since the boundary homomorphism matrix A_p is a $n_{p-1} \times n_p$ matrix (where $n_i = \mathrm{rank}(C_i)$), we say that

Definition 2 *The boundary homomorphism ∂_p is* sparse *if the boundary homomorphism matrix A_p is sparse. A triangulation K is said to be* sparse *if all its non-trivial boundary homomorphisms are sparse.*

This definition essentially corresponds to the number of p- and $(p-1)$-simplices being much larger than p, which means that the matrix A_p will be sparse. We show that

Theorem 3 *Given a sparse triangulation K, its homology groups (with integer coefficients) can be computed in expected time $O(Dn^2)$ where D is the maximum dimension of K and $n = \max_p n_p$.*

1.1.4 Which Triangulations are Sparse?

In this section we now give algebraic and geometric criteria that characterize the sparse triangulations.

We first give an algebraic condition. We use the following intuitive notion: "the κ p-simplices that a $(p-1)$-simplex σ bounds, are called the *coboundary* of σ, and we call κ the *size* of the coboundary of σ."

The following formalization is standard and elementary. We view C_p as a Z-module and for a simplex σ and an integer α, we define $\pi : C_p \rightarrow Z$ by the rule $\pi(\alpha\sigma) = |\alpha|$. π extends to chains as follows. Each chain c can be written as a unique linear combination $\sum \alpha_i \sigma_i$ over all p-simplices σ_i (with no repetition of basis elements). We then define $\pi c = \pi\left(\sum \alpha_i \sigma_i\right) = \sum\left(\pi \alpha_i \sigma_i\right) = \sum |\alpha_i|$. We define the "dual" map $\pi^* : C^p \rightarrow Z$ analogously. For any co-chain c^*, we define its *weight* to be $\pi^* c^*$. Let δ be the coboundary operator (defined to be the dual of the boundary operator; see section 7). For any p-simplex σ, we define its *coboundary size*

κ to be the weight of $\delta\sigma^*$. The *maximum* coboundary size of any p-simplex is the maximum coboundary size over all p-simplices σ, or $\max_\sigma \pi^*\delta\sigma^*$.

Claim 4 (Algebraic Condition for Sparseness) *Let κ be the maximum coboundary size of any $(p-1)$-simplex. A boundary homomorphism ∂_p is sparse if* $\operatorname{rank}(C_{p-1}) \gg p$, *and* $\operatorname{rank}(C_p) \gg \kappa$.

Proof: Definitional. □

We can now give two geometric criteria, either of which is sufficient for sparseness. We write *vertex* for "0-simplex" and *edge* for "1-simplex" where it is mellifluous. First we give the following definition, which is standard in computational geometry:

Definition 5 *A triangulation K has* k-bounded local complexity *if there is some constant k such that for all vertices v of K, v is a face of at most k edges.*

Then

Proposition 6 (Geometric Conditions for Sparseness) *Let N be the number of vertices of a D-dimensional triangulation K. Then K is sparse if either*

1. *K has k-bounded local complexity, for some $k \ll N$, or*

2. *$D \ll N$.*

Furthermore, (1) implies (2), but the converse is false.

Proof: (1) First, assume K has k-bounded local complexity, for some $k \ll N$. We give a coboundary size bound. Let σ be a $(p-1)$-simplex, for $p \geq 1$. Consider a p-simplex ρ that contains σ as a face, and a vertex v_0 of σ. Now, ρ is formed (up to sign) by taking $\sigma = [v_0, \ldots, v_{p-1}]$ and adding a vertex v_p to obtain $\rho = [v_0, \ldots, v_{p-1}, v_p]$. So, in the absence of bounded local complexity, v_p can be any vertex, so the the coboundary size of σ could be $N - p$.

Call an edge a *v_0-edge* if it contains v_0 as a face. If there exists a bound k on the number of v_0-edges, we can enumerate the possible p-simplices ρ that σ bounds. Observe that any such ρ must contain p v_0-edges. These p v_0-edges completely determine ρ (up to sign): $p-1$ of these edges are edges of σ as well, and remaining one is of the form $\pm[v_0, v_p]$. So the number of different ways we can choose p edges from a list of k edges is $\binom{k}{p}$. Hence, in a D-dimensional triangulation, a coboundary size bound for the boundary homomorphism ∂_p is given by $\binom{k}{p}$. Since $p \leq D$, a global coboundary size bound is given by $O(k^D)$.

(1) ⇒ (2). Note we cannot have $D \geq k$ since there aren't enough v_0-edges to make any simplices bigger than dimension $k-1$. Hence, the assumption that $k \ll N$ directly implies that $D \ll N$.

(2) Now, assume $D \ll N$. We consider the densest possible triangulation K on N vertices. For example, for $D = 1$, K would simply be the complete graph on N vertices. The boundary homomorphism ∂_1 can be represented by a matrix A_1 with $\binom{N}{2}$ columns and N rows. Each column will be N-long and contain two non-zero entries. Each row will be $\binom{N}{2}$-wide and contain N non-zero entries. Hence the column sparseness is measured by the ratio $2/N$ and the row sparseness by $N/\binom{N}{2} = 2/(N-1)$. So, for large N, this "densest possible 1-triangulation" is still sparse.

Now consider the general case of the densest D-triangulation on N vertices. In general, K will contain all possible simplices on N vertices, up to D dimensions. Let $1 < p \leq D$. A boundary homomorphism ∂_{p-1} can be represented by a matrix A_{p-1} with $\binom{N}{p}$ columns and $\binom{N}{p-1}$ rows. Each column will be $\binom{N}{p-1}$-long and contain p non-zero entries. Each row will be $\binom{N}{p}$-wide and contain $\binom{N}{p-1}$ non-zero entries. Hence the column sparseness is measured by the ratio

$$\frac{p}{\binom{N}{p-1}} = \frac{p!(N-p+1)!}{N!} = \frac{p!}{N(N-1)(N-1)\cdots(N-p+2)} = \frac{p!}{O(N^p)}$$

and the row sparseness by

$$\frac{\binom{N}{p-1}}{\binom{N}{p}} = \frac{p}{N-p+1},$$

where $p \leq D$. So, for $D \ll N$, these ratios are very small and hence even the "densest possible D-triangulation" is still sparse.

(2) does not imply (1). For example, consider the 1-triangulation for the complete graph on N vertices. The coboundary size of each vertex is $N-1$, but the triangulation is still not dense when N is large. □

Now, obviously, there are triangulations that violate sparseness. For example, a triangulation consisting solely of one 1000-simplex will not be sparse for $p \approx 1000$. More generally, if K consists solely of a "small number" of p-simplices, it is not sparse. Now, if p is small, the diagonalization algorithm is still fast (eg., our implementation is fast for $p < 30$ even for non sparse matrices). For larger p, the algebraic complexity will interact with the time bound. Hence "non-sparseness", as defined above, captures the notion of when K consists of a "small number" of p-simplices: this occurs when p approaches (or exceeds) $\text{rank}(C_{p-1})$.

However, in most triangulations, and virtually all geometric designs, the number of p-simplices is much greater than p. For example, even the standard triangulation of a 2-torus (or klein bottle) requires 18 2-simplices, 27 1-simplices, and 9 vertices. (There is a difficult but well-known theorem showing that this triangulation is, in fact, minimal). Physically manufacturable designs will have $D = 3$. Almost any "real" geometric design arising in applications will have hundreds of 2- and 3-simplices, and the ratio $\left(p/\text{rank}(C_{p-1})\right)$ will be very small indeed. Even configuration spaces for motion planning will be "sparse" in our sense. We were initially motivated to develop this probabilistic analysis for the sparse case because our implementation of this algorithm ran very fast in practice, and we wished to explain why the worst-case behavior was not attained. We wanted (1) to establish a rigorous account of this much better average-case behavior, and (2) to classify the triangulations on which it could be expected. (1) led to our probablistic analysis for the sparse case. (2) led to our definition of sparseness: on "sparse" triangulations, the quadratic time behavior can be expected. Of course, the general problem of computing the homology of non-sparse triangulations is still of theoretical interest, as is the problem of diagonalizing matrices over an arbitrary euclidean domain. While it is clear that one can make the diagonalization algorithm run slowly by bad choice of non-sparse examples; however, it is also clear that probabilistic analysis of the sparse case better predicts the expected complexity of homology calculation.

1.1.5 Applications

One of our interests is the efficient computation of the higher homotopy groups of simply-connected spaces (such as n-spheres). That computation is critically dependent on the results in this paper, which is called often as a subroutine. See [DC] for work in this direction.

However, there are many other applications as well. In geometric design (CAD) we could take as input two triangulations and compute whether they have the same homology groups. While isomorphism of all the homology groups does not imply homeomorphism, it does mean that their "topologies" are "similar." For example, the triangulations could represent solid finite-element meshes in mechanical design, or curved surfaces (eg, Bézier systems). We know that surfaces with torsion homology are generally not embeddable in $\Re^3$. For example, for the Klein bottle $\mathcal{K}$ and the projective plane P^2 we have $H_1(\mathcal{K}) = Z \oplus Z/2$ and $H_1(P^2) = Z/2$, resp. Hence the presence of torsion implies the physical unrealizability of a design. Thus, for example, we could compute whether a series of modifications to a design preserves its embeddability, connectivity, number of holes, etc. There is a wealth of information in the homology groups of a topological space K. In geometric design, the Euler number $\chi(K)$ has been suggested as a topological invariant. The *Euler Number* of K is computable from the betti numbers in eq (1). Let β_p be the betti number in dimension p. Then $\chi(K) = \sum_p (-1)^p \beta_p$. Hence it is clear how the homology type provides strictly more information than the euler characteristic. Because the homology groups carry so much information about the space, we could indeed imagine a CAD system where the homology type of the design was part of the design specification, or where designs were classified by homology type.

1.2 Review of Previous Work

The computability of the homology groups of a simplicial complex is well known basic mathematics; see textbooks on elementary algebraic topology, for example, [Mun; Spa]. Worst-case deterministic bounds have been given by [Smi; KB; CC; Ili]; see sec. 1.1.2. For work on computing cohomology groups, see [Hol; Gla; Lam]. For other work on computational algebraic topology, see [Bro; Ani; SS; McC; Hir; Col; Arn; KY; Can; CD]. See also [Whi]. [VY; Sno] discuss related problems of interest.

2 Mathematical Preliminaries

2.1 Review of Simplicial Complexes

We assume the reader has the intuition that a triangulation or tetrahedralization of a solid is a geometric realization of a simplicial complex. This object contains 0-simplices (vertices), 1-simplices (edges), 2-simplices (triangles), 3-simplices (tetrahedra), etc. The set of these is called a *simplicial complex.* The intersection of any two simplices must be another simplex of lower dimension. Clearly, we can represent a p-simplex by its vertices $[v_0, \ldots, v_p]$. We wish to treat a simplicial complex as a purely combinatorial object. We do this by treating the simplices of a complex as combinatorial objects "independent of their coordinates." We define

Definition 7 *An* abstract simplicial complex K *consists of a set* $\{ v \}$ *of vertices and a set* $\{ s \}$ *of finite non-empty subsets of* $\{ v \}$ *called simplexes such that*

1. *Any set consisting of exactly one vertex is a simplex.*
2. *Any non-empty subset of a simplex is a simplex.*

$\sigma \in K$ is called a *simplex* of K, and its *dimension* is one less than its cardinality. The *vertices* of K have dimension 0. The dimension of K is the dimension of the largest simplex in K.

A geometric realization of σ is an embedding $\phi : \sigma \rightarrow \Re^n$ such that the image of the vertices of σ are linearly independent. A geometric realization of K is an embedding of each simplex that preserves the adjacency relationships of K: that is, if σ, τ, ρ are simplices and $\sigma \cap \tau = \rho$, then $\phi(\sigma) \cap \phi(\tau) = \phi(\rho)$.

Next we define the orientation of a simplex:

Definition 8 *Let σ be a p-simplex. Define two orderings of its vertices of be equivalent if they differ by an even permutation. This defines two equivalence classes of vertex orderings for $p > 0$. (Vertices have only one class, hence one orientation). Each of these classes is called an* orientation *of σ. An* oriented simplex *is a simplex σ together with an orientation ± 1 of σ.*

Let $v_0, \ldots, v_p$ be linearly independent points. We shall denote by $[v_0, \ldots, v_p]$ the oriented simplex consisting of the simplex with vertices $v_0, \ldots, v_p$ and the equivalence class of the particular ordering $v_0, \ldots, v_p$.

2.2 Review of Simplicial Homology

We now review some well known mathematical background. We first define the *chain groups* for a simplicial complex K.

Definition 9 *A* p-chain *on a simplicial complex K is a function c from the oriented p-simplices of K to the integers that vanishes on all but finitely many p-simplices, such that*

$$c(\sigma) = -c(\sigma')$$

if σ and σ' are opposite orientations of the same simplex.

Definition 10 *We add p-chains by adding their values. This yields a group $C_p(K)$, called the* chain group of dimension p. *For $p < 0$ or $p >$* dim K, *$C_p(K)$ is the trivial group.*

For every oriented simplex σ, we define the *elementary chain c_σ* as follows: $c_\sigma(\sigma) = 1$, $c_\sigma(\sigma') = -1$ (if σ' is the opposite orientation of σ), and c_σ vanishes on all other oriented simplices. Now, it is convenient to write σ to denote not only an oriented simplex σ, but also to denote the elementary p-chain c_σ. Hence we write $\sigma' = -\sigma$.

We note the following fact: $C_p(K)$ is the direct sum of subgroups isomorphic to Z, one for each p-simplex of K. Hence, $C_p(K)$ is a free abelian group generated by the oriented p-simplices of K. To "compute" $C_p(K)$, we simply count the number n_p of p-simplices, and "construct" $C_p(K)$ as Z^{n_p}. This means we view the oriented p-simplices as a basis for $C_p(K)$, and we can write any chain in the group as a finite linear combination $\sum k_i\sigma_i$. One example of such a chain is the RHS of eq. (4).

Next, we define the boundary homomorphism $\partial_p : C_p(K) \rightarrow C_{p-1}(K)$ using equation (3). An example of computing ∂_2 is shown in eq. (4). This constructs a *free chain complex* (a differential graded group of degree -1, in which each C_p is free abelian):

$$C_p(K) \xrightarrow{\partial_p} C_{p-1}(K) \xrightarrow{\partial_{p-1}} \cdots \xrightarrow{\partial_2} C_1(K) \xrightarrow{\partial_1} C_0(K). \tag{5}$$

It is straightforward to show that for all p

$$\partial_p \circ \partial_{p-1} = 0. \tag{6}$$

Finally, we can define the homology groups, formalizing (2):

Definition 11 *The kernel of $\partial_p : C_p(K) \rightarrow C_{p-1}(K)$ is called the group of* p-cycles *and is denoted $Z_p(K)$. The image of $\partial_{p+1} : C_{p+1}(K) \rightarrow C_p(K)$ is called the group of* p-boundaries, *and is denoted $B_p(K)$. By (6), each boundary of a $(p+1)$-chain is a p-cycle, so $B_p(K) \subset Z_p(K)$. We define*

$$H_p(K) = Z_p(K)/B_p(K) \tag{7}$$

to be the p-th homology group *of K.*

2.3 Computing the Homology Groups

Now, let G and H be free abelian groups with bases $\sigma_1, \ldots, \sigma_n$ and $\rho_1, \ldots, \rho_m$ respectively. For a homomorphism $f : G \to H$, then

$$f(\sigma_j) = \sum_{i=1}^{j} a_{ij}\rho_i \tag{8}$$

for unique integers a_{ij}. The matrix $A = \left(a_{ij}\right)$ is called the *matrix* of f relative to the chosen bases for G and H.

We recall the following basic theorem from algebra:

Theorem 12 *Let $A = \left(a_{ij}\right)$ be an $n \times m$ matrix over a principle ideal domain R. Then A can be "diagonalized" in the sense that we can obtain a diagonal matrix*

$$PAQ \tag{9}$$

where $P \in GL_m(R)$ and $Q \in GL_n(R)$. If the diagonal elements of (9) are $d_1, d_2, \ldots, d_l$, then $d_1|d_2|\cdots|d_l$ Furthermore, if R is a euclidean domain, then (9) may be obtained by elementary row and column operations on A.

This "diagonal" form is called the "normal form" of the matrix; the algorithm to compute it is called the "reduction algorithm." Z, of course, is a euclidean domain. We further recall that the elementary row operations are as follows:

1. Add an integer multiple of row i to row j $(i \neq j)$.
2. Interchange rows i and j.
3. Multiple row i by a unit (in our case, ± 1).

Each of these corresponds to a change of basis in H. There are three similar "column" operations that correspond to a change of basis in G. Theorem 12 therefore states that we can always apply a sequence of these six operations on A to reduce it to the desired "normal form." By viewing the presentation matrices for abelian groups as Z-modules, we obtain the structure theorem for finitely-generated abelian groups (1). The reduction algorithm is central to computing the normal form of the boundary homomorphism, and hence, to computing the homology groups. We review the well-known reduction algorithm in section 4. We proceed now to describe the following classical construction from algebraic topology; see, for example, [Mun].

1. Let $Z_p = \ker \partial_p$, and $B_p = \operatorname{im} \partial_{p+1}$.

2. Let W_p be all elements σ_p of C_p such that some non-zero multiple of σ_p belongs to B_p. W_p is a subgroup of C_p, called the *group of weak p-boundaries* [Mun]. We have

$$B_p \subset W_p \subset Z_p \subset C_p.$$

3. Using the reduction algorithm, we diagonalize the matrix for ∂_p, that is, we choose bases $\sigma_1, \ldots, \sigma_n$ for C_p and $\rho_1, \ldots, \rho_m$ for C_{p-1}, relative to which the matrix of ∂_p has the normal form

$$\begin{array}{c} \\ \rho_1 \\ \vdots \\ \rho_l \\ \rho_{l+1} \\ \vdots \\ \rho_m \end{array} \begin{array}{c} \begin{array}{ccccccc} \sigma_1 & \cdots & \sigma_l & \sigma_{l+1} & \cdots & \sigma_n \end{array} \\ \left[\begin{array}{ccc|ccc} d_1 & & 0 & & & \\ & \ddots & & & 0 & \\ 0 & & d_l & & & \\ \hline & & & & & \\ & 0 & & & 0 & \\ & & & & & \end{array}\right] \end{array} \tag{10}$$

where all the d_i are positive, and $d_1 | d_2 | \cdots | d_l$. It is not hard to show that

(a) $\sigma_{l+1}, \ldots, \sigma_n$ is a basis for Z_p.

(b) $\rho_1, \ldots, \rho_l$ is a basis for W_{p-1}.

(c) $d_1\rho_1, \ldots, d_l\rho_l$ is a basis for B_{p-1}.

A straightforward computation shows that

$$H_p(K) \cong (Z_p/W_p) \oplus (W_p/B_p). \tag{11}$$

Now, the group Z_p/W_p is free, and W_p/B_p is a torsion group. Hence, we can calculate $H_p(K)$ by computing these two groups.

2.3.1 The Algorithm For Homology Group Computation

We proceed as follows. First, we choose bases for the chain groups by arbitrarily orienting the simplices of K. These orientations remain fixed for the computation. Then we construct the matrix of the boundary homomorphism relative to these bases. Its entries will all be 0, 1, or -1. We next reduce this matrix to normal form using the reduction algorithm, obtaining a matrix like (10). Now,

1. The rank of Z_p is the number of zero columns in matrix (10).

2. The rank of W_{p-1} is the number of non-zero rows in (10).

3. There is an isomorphism

$$W_{p-1}/B_{p-1} \cong Z/d_1 \oplus \cdots \oplus Z/d_l.$$

Hence, from the normal form of the boundary homomorphism (10) we can "read off" the torsion coefficients of K in dimension $p-1$; they are simply the entries of the matrix that are greater than 1. We can also "read off" the rank of Z_p and W_{p-1} (see above). Finally, the normal form for ∂_{p+1} gives us the rank of W_p. The betti number of K in dimension p is simply $b = \text{rank}(Z_p/W_p) = \text{rank}(Z_p) - \text{rank}(W_p)$.

3 Complexity: New Results

All of the steps in the algorithm given in sec. 2.3.1 are essentially linear-time operations except the reduction algorithm. The time $R(n)$ required by the reduction algorithm (section 4) in fact dominates the computation of homology type, which can clearly be done in time $O(D(n+R(n)))$. We now examine the reduction algorithm and analyze its complexity. While worst-case bounds for the reduction algorithm are poor, we describe a sparse probabilistic analysis which yields an expected time bound that is roughly quadratic.

3.1 Exactly What We Show

An (s,t)-*sparse* matrix has s non-zero entries per column and t per row. (s,t)-sparse integer boundary matrices arise in the computation of integral homology. We will give a probabilistic analysis for diagonalizing an integer (s,t)-sparse matrix into normal formal. By *normal form* of a matrix, we mean the diagonalization of the matrix over the ring of integers. We prove that under high probability the expected running time is $O(n^2)$ where n is the size of the given (s,t)-sparse matrix, i.e. this expected running time can be achieved with probability very close to 1 when $s,t \ll n$.

3.2 Definitions

We now study the classical reduction algorithm (see, e.g. [Mun]) and show that this reduction algorithm runs fast (i.e. $O(n^2)$) probabilistically on a (s,t)-*sparse* matrix with non-zero entries uniformly distributed over the set $\mathbf{Z}_{[-\wp,\wp]}\backslash\{0\}$, where $\mathbf{Z}_{[-\wp,\wp]}$ is the set of integers in the interval $[-\wp,\wp]$, $\wp$ is a very small positive integer and n is the *size* of the matrix. In the case of integral homology computation, $\wp = 1$.

Definition 13 *We denote the* algebraic complexity *of a matrix* $\mathbf{A}$ *by* $\text{alg}(\mathbf{A})$. $\text{alg}(\mathbf{A}) = \max\{|a_{ij}| \mid a_{ij}$ *is an entry in* $\mathbf{A}\}$.

Definition 14 *Henceforth we will consider diagonalization of an integer matrix* $\mathbf{A}$ *over* $\mathbf{Z}$. *We will assume* $\mathbf{A}$ *has n rows and m columns, and without loss of generality we take* $n \geq m$. *We call n the* size *of the matrix.*

Definition 15 *An $n \times m$ matrix $\mathbf{M}$ is called (s,t)-*sparse *if each row (resp. column) of $\mathbf{M}$ has exactly t (resp. s) non-zero elements and $s \ll n$ and $t \ll m$. Furthermore, we require that the non-zero entries in $\mathbf{M}$ are uniformly distributed over the set $\mathbf{Z}_{[-\wp,\wp]}\backslash\{0\}$ where $\wp$ is a very small positive integer, and each entry of this matrix has equal chance of being non-zero. For convenience, we define $\alpha = \log_2 n$, $\beta = \log_2 m$ and assume $n \geq m$ wlog.*

3.3 How We Prove It: the Basic Idea

In this section, we give a brief overview of our approach in order to give the reader the general idea; a formal and careful exposition comes in section 5.

We proceed as follows. The approach we take in analyzing the probabilistic complexity of normal form computation is to examine *pre-reduction* complexity. This operation reduces the computation on an $n \times m$ matrix to the computation on an $(n-1) \times (m-1)$ matrix.

Definition 16 *When an arbitrary matrix is in the form of eq. (1), where $a_{1,1}$ divides each element of $\mathbf{B}$, we say the matrix is in* pre-reduced *form. We call the algorithmic process of bringing a matrix into pre-reduced form* pre-reduction. *We call the matrix $\mathbf{B}$ in eq. (1) the* remaining matrix *after a pre-reduction.*

$$\left(\begin{array}{c|c} a_{1,1} & 0 \cdots 0 \\ \hline 0 & \\ \vdots & \mathbf{B} \\ 0 & \end{array}\right) \tag{12}$$

Examining the steps of the reduction algorithm (reviewed in section 4), and keeping track of complexity, we note the following simple

Proposition 17 *Let ℓ_1 be the smallest non-zero element of the initial matrix A. Then pre-reducing the matrix can be done in time $O(\ell_1 nm)$. If A is sparse, then pre-reduction can be done in time $O(\ell_1 n)$.* $\square$

By a further examination of the reduction algorithm, it is evident that the change in algebraic complexity of an entry after a pre-reduction depends only on the algebraic complexity before the pre-reduction. Suppose we begin with an (s,t)-sparse $n \times m$ matrix $\mathbf{A}$. We first notice that our original (s,t)-sparse matrix gets denser after each pre-reduction but the algebraic complexity remains the same (probabilistically) for the first few pre-reductions. That is, initially, as the matrix is diagonalized, it remains "sparse enough" that a subsequent pre-reduction will increase its density, but not its algebraic complexity. However, the remaining matrix becomes increasingly dense, and this "*sparse enough*" property is eventually violated. At this point, we call the remaining matrix "*dense*". We show that the dense remaining matrix we obtain has uniformly distributed entries all of low algebraic complexity. From our point of view, the difference between "sparse enough" and "dense" is as follows:

Definition 18 "Sparse enough" *matrices have non-zero elements that are uniformly distributed and of low algebraic complexity. A pre-reduction of a* "sparse enough" *matrix will make it somewhat denser, but will not raise its expected algebraic complexity.*

Definition 19 *The entries of a* "dense" *matrix are uniformly distributed and of low algebraic complexity. A pre-reduction of a* "dense" *matrix is expected to raise its algebraic complexity.*

We show that the expected number of pre-reductions we can perform before obtaining a dense remaining matrix $\mathbf{B}$ is at least $n - \sqrt[4]{n}$, and hence the resulting dense remaining matrix $\mathbf{B}$ is of expected size at most $\sqrt[4]{n}$ (recall the definition of size, def. 14). Furthermore, the entries in this dense remaining matrix $\mathbf{B}$ of size at most $\sqrt[4]{n}$ are uniformly distributed over the integral interval $\mathbf{Z}_{[-\wp,\wp]}$ with $\wp$ a very small positive integer. From prop. 17, we know that each pre-reduction of a "sparse enough" matrix of size n takes time $O(n)$, and hence we can obtain our dense remaining matrix $\mathbf{B}$ in time $O(n^2)$.

It then remains to analyze the complexity of diagonalizing the remaining dense matrix $\mathbf{B}$. First, we will prove that the algebraic complexity of $\mathbf{B}$ changes by a constant amount after each successive pre-reduction. We show that for the case of integral homology computation the expected value of this constant is in fact 1. Pre-reducing a general dense $n \times m$ matrix of uniform algebraic complexity $\wp$ can be done in time $O(\wp mn)$ (prop. 17). Let $r = \sqrt[4]{n}$. Hence we can diagonalize an $r \times r$ dense remaining matrix $\mathbf{B}$ with initial algebraic complexity $\wp = 1$ in expected time

$$r^2 + 2(r-1)^2 + 3(r-2)^2 + \cdots = \sum_{i=1}^{r} i(r-i+1)^2 = O(r^4). \tag{13}$$

Clearly, $O(r^4) = O(n)$. Hence $\mathbf{B}$ can be diagonalized in linear ($O(n)$) expected time. We conclude that the complexity of normal form computation for a (s,t)-sparse matrix $\mathbf{A}$ is $O(n^2)$ probabilistically where n is the size of $\mathbf{A}$, i.e. This expected running time can be achieved with probability very close to 1 when $s, t \ll n$.

In this paper, we analyze our algorithm in the real RAM arithmetic-complexity model. However, We also notice that the diagonalization of an $r \times r$ dense remaining matrix $\mathbf{B}$ can be done in time $O(n\log^2 n)$ in the bit-complexity model. More generally, let $\mathrm{m}(s)$ be the time required to multiply 2 integers of size $\leq s$. Then we rewrite eq. (13) and instead of eq. (13) we obtain:

$$\begin{aligned} & \sum_{i=1}^{r} \mathrm{m}(i) i(r-i+1)^2 \\ \leq & \sum_{i=1}^{r} \mathrm{m}(r) i(r-i+1)^2 \\ = & \ \mathrm{m}(r) O(r^4) \\ = & \ O(\mathrm{m}(\sqrt[4]{n})n) \end{aligned}$$

So even if $\mathrm{m}(s) = \log^2 s$, we have $O(\mathrm{m}(\sqrt[4]{n})n) = O(n\log^2 n)$. Since until the dense remaining matrix $\mathbf{B}$ is obtained, the expected algebraic complexity is very low, we can therefore conclude that the bit-complexity of normal form computation for a (s,t)-sparse matrix $\mathbf{A}$ is also $O(n^2)$ probabilistically where n is the size of $\mathbf{A}$, i.e. This expected running time can be achieved with probability very close to 1 when $s, t \ll n$.

In order to prove this result, we describe a number of tools. First we develop a technique for analyzing the combinatorics of diagonalization, by gathering successive pre-reduction steps together into "groups" called *groups of pre-reductions.* By *group pre-reduction* we mean a sequence of pre-reductions having the property that each pre-reduction in the sequence increases the number of non-zero entries in a row by the same expected amount. We then gather the group pre-reductions into sequences called "*phases*". By *a phase of group pre-reductions* we mean a sequence of at most $\sqrt{2\alpha}$ successive group pre-reductions, where $\alpha = \log_2 n$. Phases of group pre-reductions can be combinatorially cascaded in order to effect a complexity analysis. We use discrete random variables to model integral matrix entries, and thereby determine bounds on their expected growth and density. The probabilistic analysis is complicated by the destruction of uniformness and independence of the matrix entries after the dense matrix is obtained. However, we are able to show that the entries are nevertheless *conditionally independent* and *uniform* (on the *outer row* and *column*). This admits an inductive probabilistic argument (based on the recursive conditioning) that we use to derive our theorem on the constant algebraic complexity growth per pre-reduction in a dense matrix with low algebraic complexity.

In section 6, we generalize our probabilistic analysis to a non-uniform and dependent $n \times m$ (s,t)-sparse matrix, that is a (s,t)-sparse matrix whose entries have dependent and non-uniform probability distributions on being non-zero. We propose a pre-processing algorithm using *active randomization* to destroy the non-uniformness and dependence in order to obtain a uniform and independent (s,t)-sparse matrix, that is a (s,t)-sparse matrix whose entries have independent and uniform probability distributions on being non-zero. Then we can use the reduction algorithm to diagonalize this uniform and independent (s,t)-sparse matrix. The randomization corresponds to a random "change of basis", hence the Smith normal form will be the same. We show that the active randomization algorithm takes time $O(n+m)$. Hence, we prove that diagonalizing a non-uniform and dependent $n \times m$ (s,t)-sparse matrix can be done in expected time $O(n^2)$ with probability very close to 1 as n is large. Therefore, diagonalizing a (s,t)-sparse matrix takes expected time $O(n^2)$.

4 The Reduction Algorithm

We now prove theorem 12 by giving the classical reduction algorithm. In the process we keep track of the complexity. We observe first that all elementary row (resp. column) operations take time $O(m)$ (resp. $O(n)$). The proof uses the division algorithm (i.e., the fact that Z is a euclidean domain).

Proof (of therorem 12): Let $A = \left(a_{ij}\right)$ be an $m \times n$ matrix over Z.

Step 1 (preprocessing): We find the smallest non-zero element, ℓ_1 of A. (Here *smallest* means smallest "size" in the ring—i.e., smallest absolute value). Permute the rows and columns to move the smallest element to the upper left hand corner, a_{11}. Multiply the first row by ± 1 to make a_{11} positive.
Step 2: Suppose a_{11} doesn't divide some entry in the first column, say, $a_{11} \not| a_{i1}$:

$$\begin{pmatrix} a_{11} & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ a_{i1} & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix} \tag{14}$$

Use the division algorithm to obtain

$$a_{i1} = qa_{11} + r \tag{15}$$

where $q, r \in Z$ and $0 < r < a_{11}$. Now we add $(-q)$ times the first row to the i^{th} row. This replaces a_{i1} by r, while increasing the "size" of the numbers in row i by a factor of q. Since $r < a_{11}$, a_{11} is no longer the minimal element of A. The new minimal element is somewhere in row i (it may not be r). Repermute the matrix so that the new minimal element is in the upper left corner.
Step 3: Repeat step 2 until a_{11} divides every a_{i1} in the first column.
Step 4: Similarly, we may use column operations to reduce to the case where a_{11} divides every a_{1j} in the first row.

We observe that step 1 takes time $O(mn)$, step 2 takes time $O(m+n)$. We observe that steps 3 and 4 terminate because at each step, a_{11} is decremented by at least one. Hence, we reach step 5 after $O(mn + \ell_1(n+m))$ time.
Step 5: Subtract off multiples of the first row and column to make all the elements a_{i1} in the first column and a_{1j} in the first row zero, for $i, j \neq 1$. Now the matrix looks like this:

$$\begin{pmatrix} a_{11} & 0 \cdot \cdot \cdot 0 \\ 0 & \\ \cdot & \\ \cdot & \boxed{\quad B \quad} \\ \cdot & \\ \cdot & \\ 0 & \end{pmatrix} \tag{16}$$

Now, we wish to ensure that a_{11} divides every element of the submatrix B. We number B's rows and columns starting at 2.
Step 6: Suppose some entry b_{ij} of B is not divisible by a_{11}. We add row i to row 1 to obtain:

$$\begin{pmatrix} a_{11} & b_{i2} & \cdots & b_{ij} & \cdots & b_{in} \\ 0 & \multicolumn{5}{c}{} \\ \cdot & & & & & \\ \cdot & & & \boxed{\mathrm{B}} & & \\ \cdot & & & & & \\ \cdot & & & & & \\ 0 & & & & & \end{pmatrix}. \tag{17}$$

Next, we apply the division algorithm to obtain

$$b_{ij} = qa_{11} + r \tag{18}$$

with $q, r \in Z$, $0 < r < a_{11}$. We subtract q times column 1 from column j to obtain:

$$\begin{pmatrix} a_{11} & b_{i2} & \cdots & r & \cdots & b_{in} \\ 0 & & & & & \\ \cdot & & & & & \\ \cdot & & & \boxed{\mathrm{B}} & & \\ \cdot & & & & & \\ \cdot & & & & & \\ 0 & & & & & \end{pmatrix}. \tag{19}$$

Step 7: Now, at this point, a_{11} is no longer the minimal element of A, since $r < a_{11}$. We go back to step 1, moving r to a_{11}. Since the corner element is decreased by at least one each time, the (entire) process terminates after at most ℓ_1 iterations. Finally, a_{11} will divide every entry in B, and the first row and column will be zero, except for a_{11}. Hence, the matrix has the form (16), and a_{11} divides each element of B.
Step 8 (recursive step): We note that if a is an integer that divides each entry of A, and that if A' is obtained from A by any row or column operation, then a still divides each entry of A'. Hence, we can recursively invoke the algorithm on submatrix B. □

5 Normal Form Computation of a (s,t)-sparse Matrix

Consider a pre-reduction computation. For the given matrix $\mathbf{A} = \left(a_{i,j} \right)$, we can always find an element a with the smallest size and transform it to the $(1,1)$-position. Next, we can use elementary row operations to ensure that this element will divide all the entries in the first column. We can then subtract off multiples of rows to zero out the entries in the first column (except for $a_{1,1}$). This process halts after most $|a|$ row operations (as can be seen from the Euclidean division algorithm). The entries in the first row can be zeroed out similarly. Recall prop. 17. Without loss of generality, will assume $a_{1,1} = 1$, because this case can maximize the possible worst-case increase in algebraic complexity. Now, let's consider the following step in a pre-reduction.

$$\begin{pmatrix} 1 & \dots & b & \dots \\ \dots & \dots & \dots & \dots \\ q & \dots & a & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \mapsto \begin{pmatrix} 1 & \dots & b & \dots \\ \dots & \dots & \dots & \dots \\ 0 & \dots & a-bq & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \mapsto \begin{pmatrix} 1 & \dots & 0 & \dots \\ \dots & \dots & \dots & \dots \\ 0 & \dots & a-bq & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \tag{20}$$

Definition 20 *We call the step illustrated in eq. (20) a* basic step *in a pre-reduction.*

Definition 21 *We call the first row of the matrix* $\mathbf{A}$ *the* outer row, *and the row containing* a *and* q hit row. *The* outer *and* hit columns *are defined analogously.*

We are interested in the following two questions for a (s,t)-sparse matrix $\mathbf{A}$:

Question 22 *Will a (s,t)-sparse matrix get denser after a sequence of pre-reductions? If so, how fast?*

Question 23 *Will the* algebraic complexity alg($\mathbf{A}$) *grow after a sequence of pre-reductions? If so, how fast?*

5.1 Change of Density

We notice that a row can get denser after performing a basic step if and only if some zero entries are converted to non-zero. In the following, we will define a random variable X to measure the density growth after one basic step.

5.1.1 Discrete Random Variables, Expected Values and Variance of a Basic Step

For a (s,t)-sparse matrix $\mathbf{A}$, the discrete random variable X is defined as a map from the set of indices of non-zero entries in the outer row (except the first entry on that row), to the set $\{0,1,...,t-1\}$(i.e. the set of possible number of zero entries being converted to non-zero in a basic step). We know that the probability of a entry being non-zero in a (s,t)-sparse matrix $\mathbf{A}$ is $P(a_{i,j} \neq 0) = 1-(1-\frac{s}{n})(1-\frac{t}{m}) = \frac{s}{n}+\frac{t}{m}-\frac{st}{mn}$. For convenience, we define p to be $\frac{s}{n}+\frac{t}{m}-\frac{st}{mn}$. Clearly, $p \to 0$ as $\frac{s}{n}, \frac{t}{m} \to 0$ asymptotically. So we have

Definition 24 *The probability distribution function f is defined as follows,*

$$f(z) = P(X = z) = \binom{t-1}{z}(1-p)^z(p)^{t-z-1} \tag{21}$$

f here is usually called the binomial distribution function.

We know that the expectation $E(X)$ and variance $V(X)$ can be computed easily as follows,

$$\begin{aligned} E(X) &= (t-1)(1-p) && (22)\\ &= (t-1)(1-\frac{s}{n}+\frac{t}{m}-\frac{st}{mn}) \rightarrow t-1 \textit{ as } p \rightarrow 0 && (23) \end{aligned}$$

$$\begin{aligned} V(X) &= (t-1)p(1-p) && (24)\\ &< (t-1)p && (25)\\ &= \frac{t(t-1)}{n}+\frac{s(t-1)}{m}-\frac{st(t-1)}{mn} \rightarrow 0 \textit{ as } p \rightarrow 0 && (26) \end{aligned}$$

$E(X)$ and $V(X)$ tell us that, $t-1$ zero entries are converted to non-zero in a basic step of a pre-reduction with probability very close to 1 as $p \rightarrow 0$, i.e., the hit row now is expected to have $2t-2$ non-zero entries and no non-zero entries on the hit row are expected to be hit in a basic step. So, we can also conclude that, the algebraic complexity stays unchanged with probability very close to 1 as $p \rightarrow 0$. Since in a pre-reduction there are exactly s rows involved, so we have

Proposition 25 *After n/s pre-reductions, we obtain an $2^\alpha(1-\frac{1}{s}) \times (2^\beta - \frac{2^\alpha}{s})$ remaining matrix **B** of expected sparsity (2s-2, 2t-2) with probability very close to 1 as $\frac{s}{n}, \frac{t}{m} \rightarrow 0$ asymptotically where $n = 2^\alpha, m = 2^\beta, n > m$.*

Proof: Let us consider the remaining matrix **B** after the first pre-reduction. The matrix looks like the one in eq. (12). We know that after the first pre-reduction, some rows of **B** have $2t-2$ non-zero entries and some columns of **B** have $2s-2$ non-zero entries. In order that the outer row (resp. column) of **B** has $2t-2$ (resp. $2s-2$) non-zero entries, $a_{1,2}$ and $a_{2,1}$ have to be non-zero in the original matrix **A**. But we know that $P(a_{1,2} \neq 0) \rightarrow 0$ and $P(a_{2,1} \neq 0) \rightarrow 0$ as $\frac{s}{n}, \frac{t}{m} \rightarrow 0$. So, the outer row (resp. column) of **B** will still have t (resp. s) non-zero entries with probability very close to 1 as $\frac{s}{n}, \frac{t}{m} \rightarrow 0$. Then, the second pre-reduction will convert $t-1$ (expected number of) zero entries to non-zero in each of its hit rows. By the same analysis, we know that we can perform an expected number n/s of pre-reductions such that during each pre-reduction only $t-1$ (expected number of) zero entries are converted to non-zero in a hit row with probability very close to 1 as $\frac{s}{n}, \frac{t}{m} \rightarrow 0$. After an expected number n/s of pre-reductions, each row will have at most $2t-2$ expected non-zero entries and each column will have at most $2s-2$ expected non-zero entries. □

5.1.2 Group Pre-reductions and Phases of Group Pre-reductions

Proposition 25 tells us that during the first n/s (expected number of) pre-reductions, the expected number of zero entries converted to non-zero in a hit row stays the same with probability very close to 1 as $\frac{s}{n}, \frac{t}{m} \rightarrow 0$ asymptotically. So,

Definition 26 *We group these pre-reductions together and call them the first* group pre-reduction. *In general,* group pre-reduction *means that we group a sequence of successive pre-reductions together when this sequence of pre-reductions has the property that the expected number of zero entries converted to non-zero in a hit row stays the same after each pre-reduction in this sequence of pre-reductions.*

We know that after the first group pre-reduction, we get a denser matrix, but the probability of any entry of this matrix being non-zero is very close to 0 as the ratios $(2s-2)/(2^\alpha(1-\frac{1}{s})), (2t-2)/(2^\beta - \frac{2^\alpha}{s}) \rightarrow 0$ asymptotically where $2t-2$ (resp. $2s-2$) is the expected number of non-zero entries in each row (resp. column) and $2^\alpha(1-\frac{1}{s})$ (resp. $2^\beta - \frac{2^\alpha}{s}$) is the number of rows (resp. columns) of the remaining matrix after the first group pre-reduction. In other words, the remaining matrix is still "sparse enough" in the sense that an additional pre-reduction will not change its expected algebraic complexity (we see this from the proof of proposition 1 above). Recall that (Def. 19) a *dense* matrix B is one where a pre-reduction of B results in an expected increase in algebraic complexity. In general, we want to know how many such group pre-reductions we can perform before we reach such a dense matrix. We observe that after one group pre-reduction, the number of non-zero entries in a row is nearly doubled and $P(a_{ij} \neq 0)$, where a_{ij} is any entry in the remaining matrix, is also doubled. But as long as $P(a_{ij} \neq 0) \rightarrow 0$ we can keep performing the next group pre-reduction.

From the proof of proposition 25 and the definition of group pre-reduction, we can derive that the expected number of pre-reductions in a group pre-reduction is n'/s' where n' is the size of the matrix before the group pre-reduction and s' is the number of non-zero entries in the outer column of the matrix before the group pre-reduction. We now derive the expected number of group pre-reductions we can perform before $P(a_{ij} \neq 0) \not\rightarrow 0$. Suppose the i^{th} group pre-reduction contains p_i pre-reductions. Then the total expected number of pre-reductions we can perform before obtaining a dense remaining matrix is $\sum_i p_i$. We bound this sum below.

For convenience in our analysis, we gather successive group pre-reductions into sequences called "phases".

Definition 27 *The i^{th}* <u>phase</u> of group pre-reductions *consists of a sequence of n_i group pre-reductions with $n_i \leq \sqrt{2\alpha}$ such that after this sequence of group pre-reductions we obtain a matrix of size $2^{a\alpha}$ with $0 < a < 1$ for some a, where $n = 2^\alpha$ is the size of the original matrix.*

Also, we will assume that after each group pre-reduction, the number of non-zero entries in each row is doubled. Now, we claim the following,

Theorem 28 *For a given (s,t)-sparse matrix of size $n = 2^\alpha$: The expected number of phases of group pre-reductions is k and the expected number of group pre-reductions we can perform before obtaining a dense remaining matrix is $n_1 + n_2 + \cdots + n_k$ where $n_i \leq \sqrt{2\alpha}$ is the number of group pre-reductions in the i^{th} phase of group pre-reductions for $i = 1, 2, ..., k$ and $k = \lfloor \frac{\sqrt{\alpha}}{8} \rfloor$.*

Corollary 29 *The remaining dense matrix is of size at most $2^{\frac{\alpha}{4}} = \sqrt[4]{n}$.*

In order to prove theorem 28, we will make several observations about the size (def. 14) of the remaining matrix after each group pre-reduction. Let us consider the following sequence of values, each of which represents the size of the remaining matrix after each successive group pre-reduction. We start out with a matrix of size $n = 2^\alpha$ for some α:

$$
\begin{array}{rll}
2^\alpha & \xrightarrow{after\ first\ group\ pre-reduction} & 2^\alpha(1-\frac{1}{s}) \qquad (27)\\
 & \xrightarrow{after\ second\ group\ pre-reduction} & 2^{\alpha-1}(1-\frac{1}{s})(2-\frac{1}{s}) \qquad (28)\\
 & \xrightarrow{after\ third\ group\ pre-reduction} & 2^{\alpha-3}(1-\frac{1}{s})(2-\frac{1}{s})(4-\frac{1}{s}) \qquad (29)\\
 & \vdots & \qquad (30)\\
 & \xrightarrow{after\ i^{th}\ group\ pre-reduction} & 2^{\alpha-\gamma_i^{(1)}}(1-\frac{1}{s})(2-\frac{1}{s})(4-\frac{1}{s})\cdots(2^i-\frac{1}{s}) \qquad (31)
\end{array}
$$

In deriving the above sequence of sizes of the remaining matrix after each successive group pre-reduction, we obtained the following recurrence relation on $\gamma_i^{(1)}$:

$$
\begin{aligned}
\gamma_0^{(1)} &= 0 && (32)\\
\gamma_i^{(1)} &= \gamma_{i-1}^{(1)} + i && (33)
\end{aligned}
$$

The $\gamma_i^{(1)}$ here is a combinatorial device to help us find the transition point from the first phase of group pre-reductions to the second phase of group pre-reductions. Precisely, when $\gamma_i^{(1)}$ reaches α, we obtain a sequence of i group pre-reductions, and we will show that $i \leq \sqrt{2\alpha}$. According to the definition of a phase of group pre-reductions, we will call this sequence of i group pre-reductions the *first phase* of group pre-reductions.

Proposition 30 *In the above sequence (27) - (32), if $\gamma_i^{(1)} = \alpha$, then α and i satisfy the relation $\alpha = \frac{i(i+1)}{2}$ and the size of the remaining matrix is $(1-\frac{1}{s})(2-\frac{1}{s})\cdots(2^i-\frac{1}{s}) = 2^{a_1\alpha}$ for some real $0 < a_1 < 1$.*

Proof: From recurrence relation given by the equations (33) - (34) above, we obtain

$$
\begin{aligned}
\gamma_i^{(1)} &= \gamma_{i-1}^{(1)} + i\\
&= 1 + 2 + \cdots + i\\
&= \frac{i(i+1)}{2}
\end{aligned}
$$

So, after i^{th} group pre-reduction, the remaining matrix has $2^{a_1\alpha}$ number of rows with $0 < a_1 < 1$ for some real a_1 and

$$
(1-\frac{1}{s})(2-\frac{1}{s})\cdots(2^i-\frac{1}{s}) = 2^{a_1\alpha} \to \infty \; as \; \alpha \to \infty
$$

and the number of non-zero entries in a row (resp. column) is $2^i t$ (resp. $2^i s$). □

We call the above ((27) - (32)) sequence of group pre-reductions before $\gamma_i^{(1)}$ reaches α the first phase of group pre-reductions. For convenience, we denote n_1 to be i, i.e. n_1 is the number of group pre-reductions in the first phase of group pre-reductions. From proposition 30, we know that $n_1 \leq \sqrt{2\alpha}$. The remaining matrix obtained after the n_1^{th} group pre-reduction is of size $(1-\frac{1}{s})(2-\frac{1}{s})\cdots(2^{n_1}-\frac{1}{s})$, and the number of non-zero entries in a row (resp. column) of this remaining matrix is at most $2^{n_1}t$ (resp. $2^{n_1}s$) and the ratio

$$\frac{2^{n_1}s}{(1-\frac{1}{s})(2-\frac{1}{s})\cdots(2^{n_1}-\frac{1}{s})} \tag{34}$$

$$< \frac{1}{(1-\frac{1}{s})(2-\frac{1}{s})\cdots(2^{n_1-2}-\frac{1}{s})} \tag{35}$$

$$\rightarrow 0 \; as \; \alpha \rightarrow \infty \; and \; n_1 \approx \sqrt{2\alpha} \; by \; proposition \; 30. \tag{36}$$

For convenience, in our case we define the following,

Definition 31 *We define the* column sparseness ratio *of a matrix to be the ratio of the number of non-zero entries in a column to the size of the column. The* row sparseness ratio *is defined analogously. We will simply write* sparseness ratio *when they are the same.*

For example, eq. (34) is the sparseness ratio of the remaining matrix after the first phase of group pre-reductions.

Definition 32 *We let n' (resp. m') denote the number of rows (resp. columns) of the remaining matrix after the first phase of group pre-reductions, and t' (resp. s') the number of non-zero entries in a row (resp. column).*

The total number of non-zero entries in the remaining matrix after the first phase of group pre-reductions is $n't' = m's'$. So, the ratio $\frac{t'}{m'} \rightarrow 0$ as $\alpha \rightarrow \infty$, i.e. the probability of any entry being non-zero in the remaining matrix after the first phase of group pre-reductions is very close to 0 as $\frac{s'}{n'}, \frac{t'}{m'} \rightarrow 0$ asymptotically. Hence, we can keep performing group pre-reductions.

Now we start out with the remaining matrix of size $2^{a_1\alpha}$ and obtain the following sequence of values, each of which represents the size of the remaining matrix after each successive group pre-reduction.

$$2^{a_1\alpha} \xrightarrow{after\ 1^{st}\ group\ pre-reduction} 2^{a_1\alpha-n_1}(2^{n_1}-\frac{1}{s}) \tag{37}$$

$$\xrightarrow{after\ 2^{nd}\ group\ pre-reduction} 2^{a_1\alpha-n_1-(n_1+1)}(2^{n_1}-\frac{1}{s})(2^{n_1+1}-\frac{1}{s}) \tag{38}$$

$$\xrightarrow{after\ 3^{rd}\ group\ pre-reduction} 2^{a_1\alpha-n_1-(n_1+1)-(n_1+2)}(2^{n_1}-\frac{1}{s})(2^{n_1+1}-\frac{1}{s})(2^{n_1+2}-\frac{1}{s}) \tag{39}$$

$$\vdots \tag{40}$$

$$\xrightarrow{after\ i^{th}\ group\ pre-reduction} 2^{a_1\alpha-\gamma_i^{(2)}}(2^{n_1}-\frac{1}{s})(2^{n_1+1}-\frac{1}{s})(2^{n_1+2}-\frac{1}{s})\cdots(2^{n_1+i-1}-\frac{1}{s}) \tag{41}$$

In deriving the above sequence of sizes of the remaining matrix after each successive group pre-reduction, we obtained the following recurrence relation on $\gamma_i^{(2)}$:

$$\begin{aligned} \gamma_0^{(2)} &= n_1 && (42) \\ \gamma_i^{(2)} &= \gamma_{i-1}^{(2)} + n_1 + i - 1 && (43) \end{aligned}$$

The $\gamma_i^{(2)}$ here is a combinatorial device to help us find the transition point from the second phase of group pre-reductions to the third phase of group pre-reductions. Precisely, when $\gamma_i^{(2)}$ reaches $a_1\alpha$, we obtain a sequence of i group pre-reductions, and we will show that $i \leq \sqrt{2\alpha}$. According to the definition of a phase of group pre-reductions, we will call this sequence of i group pre-reductions the *second phase* of group pre-reductions.

Proposition 33 *In the above sequence (35) - (40), if $\gamma_i^{(2)} = a_1\alpha$, then $a_1\alpha$ and i satisfy the relation $a_1\alpha = \frac{i(i-1)}{2} + n_1 i$ where n_1 is the number of group pre-reductions in the first phase of group pre-reductions and the size of the remaining matrix is $(2^{n_1} - \frac{1}{s})(2^{n_1+1} - \frac{1}{s}) \cdots (2^{n_1+i-1} - \frac{1}{s}) = 2^{a_1 a_2 \alpha}$ for some real $0 < a_2 < 1$.*

Proof: From the recurrence relation given by the equations (41) - (42) above, we get

$$\begin{aligned} \gamma_i^{(2)} &= \gamma_{i-1}^{(2)} + n_1 + i - 1 \\ &= n_1 + (n_1 + 1) + (n_1 + 2) + \cdots + (n_1 + i - 1) \\ &= \frac{i(i-1)}{2} + n_1 i \end{aligned}$$

So, after i^{th} group pre-reduction, the remaining matrix has $2^{a_1 a_2 \alpha}$ number of rows with $0 < a_2 < 1$ for some real a_1 and

$$(2^{n_1} - \frac{1}{s})(2^{n_1+1} - \frac{1}{s}) \cdots (2^{n_1+i-1} - \frac{1}{s}) \; = \; 2^{a_1 a_2 \alpha} \to \infty \text{ as } \alpha \to \infty.$$

and the number of non-zero entries in a row (resp. column) is $2^{n_1+i}t$ (resp. $2^{n_1+i}s$). □

We call the above ((35) - (40)) sequence of group pre-reductions before $\gamma_i^{(2)}$ reaches α the second phase of group pre-reductions. For convenience, we denote n_2 to be i, i.e. n_2 is the number of group pre-reductions in the second phase of group pre-reductions. From proposition 33, we know that $n_2 \leq \sqrt{2\alpha}$. The remaining matrix obtained after the n_2^{th} group pre-reduction is of size $(2^{n_1} - \frac{1}{s})(2^{n_1+1} - \frac{1}{s}) \cdots (2^{n_1+n_2-1} - \frac{1}{s})$ and the number of non-zero entries in a row (resp. column) of this remaining matrix is at most $2^{n_1+n_2}t$ (resp. $2^{n_1+n_2}s$) and the sparseness ratio

$$\begin{aligned}
& \frac{2^{n_1+n_2}s}{(2^{n_1}-\frac{1}{s})(2^{n_1+1}-\frac{1}{s})\cdots(2^{n_1+n_2}-\frac{1}{s})} \\
< & \frac{1}{(2^{n_1}-\frac{1}{s})(2^{n_1+1}-\frac{1}{s})\cdots(2^{n_1+n_2-2}-\frac{1}{s})} \\
\rightarrow & \ 0 \textit{ as } a_1\alpha \rightarrow \infty \textit{ and } n_1 \approx \sqrt{2\alpha} \textit{ by proposition } 3.
\end{aligned}$$

For convenience, we define the following,

Definition 34 *We let n'' (resp. m'') denote the number of rows (resp. columns) of the remaining matrix after the second phase of group pre-reduction, and t'' (resp. s'') the number of non-zero entries in a row (resp. column).*

The total number of non-zero entries in the remaining matrix after the second phase of group pre-reductions is $n''t'' = m''s''$. So, the ratio $\frac{t''}{m''} \rightarrow 0$ as $\alpha \rightarrow \infty$, i.e. the probability of any entry being non-zero in the remaining matrix after the second phase of group pre-reductions is very close to 0 as $\frac{s''}{n''}, \frac{t''}{m''} \rightarrow 0$. Hence, we are ensured that we can perform at least $n_1 + n_2$ group pre-reductions.

In general, we can use this combinatorial device $\gamma_i^{(j-1)}$ to find the transition point from the $(j-1)^{th}$ phase of group pre-reductions to the j^{th} phase of group pre-reductions. After the j^{th} phase of group pre-reductions, we obtain a remaining matrix of size $\Gamma_j = 2^{a_1 a_2 \cdots a_j \alpha}$ with $0 < a_i < 1$ for $i = 1, 2, ..., j$, and each row (resp. column) of this remaining matrix has $\Upsilon_j = 2^{n_1+n_2+\cdots+n_j}t$ (resp. $\Lambda_j = 2^{n_1+n_2+\cdots+n_j}s$) non-zero entries. Since the remaining matrix becomes denser after each phase of group pre-reductions, the process of performing pre-reductions without changing the expected algebraic complexity of the remaining matrix has to stop after the k^{th} phase of group pre-reductions for some k, i.e. the remaining matrix we obtain after the k^{th} phase of group pre-reductions is dense (recall def. 19).

Proposition 35 *Before we reach a dense remaining matrix, the sparseness ratio after each phase of group pre-reductions is very small (i.e. very close to 0).*

Proof: Let us suppose that after the j^{th} phase of group pre-reductions, we obtain a dense remaining matrix, i.e., with sparseness ratio (see def. 31) $\frac{\Lambda_j}{\Gamma_j} \approx 1$. After each phase of group pre-reductions, the size of the matrix shrinks while the density grows. So, without loss of generality, we only need to consider the sparseness ratio after the $(j-1)^{th}$ phase of group pre-reductions. We have $\Lambda_{j-1} \leq \Lambda_j/2$ since $\Lambda_{j-1} = 2^{n_1+n_2+\cdots+n_{j-1}}s, \Lambda_j = 2^{n_1+n_2+\cdots+n_j}s$, and $n_j \geq 1$. Also we have $\Gamma_{j-1} = \Gamma_j^c$ since $\Gamma_j = 2^{a_1 a_2 \cdots a_{j-1}\alpha}, \Gamma_j = 2^{a_1 a_2 \cdots a_j \alpha}$, and $a_j \approx \frac{1}{c}$ for some integer $c \gg 2$. Therefore, the sparseness ratio after the $(j-1)^{th}$ phase of group pre-reductions:

$$\frac{\Lambda_{j-1}}{\Gamma_{j-1}} \leq \frac{\Lambda_j}{2\Gamma_{j-1}}$$

$$\begin{aligned} &\leq \frac{\Lambda_j}{2\Gamma_j^c} \\ &\approx \frac{1}{2\Gamma_j^{c-1}} \end{aligned}$$

is very small indeed. □

Therefore, before we reach a dense remaining matrix, each pre-reduction can be done in linear time (i.e. $O(n)$). Now, we are ready to prove theorem 28.

Proof of theorem 28:

We obtain a bound k on the number of expected phases of group pre-reductions. We know that after $n_1 + n_2 + \cdots + n_k$ of group pre-reductions, the size of the remaining matrix is $2^{a_1 a_2 \cdots a_k \alpha}$. Now, suppose $k > \frac{\sqrt{\alpha}}{8}$. We derive a contradiction. Since $\alpha \to \infty$, $\frac{\sqrt{\alpha}}{8} \to \infty$. Hence, $\prod_{i=1}^{k} a_i \to 0$ as $k \to \infty$ because $0 < a_i < 1, i = 1, 2, ..., k$. Then $2^{n_1} \geq 2^{a_1 a_2 \cdots a_k \alpha}$ because $n_1 \approx \sqrt{2\alpha}$ and lemma 1 below. Now, the number of non-zero entries after k phases is $s_k = 2^{n_1 + n_2 + \cdots + n_k}$, and the size of the remaining matrix is $N_k = 2^{a_1 a_2 \cdots a_k \alpha}$. Clearly, $s_k > N_k$ as $k \to \infty$, which is a contradiction. Thus the assumption that that $k > \frac{\sqrt{\alpha}}{8}$ is false. Hence, $k \leq \frac{\sqrt{\alpha}}{8}$. Therefore, we can perform at most $\frac{\sqrt{\alpha}}{8}$ phases of group pre-reductions, i.e. k is at most $\frac{\sqrt{\alpha}}{8}$. By proposition 31, the remaining matrix is sparse enough before we reach a dense remaining matrix. So, we can perform exactly $\lfloor \frac{\sqrt{\alpha}}{8} \rfloor$ phases of group pre-reductions. By propositions 30 & 33, we see that $n_i < \sqrt{2\alpha}, i = 1, 2, ..., k$, therefore $n_1 + n_2 + \cdots + n_k < \frac{\alpha}{4}$. Hence, the remaining dense matrix is of size at most $2^{\frac{\alpha}{4}} = \sqrt[4]{n}$. □

Lemma 36 *Let $n_1 \approx \sqrt{2\alpha}$ and $\prod_{i=1}^{k} a_i \to 0$ as $k \to \infty$ where $0 < a_i < 1$ for $i = 1, 2, ...$ Then $2^{n_1} \geq 2^{a_1 a_2 \cdots a_k \alpha}$ as $k \to \infty$.*

Proof: Since we have

$$\begin{aligned} \log_\alpha 2 &> 0 \\ \log_\alpha \prod_{i=1}^{k} a_i &\to -\infty \textit{ as } k \to \infty \end{aligned}$$

Therefore,

$$\begin{aligned} \log_\alpha \sqrt{2\alpha} &= \frac{1}{2}(\log_\alpha 2 + 1) \\ &> 1 + \log_\alpha \prod_{i=1}^{k} a_i \textit{ as } k \to \infty \\ &= \log_\alpha a_1 a_2 \cdots a_k \alpha \end{aligned}$$

Hence, $\sqrt{2\alpha} > a_1 a_2 \cdots a_k \alpha$ i.e, $2^{n_1} \geq 2^{a_1 a_2 \cdots a_k \alpha}$ as $k \to \infty$. □

is at most $\frac{\sqrt{\alpha}}{8}$ and the expected size of the remaining dense matrix is at most $2^{\frac{\alpha}{4}} = \sqrt[4]{n}$. In the next section, we consider diagonalizing this remaining matrix. We will show that the algebraic complexity is increased by a constant after each pre-reduction on a dense matrix with low algebraic complexity. In the case of integral homology group computation, the algebraic complexity is initially 1, and we show that the expected algebraic complexity in this case is actually increased by 1 after each pre-reduction. Recall proposition 17, that each pre-reduction on a dense matrix takes time $O(\ell mn)$ where ℓ the smallest (in size) non-zero entry of the dense matrix and n (resp. m) is the number of rows (resp. columns) of this dense matrix. So, the total expected running time for a dense matrix of size $r = \sqrt[4]{n}$ with initial algebraic complexity 1 is given by eq. (13) which is $O(n)$. Hence, the dense remaining matrix obtained from pre-reducing a sparse enough initial matrix of size n can be diagonalized in expected linear time. This expected running time is achieved with probability very close to 1 when n is large.

5.2 Change of Algebraic Complexity

From last section, we know that after $k \leq \frac{\sqrt{\alpha}}{8}$ phases of group pre-reductions, i.e., after $\sum_{i=1}^{k} n_i$ group pre-reductions with $n_i \leq \sqrt{2\alpha}$. We are left with a dense matrix of entries uniformly distributed over $\mathbf{Z}_{[-\wp,\wp]}$. Pre-reducing a dense matrix will change the algebraic complexity since the non-zero entries will be hit. We want to find a fast probabilistic bound on the growth of algebraic complexity. In the following, we will prove that, the algebraic complexity is increased by a expected constant per pre-reduction with probability very close to 1 when the size of the original matrix n is large.

5.2.1 Pre-reducing a Dense Matrix With Low Algebraic Complexity

In this section, we only consider a dense matrix $\mathbf{A}$ having entries uniformly distributed over $\mathbf{Z}_{[-1,1]}$. We would like to know whether after a pre-reduction, the remaining matrix has independent entries that are uniformly distributed.

Proposition 37 *After the first pre-reduction of the dense matrix* $\mathbf{A}$*, the uniformness and independence of entries in* $\mathbf{B}$ *are destroyed where* $\mathbf{B}$ *is the remaining matrix obtained after first pre-reduction.*

Proof: Let us consider eq. (20). We have over all 27 ways to choose the triple (a, b, q) from $\mathbf{Z}_{[-1,1]} \times \mathbf{Z}_{[-1,1]} \times \mathbf{Z}_{[-1,1]}$. Among these 27 triples, 9 triples will result in $|a - bq| = 0$, 14 triples will result in $|a - bq| = 1$ and 4 will result in $|a - bq| = 2$ after a basic step in a pre-reduction. We conclude that, the uniformness of entries in $\mathbf{B}$ is not preserved after a pre-reduction.

Let us now consider the following basic step in a pre-reduction:

$$\begin{pmatrix} 1 & \dots & X & \dots & Y & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ Z & \dots & U & \dots & V & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \mapsto \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & U-XZ & \dots & V-YZ & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \qquad (44)$$

where X, Y, Z, U and V are discrete random variables taking values in $\mathbf{Z}_{[-1,1]}$. In order to determine whether $U - XZ$ and $V - YZ$ are independent, we will check whether $E(U - XZ)E(V - YZ)$ is equal to $E((U - XZ)(V - YZ))$. We have

$$\begin{aligned} & E(U-XZ)E(V-YZ) \\ = \ & (E(U)-E(X)E(Z))(E(V)-E(Y)E(Z)) \\ = \ & E(U)E(V)-E(U)E(Y)E(Z)-E(V)E(X)E(Z)+E(X)E(Y)E(Z)^2 \\ & E((U-XZ)(V-YZ)) \\ = \ & E(UV-UYZ-VXZ+XYZ^2) \\ = \ & E(U)E(V)-E(U)E(Y)E(Z)-E(V)E(X)E(Z)+E(X)E(Y)E(Z^2) \end{aligned}$$

But we know that $E(Z^2) \neq E(Z)^2$, so $E(U - XZ)E(V - YZ) \neq E((U - XZ)(V - YZ))$, i.e. $U - XZ$ and $V - YZ$ are not independent. □

Proposition 37 tells us that if we only consider the probability distribution for the algebraic complexity of entries in the remaining matrix **B**, it will be very complicated to derive the probability distribution of algebraic complexity growth after in turn pre-reducing this matrix **B**, since we don't have independence and uniformness on the entries in matrix **B**. However, we notice that the change of algebraic complexity of entries in matrix **B** depends solely on the outer row and column of matrix **A**, and the entries in matrix **A** are independent and uniform. So, in the next section, we will introduce *conditional independence* and *uniformness* of entries in the remaining matrix (conditioned on the outer row and column). This essentially enables us to derive a theorem on constant growth of algebraic complexity after each pre-reduction by an inductive probabilistic argument.

5.2.2 Conditional Independence and Uniformness After Pre-reducing a Dense matrix

Definition 38 *Events A and B are called* conditionally independent on event C *if* $P(AB|C) = P(A|C)P(B|C)$.

Definition 39 *Events $A_1, A_2, ..., A_n$ are called* conditionally uniform on event C *if* $P(A_1|C) = P(A_2|C) = \cdots = P(A_n|C)$.

Since we have that entries in the outer row and column of the matrix **A** are uniformly distributed and independent, we claim the following

Proposition 40 *The entries in the remaining matrix* $\mathbf{B}$ *obtained after pre-reducing the dense matrix* $\mathbf{A}$ *are conditionally uniform and independent on the outer row and column in* $\mathbf{A}$.

Proof: Let us first prove the conditional uniformness of entries in $\mathbf{B}$. We consider the following basic step in a pre-reduction.

$$\begin{pmatrix} 1 & \dots & Y & \dots \\ \dots & \dots & \dots & \dots \\ X & \dots & Z & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \mapsto \begin{pmatrix} 1 & \dots & 0 & \dots \\ \dots & \dots & \dots & \dots \\ 0 & \dots & Z - XY & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \tag{45}$$

where X, Y and Z are discrete random variables taking values in $\mathbf{Z}_{[-1,1]}$. We want to show that the random variable $Z - XY$ is conditioned on the values of X and Y. Fix X and Y to be constants c_1 and c_2 respectively. We know that Z is uniformly distributed before the basic step is performed. Now subtracting a constant c_1c_2 from Z simply means shifting the uniform interval by c_1c_2. So, $Z - c_1c_2$ is uniformly distributed. Hence, $Z - XY$ is conditionally uniform on the outer row and column.

Now let us derive the conditional independence of entries in $\mathbf{B}$. We have three cases to consider. They are 1) entries in the same row, 2) entries in the same column, and 3) entries in different rows and columns.
Case 1) independence of entries in the same row:
We consider the following basic step in a pre-reduction.

$$\begin{pmatrix} 1 & \dots & X & \dots & Y & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ Z & \dots & X' & \dots & Y' & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \mapsto \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & X' - XZ & \dots & Y' - YZ & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \tag{46}$$

where X, X', Y, Y' and Z are discrete random variables taking values in $\mathbf{Z}_{[-1,1]}$. Again, by conditioning on the outer row and column, we fix these discrete random variables X, Y and Z to be some constants c_1, c_2 and c_3 respectively. Since X' and Y' are independent, so $X' - c_1c_3$ and $Y' - c_2c_3$ are independent. Hence, $X' - XZ$ and $Y' - YZ$ are independent conditioned on the outer row and column.
Case 2) independence of entries in the same column:
This case is symmetric to case 1), so the analysis is similar to that of case 1).
Case 3) independence of entries in different rows and columns:
We again consider the following basic step in a pre-reduction.

$$\begin{pmatrix} 1 & \dots & X & \dots & Y & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ Z & \dots & X' & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ Z' & \dots & \dots & \dots & Y' & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \mapsto \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & X' - XZ & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & Y' - YZ' & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \tag{47}$$

where X, X', Y, Y', Z and Z' are discrete random variables taking values in $\mathbf{Z}_{[-1,1]}$. Again, by conditioning on the outer row and column, we fix these discrete random variables X, Y, Z and Z' to be some constants c_1, c_2, c_3, and c_4 respectively. Since X' and Y' were independent, so $X' - c_1c_3$ and $Y' - c_2c_4$ are independent. Hence, $X' - XZ$ and $Y' - YZ'$ are independent conditioned on the outer row and column. $\square$

In general, the uniformness and independence of entries in $\mathbf{A}_i$, which is the remaining matrix obtained after i^{th} pre-reduction, is recursively conditioned on the outer row and column in $\mathbf{A}_{i-1}$ which is the remaining matrix obtained after $(i-1)^{th}$ pre-reduction. There are two cases. In the first, the dense remaining matrix is "small". In the second, it is "large". We must show that in both cases, it can be quickly diagonalized. To do this we must show that the algebraic complexity grows slowly.

Remark 41 *As we see from the sparsness property and the proof of proposition 37 above, during the first few (i.e. O(1)) pre-reductions of a dense matrix of low algebraic complexity, the algebraic complexity is increased by a expected constant after each successive pre-reduction. So, for a constant size (i.e. $O(1)$) dense matrix, we can diagonalize this matrix in constant time $O(1)$. On the other hand, when the size of a dense matrix of low algebraic complexity is large, Theorem 42 below ensures us that the expected algebraic complexity is still only increased by a constant amount after each successive pre-reduction.*

Hence we conclude this section with the following theorem.

Theorem 42 *The algebraic complexity of a dense matrix increases by 1 after each pre-reduction with probability very close to* 1 *when the size of the original dense matrix is asymptotically large.*

Proof: We proceed by induction on the number of pre-reductions.

Base case: entries $a_{i,j}$ in $\mathbf{A}_1$ are independent and uniformly distributed over $\mathbf{Z}_{[-2,2]}$ conditioning on the outer row and column of matrix $\mathbf{A}$ by proposition 37.

Inductive hypothesis: $\text{alg}(\mathbf{A}_i) = \text{alg}(\mathbf{A}_{i-1}) + 1$ with high probability.

Inductive step: We perform a pre-reduction on $\mathbf{A}_i$. Recall the basic step as illustrated in eq. (20), if $q, a, b \in \mathbf{Z}_{[-i+1,i-1]}$, then by inductive hypothesis, with very high probability $a - bq \in \mathbf{Z}_{[-i,i]}$. By enlarging $\mathbf{Z}_{[-i+1,i-1]}$ to $\mathbf{Z}_{[-i,i]}$, we want to know how many ways there are to choose q, a, b so that $|a - bq| > i + 2$. We have the following three cases to consider:

Case 1) We fix q to be i, then we have $4i^2$ ways to choose a and b so that $|a - bq| > i + 2$. Similarly for fixing q to be $-i$.

Case 2) We fix b to be i, then we have $4i^2$ ways to choose a and b so that $|a - bq| > i + 2$. Similarly for fixing b to be $-i$.

Case 3) We fix a to be i, then we have $4(i-1)^2$ ways to choose q and b so that $|a - bq| > i + 2$. Similarly for fixing a to be $-i$.

So, totally we have $16i^2 + 8(i-1)^2$ ways to choose a, b and q so that $|a - bq| > i + 2$. But there exist $8i^3$ ways of choosing a, b, q over all. since $\frac{16i^2+8(i-1)^2}{8i^3} \approx \frac{1}{i} \rightarrow 0$ *as* $i \rightarrow \infty$, and entries of $\mathbf{A}_i$ are independent and uniformly distributed over $\mathbf{Z}_{[-i+1,i-1]}$ conditioned on the outer row and column of $\mathbf{A}_{i-1}$, so the probability of having a, b and q so that $|a - bq| > i + 2$ is very close to 0 as $i \longrightarrow \infty$.

Hence, we have $\text{alg}(\mathbf{A}_{i+1}) = \text{alg}(\mathbf{A}_i) + 1$ with probability very close to 1 when the size of the original dense matrix is large. $\square$

6 Active Randomization of a (s, t)-sparse Matrix

6.1 Introduction

Our motivating technical problem (sec. 1.1.3) was to consider "random" sparse simplicial complexes. Our analysis has concerned "random" (s, t)-sparse integer matrices. Unfortunately, these two categories are not in one-to-one correspondence because not every random (s, t)-sparse integer matrix corresponds to a legal boundary matrix of a triangulation. We now show how using *active randomization* we can get around this problem. Specifically, while the boundary matrices arising in homology-type computation of a triangulated geometric design are sparse as discussed above, they can have non-uniform distribution and dependence on the probability of their entries being non-zero. Namely, two q-simplices in a triangulation can intersect at some common face, hence constraining some entries in the boundary matrix to be zero. This constraint arises from the simple fact that two simplices of the same dimension must have at least one different vertex. Hence the boundary matrices have non-uniform distribution and dependence on the probability of their entries being non-zero (see sec. 1.1.3 for discussion on the boundary matrices). However, in our probabilistic analysis of normal form computation of a (s, t)-sparse matrix, we assumed that a given (s, t)-sparse matrix has uniform distribution and independent probabilities on its entries being non-zero. Now, we want to relax this assumption. We show that our results go through for a given (s, t)-sparse matrix with non-uniform distribution and dependence on the probability of its entries being non-zero. For convenience, we will adopt the following definitions.

Definition 43 *A (s, t)-sparse matrix is called* non-uniform *and* dependent *if it has non-uniform distribution and dependence on the probability of its entries being non-zero. It is called* uniform *and* independent *if it has uniform distribution and independent probabilities on its entries being non-zero.*

Definition 44 *A permutation σ on n digits is called* random *if σ is uniform among all $n!$ permutations on n digits, i.e., for $j \in \mathbf{Z}_{[1,n]}$, the probability $P(\sigma(i) = j) = \frac{1}{n}$ and for $a_1, a_2, ..., a_{i-1}$ all distinct and different from j, $P(\sigma(i) = j \mid \sigma(1) = a_1, \sigma(2) = a_2, ..., \sigma(i-1) = a_{i-1}) = \frac{1}{n-i+1}$ with $1 \leq i \leq n$.*

In order to cope with non-uniform and dependent boundary matrices in our probabilistic analysis of normal form computation, we propose to *actively randomize* a given non-uniform and dependent (s, t)-sparse matrix. By *active randomization*, we mean the following:

Definition 45 *Consider a $n \times m$ (s,t)-sparse matrix A, that is non-uniform and dependent. We define a new $n \times m$ matrix A' as follows. We generate a random permutation σ on n digits, and initialize the $\sigma(i)^{th}$ row in matrix A' to be equal to the i^{th} row of matrix A for all i with $1 \leq i \leq n$. This is called a* random row permutation.

Definition 46 *A* random column permutation *is defined analogously.* Active randomization *is a sequence of random row and column permutations.*

We will derive the number β of random row and column permutations we need to perform during active randomization in order to obtain a uniform and independent $n \times m$ (s,t)-sparse matrix. We show that β is two, i.e., one random row permutation and one random column permutation. Because of the (s,t)-sparseness of the matrix, each random permutation takes linear time (i.e., $O(n)$). So, even a non-uniform and dependent (s,t)-sparse matrix can be diagonalized into normal form in expected time $O(n^2)$ with very high probability, i.e., this probability is very close to 1 as n is large.

6.2 Active Randomization of a Non-uniform and Dependent (s,t)-sparse Matrix

We first describe a pre-processing algorithm employing active randomization to convert a $n \times m$ non-uniform and dependent (s,t)-sparse matrix into a uniform and independent one with two random row and column permutations. In other words, we perform one random row permutation and one random column permutation. Then we will show that these two operations suffice to perform to actively randomize a non-uniform and dependent (s,t)-sparse matrix so that a uniform and independent (s,t)-sparse matrix can be obtained.

Lemma 47 *A random permutation on n digits can be generated in linear time (i.e. $O(n)$).*

Proof:
We start out with a permutation σ such that $\sigma(i) = i$ for $1 \leq i \leq n$. We loop $n-1$ times. At the i^{th} iteration, we pick a random number j from $\mathbf{Z}_{[i,n]}$ and interchange $\sigma(i)$ with $\sigma(j)$. At the end, we obtain a new permutation σ. For any $k \in \mathbf{Z}_{[1,n]}$, the probability $P(\sigma(i) = k) = \frac{1}{n}$ and for $a_1, a_2, ..., a_{i-1} \in \mathbf{Z}_{[1,n]}$ all distinct and different from k, $P(\sigma(i) = k \mid \sigma(1) = a_1, \sigma(2) = a_2, ..., \sigma(i-1) = a_{i-1}) = \frac{1}{n-i+1}$, so the newly obtained σ is a random premutation. Clearly, the above process of generating a random permutation takes linear time, i.e., $O(n)$. □

6.2.1 The Algorithm for Active Randomization

We proceed as follows. Let A be the given non-uniform and dependent $n \times m$ (s,t)-sparse matrix and A', A'' be dummy $n \times m$ matrices

Algorithm 48 *(Active Randomization)*

Step 1) We generate a random permutation σ on n digits. We loop n times. At the i^{th} iteration, we replace the $\sigma(i)^{th}$ row of matrix A' with the i^{th} row of matrix A.

Step 2) We generate a random permutation τ on m digits. We loop m times. At the i^{th} iteration, we replace the $\tau(i)^{th}$ column of A'' with the i^{th} column of matrix A'.

When diagonalizing a given non-uniform and dependent (s,t)-sparse matrix, we first apply the active randomization algorithm as a pre-processing step to obtain a uniform and independent (s,t)-sparse matrix. Then we apply the reduction algorithm to obtain the normal form of the original non-uniform and dependent (s,t)-sparse matrix.

6.2.2 Probabilistic Analysis of Active Randomization

Clearly, the normal form of a matrix is unchanged under any permutations of rows and columns since such permutations are part of the elementary row and column operations in the reduction algorithm. So, the normal form of a matrix is preserved under random permutations of rows and columns. More importantly, we are interested in the following question:

Question 49 *Can we perform active randomization on a non-uniform and dependent (s,t)-sparse matrix such that a finite number of random row and column permutations results in a uniform and independent matrix? If so, what is the number of random row and column permutations we need to perform?*

For the purpose of this paper, we can wlog assume that the non-zero entries in the non-uniform and dependent (s,t)-sparse matrix are taken from the set $\{-1,1\}$. We can easily generalize to non-uniform and dependent (s,t)-sparse matrices with small algebraic complexity.

In order to answer the above question, let us first introduce some standard tools from probability theory, namely, discrete random variables. Let X be a discrete random variable defined on $\mathbf{Z}_{[1,n]} \times \mathbf{Z}_{[1,m]}$ and taking values from the set $\mathbf{Z}_{[-1,1]}$. Let X_i (resp. X_j) be discrete random variable defined on $\mathbf{Z}_{[1,n]}$ (resp. $\mathbf{Z}_{[1,m]}$) and taking values from $\mathbf{Z}_{[1,n]}$ (resp. $\mathbf{Z}_{[1,m]}$). Clearly, the composite of X with X_i and X_j is also a discrete random variable. In addition, we require that the discrete random variables X have the following probability distribution,

$$\begin{aligned} P(X=0) &= 1-\frac{s}{n}-\frac{t}{m}+\frac{st}{nm} && (48)\\ P(X\neq 0) &= \frac{s}{n}+\frac{t}{m}-\frac{st}{nm} && (49)\\ P(X=-1) &= P(X=1), && (50)\end{aligned}$$

X_i and X_j have uniform probability distribution, and X, X_i, X_j are independent discrete random variables. For a given non-uniform and dependent matrix B, we want to know whether we can actively randomize A to obtain a matrix A' so that

$$P(A'_{i,j}\neq 0) = P(X(i,j)\neq 0) \qquad (51)$$

and the probabilities of its entries being non-zero are independent. Recall the probability distribution of entries being non-zero for a uniform and independent (s,t)-sparse matrix in section 2.1.1. By requiring the above probability distribution (i.e., eq. (48), (49) & (50)) and independence for the discrete random variables X, X_i and X_j, we see that the composite of X with X_i and X_j, i.e., $X(X_i, X_j)$, in fact describes the independent probability distribution of an entry in a uniform and independent (s,t)-sparse matrix. So, if the probabilities of entries in the matrix (after active randomization) satisfy eq. (51), we obtain a uniform and independent (s,t)-sparse matrix. Also we will see later that X_i and X_j capture a random position to which an entry in the original non-uniform and dependent $n \times m$ (s,t)-sparse matrix is moved after one random row permutation and one random column permutation.

We know that after applying the active randomization algorithm, each entry in the original matrix A is moved to somewhere. The following lemma will tell us that each entry of A is moved to a random position.

Lemma 50 *Let A be a non-uniform and dependent $n \times m$ (s,t)-sparse matrix and A' the resulting matrix. Then $A'_{X_i(k),X_j(l)} = A_{k,l}$ where X_i and X_j are discrete random variables defined above.*

Proof:
We wlog assume $A'_{u,v} = A_{k,l}$. Let $\diamond$ sit at the (k,l)-position and $\triangle$ at the (u,v)-position. we have the following picture:

$$\begin{pmatrix} \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \triangle & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \diamond & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \tag{52}$$

Setting X_i (resp. X_j) to some value u (resp. v) with probability $\frac{1}{n}$ (resp. $\frac{1}{m}$) corresponds to randomly permuting to a row (resp. column) which happens to be the u^{th} row (resp. v^{th} column) and moving $\diamond$ to the (u,v)-position. So, pair (X_i, X_j) captures a random position to which an entry in the original non-uniform and dependent (s,t)-sparse matrix is moved. □

Theorem 51 *After actively randomizing a given non-uniform and dependent $n \times m$ (s,t)-sparse matrix by performing one random row permutation and one random column permutation, we obtain an (s,t)-sparse matrix with uniform distribution on the probability of its entries being non-zero.*

Proof:
Let A be a given non-uniform and dependent (s,t)-sparse matrix. We want to show that after one random row permutation and one random column permutation, we can obtain a resulting matrix A' with

$$P(A'_{k,l} \neq 0) = P(X(k,l) \neq 0) \tag{53}$$

where $1 \leq k \leq n, 1 \leq l \leq m$. In order to achieve the equality in eq. (53), we certainly need to assign

$$A'_{X_i(k),X_j(l)} := A_{k,l}. \tag{54}$$

By lemma 50, eq. (54) is guaranteed to be achieved. □

Theorem 52 *After actively randomizing a given non-uniform and dependent $n \times m$ (s,t)-sparse matrix by performing one random row permutation and one random column permutation, we obtain a (s,t)-sparse matrix with independent probabilities of its entries being non-zero.*

Proof:
Let A be our given non-uniform and dependent (s,t)-sparse matrix. We want to show that after one random row permutation and one random column permutation, we can obtain a resulting matrix A' such that $P(A'_{k,l} \neq 0)$ and $P(A'_{u,v} \neq 0)$ are independent with $1 \leq k,u \leq n, 1 \leq l,v \leq m$ and $k \neq u$ or $l \neq v$. In order to achieve these independent probabilities, we certainly need to have

$$\begin{aligned} A'_{k,l} &= X(k,l) && (55)\\ A'_{u,v} &= X(u,v). && (56) \end{aligned}$$

In order to achieve eq. (56) & (56), we need to assign

$$\begin{aligned} A'_{X_i(k),X_j(l)} &:= A_{k,l} && (57)\\ A'_{X_i(u),X_j(v)} &:= A_{u,v}. && (58) \end{aligned}$$

By lemma 50, eq. (57) & (58) are guaranteed to be achieved. □

Theorems 51 & 52 prove the correctness of our active randomization algorithm. We have already noticed that uniformness and independence are stable in the sense that performing any additional active randomization on a uniform and independent (s,t)-sparse matrix will not destroy its uniformness and independence. Therefore, we conclude that performing active randomization by using one random row permutation and one random column permutation on a $n \times m$ non-uniform and dependent (s,t)-sparse matrix will result in a uniform and independent (s,t)-sparse matrix.

7 Cohomology Group Computation of a Triangulation

We can also compute all the cohomology groups of a triangulation in the same time bound. Henceforth, all the cohomology and cochain groups will be taken to have integer coefficients. The cohomology computation for a simplicial complex K is effected as follows:

First, we recall that the chain group $C_p(K)$ of p-chains is free abelian; it has a standard basis consisting of all the oriented p-simplices. Let $\{\sigma_\alpha\}_{\alpha\in J}$ be the standard basis. Let $\{\sigma_\alpha^*\}_{\alpha\in J}$ be the dual basis such that

$$\sigma_\alpha^*(\sigma) = \begin{cases} 1 & \text{if } \sigma = \sigma_\alpha \\ 0 & \text{otherwise} \end{cases}$$

So, every element of $C^p(K;Z) = \mathrm{Hom}(C_p(K), Z)$ is a linear combination of elements of the dual basis $\{\sigma_\alpha^*\}_{\alpha\in J}$, i.e. $C^p(K;Z)$ is a free abelian group with basis $\{\sigma_\alpha^*\}_{\alpha\in J}$.

Second, the coboundary operator δ is defined to be the dual of the boundary operator $\partial : C_{p+1} \to C_p$. Thus,

$$\delta : C^p(K;Z) \to C^{p+1}(K;Z),$$

defined by

$$\langle \delta c^p, d_{p+1}\rangle = \langle c^p, \partial d_{p+1}\rangle$$

where $\langle\cdot,\cdot\rangle$ denotes evaluation, $c^p \in C^p(K;Z), d_{p+1} \in C_{p+1}(K;Z)$. Clearly, δ is a homomorphism.

Third, define $Z^p(K;Z)$ to be the kernel of δ and $B^{p+1}(K;Z)$ the image of δ. Then, the p^{th} cohomology group is defined to be

$$H^p(K;Z) = Z_p(K;Z)/B_p(K;Z).$$

Finally, the matrix associated with ∂ is called the boundary matrix. By the structural theorem of free abelian groups, computing the normal form of the boundary matrix of ∂ gives rise to the structure of $H_p(K;Z)$. From linear algebra, we know that the matrix associated with δ, which is called the coboundary matrix, is the transpose of the corresponding boundary matrix. Since the cohomology groups are also free abelian, computing the normal form of the coboundary matrix (i.e. the transpose of the boundary matrix) gives the structure of the cohomology groups.

8 Conclusions and Future Work

We have described an algorithm for computing all the homology and cohomology groups of a triangulation. While the worst-case analysis of the algorithm indicates that it could run in doubly-exponential time, we identified a class of common simplicial complexes of the kind most frequently encountered in geometric design which have a "sparseness" property that enables the algorithm to run quickly. We formalized this sparseness measure, and then gave a probabilistic analysis of the sparse case to show that the expected running time of the algorithm is roughly quadratic in the geometric complexity (number of simplices) and linear in the dimension.

Cohomology group computation is one the essential steps in computing the higher homotopy groups of a simply-connected triangulation. By using our fast probabilistic algorithm (in section 7), we are working to obtain efficient and simple algorithms for computing the higher homotopy groups of a simply-connected triangulation (see [DC]).

There is a great deal of future work to be done. For example: (1) Probabilistic computation of homology for regular cell complexes and CW complexes. (2) Probablistic computation of other topological invariants. (3) Computation of the cohomology ring operations. As is well-known, the cohomology groups of a space are completely determined by its homology. However, they come equipped with a natural "cohomology operation" that gives the groups a ring structure. Spaces with identical homology and cohomology groups can have different ring structure, hence, the ring structure distinguishes "more finely" between spaces. Deciding ring isomorphism for graded Hopf algebras is a central problem here. Fast algorithmic solutions to this set of problems will prove a considerable challenge to researchers in computational algebraic topology.

9 References

[AHU] **Aho, A. V., Hopcroft, J. E., and Ullman J. D.** *The Design and Analysis of Computer Algorithms,* Addison Wesley (1974).

[Ani] **Anick, D.** *Computing Rational Homotopy Groups is #P-Hard,* in *Computers in Geometry and Topology,* Lecture Notes in Pure and Applied Mathematics, ed, M. Tangora, Marcel Dekker: New York (1989).

[ACM] **Arnon, D., Collins, G., and McCallum, S.** *Cylindrical Algebraic Decomposition I: The Basic Algorithm,* CDS TR-427, Dept. Comp. Sci, Purdue University (1982).

[BKR] **Ben-Or M., Kozen D., and Reif J.**, "The Complexity of Elementary Algebra and Geometry", J. Comp. and Sys. Sciences, Vol. 32, (1986), pp. 251-264.

[BHP] **Boone, W., Haken, W., and Poenaru, V.** *On Recursively Unsolvable Problems in Topology and their Classification,* in "Contributions to Mathematical Logic," ed. H. Schmidt, *et al,* North Holland, pp 37-74 (1968).

[Bro] **Brown, E. H.** *Finite computability of Postnikov complexes,* Ann. Math., **65**, 1-20 (1957).

[Can] **Canny, J.F.** *A New Algebraic method for Robot Motion Planning and Real Geometry,* FOCS (1987).

[CD] **Canny, J.F. and Donald, B. R.** *Simplified Voronoi Diagrams,* Discrete and Computational Geometry **3** (3), pp. 219-236. (1990).

[CC] **Chou, T. J. and Collins, G. E.** *Algorithms for the solution of linear Diophantine equations,* SIAM J. Comput. **11**, pp 687-708. (1982).

[Chu] **Chung, K. L.** *Elementary Probability Theory with Stochastic Processes,* Springer-Verlag. (1979).

[Col] **Collins G. E.** "Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition" Lecture Notes in Computer Science, No. 33, Springer-Verlag, New York, (1975), pp. 135-183.

[Cra] **Crapo, H.** *Applications of Geometric Homology,* Geometry and Robotics, Springer-Verlag LNCS 391, ed. Boissonnat and Laumond (1989).

[Dom] **Domich, P. D.** *Residual Methods for Computing Hermite and Smith Normal Forms,* Ph.D. Thesis, Cornell University. (1985).

[Fel] **Feller, W.** *An Introduction to Probability Theory and its Applications,* John Wiley & Sons. (1968).

[Hir] **Hironaka, H** *Triangulations of Algebraic Sets,* Proc. Symp. Pure Mathematics, American Mathematical Society, (29), pp 165-185 (1975).

[Gla] **Glazunov, N.M.** *Algorithms for calculations of cohomology groups of finite groups,* in: "Computations in algebra and combinatorial analyis", *Akadem. Nauka Ukrainsk. SSR Instituta Kibernetika,* Kiev, (Math. Reviews 82f 20080). (1978).

[Hol] **Holt, D.F.** *The mechanical computation of first and second cohomology groups,* J. Symbolic Computation, (1), pp 351-361 (1985).

[Ili1] **Iliopoulos, C. S.** *Worst-case complexity bounds on algorithms for computing the canonical structure of finite abelian groups and Hermite and Smith normal form of an integer matrix,* SIAM J. Computing, (18) 4, p 658-669 (1989).

[Ili2] **Iliopoulos, C. S.** *Worst-case complexity bounds on algorithms for computing the canonical structure of infinite abelian groups and solving systems of linear diophantine equations,* SIAM J. Computing, (18) 4, p 670-78 (1989).

[Jac] **Jacobson, N.** *Basic Algebra I,* Freeman. (1985).

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

[KY] **Kozen, D. and Yap, C.K.** *Algebraic cell decomposition in* NC, Proc. 26th Symp. Found. Comput. Sci., October 515–521. (1985).

[Lam] **Lambe, L.** *Algorithms for Computing the Cohomology of Nilpotetnt Groups,* in *Computers in Geometry and Topology,* Lecture Notes in Pure and Applied Mathematics, ed, M. Tangora, Marcel Dekker: New York (1989).

[Mar] **Markov. A. A.** *N*erazreshimost' *P*roblemy *G*omeomorfiy, Proc. Intl. Cong. Mathematicians, Cambridge Univ. Press, ed J.A. Todd pp 300-306 (1958).

[McC] **McCallum, S.** *Constructive Triangulation of Real Curves and Surfaces,* University of Sidney (1979).

[Mas] **Massey, W.S.** *Algebraic Topology: An Introduction,* Springer-Verlag, New York (1967).

[Mun] **Munkres** *Elements of Algebraic Topology,* Addison-Wesley: Menlo-Park, CA (1984).

[Smi] **Smith, H.J.S.** *On systems of indeterminate equations and congruences,* Philos. Trans., **151** pp. 293-326 (1861).

[Spa] **Spanier, E. H.** *Algebraic Topology,* Springer-Verlag, New York (1966).

[ST] **Seifert, H., and Threlfall, W.** *Lehrbuch der Topologie,* New York: Chelsea. Chap. 7 (1947).

[SS] **Schwartz J. and Sharir M.,** "On the 'Piano Movers' Problem, II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds," in "Planning, Geometry and Complexity of Robot Motion", ed. by J. Schwartz, J. Hopcroft and M. Sharir, Ablex publishing corp. New Jersey, (1987), Ch. 5, pp. 154-186.

[Sno] **Snoeyink** *A Trivial Knot Whose Spanning Disks Have Exponential Size,* Proc. ACM Symp. on Computational Geometry, Berkeley, CA (1990).

[Tar] **Tarasov, L.** The World Is Built On Probability,, MirPublishers Moscow. (1988).

[VY] **Vegter, G., and Yap, C.K.** *Computational Complexity of Combinatorial Surfaces,* Proc. ACM Symp. on Computational Geometry, Berkeley, CA (1990).

[War] **Warren, J.** *The Effect of Base Points on Rational Bezier Surfaces,* Dept Comp. Sci, Rice University (1990).

[Whi] **Whitney, H.** *Elementary Structure of Real Algebraic Varieties,* Ann. Mathematics. (66) 3, pp 545- 556 (1957).

[Yap] **Yap, C.** *Algorithmic Motion Planning,* Advances in Robotics: Volume 1 (1986).edited by J. Schwartz and C. Yap, Lawrence Erlbaum Associates.

Acknowledgements

We are very grateful to Chee Yap for his comments and suggestions on active randomization, which were very helpful to us. We would also like to thank Jonathan Rees, Peter Kahn, Henry Kesten, Dexter Kozen and Leslie Trotter, Amy Briggs, Pat Xavier, T.V. Raman, Rich Zippel and Mike Erdmann for their comments, and very useful suggestions and technical conversations and advice. Thanks also to John Canny and Andy Neff for encouraging us to write this paper.