Linear Algebraic Representation for Topological Structures

Antonio DiCarlo[†] Alberto Paoluzzi[†] Vadim Shapiro[‡]
[†] *Università "Roma Tre", Italy* [‡] *University of Wisconsin, Madison, USA*

Abstract

With increased complexity of geometric data, topological models play an increasingly important role beyond boundary representations, assemblies, finite elements, image processing, and other traditional geometric modeling applications. While many graph- and index-based data structures have been proposed, no standard representation has emerged as of now. Furthermore, such representations typically do not deal with representations of mappings and functions and do not scale to support parallel processing, open source, and client-based architectures. We advocate that a proper mathematical model for all topological structures is a (co)chain complex: a sequence of (co)chain spaces and (co)boundary mappings. This in turn implies all topological structures may be represented by a collection of sparse matrices. We propose a specific Linear Algebraic Representation (LAR) scheme for mod 2 (co)chain complexes using CSR matrices and show that this representation supports variety of topological computations using standard matrix algebra, without any overhead in space or running time. We also study validity and limitations of the proposed representation scheme and demonstrate its use in a number of common geometric modeling applications.

Key words: solid modeling, representation scheme, sparse matrix, chain complex

1. Introduction

Present-day computational problems in science and technology must deal with increasingly complex geometric information and applications. Complexity of geometric information stems from dramatic increase in size, diversity, and complexity of geometric data: point clouds, boundary meshes, NURBs representations, finite element meshes, CT scans, and so on. Complexity of applications is apparent in increasingly complicated semantics, usually expressed in terms of incidences and relations involving geometric data: large-scale assemblies, topology of microstructure, image segmentation, multi-physics simulation, to name a few. Novel applications require the convergence of shape synthesis and analysis from computer graphics, computer imaging, and computer-aided geometric design, with discrete meshing of domains used for physical simulations. Pursuing such an objective calls for most methods underlying solid and physical modeling to be rethought from scratch to make them distributed and parallel.

The evolution of the field of geometric representations for 3D applications, that can be generally recognized as graph-based data structures representing one of several possible cells complexes partitioning either the boundary or the interior of the represented model. Variety of assumptions about the cell complexes and graph representations make standardization difficult, complicate the issues of data exchange and transfer, and lead to proliferation of incompatible algorithms. Also, graph-based structures are difficult to parallelize: e.g. boundary representation algorithms are dominated by graph searching algorithms (boundary traversals) that tend to force serial processing. To pursue the challenges presented by the current technology, specialized data structures based on cell complexes, that have driven the evolutionary development of the field, do not seem anymore sufficient to us.

In this paper we claim that all representations of cell complexes are properly represented by a (co)chain complex, that captures all combinatorial relationships of interest in solid and physical modeling formally and unambiguously, using standard algebraic topology techniques, that may be implemented with efficient sparse matrix methods. In particular, the linear algebraic representation (LAR) discussed in this paper provides fast processing of topological connectivity encoded within any finite cell decomposition of a geometric domain.

The literature on *representation schemes* in solid modeling started with the foundational paper [12], where a mathematical framework for characterising the important aspects of *computer representations of solids* was introduced. The ground-breaking paper [1] had previously supplied the first but already efficient *boundary representation* scheme.

His author also introduced *Euler operators*, elementary operations for step-wise building well-formed polyhedra, using atomic software actions satisfying the Euler-Poincaré formula at each stage.

The *Quad-Edge* data structure, providing efficient primitives for planar subdivisions and Voronoi diagrams, was proposed in [9], and is still largely used in computational geometry algorithms and in geometric libraries. Variations of the *radial-edge* non-manifold representation by [16] have been embodied in almost every commercial CAD system that uses a non-manifold boundary representation. The *Cell-Tuple* structure [2] was introduced as a simple, uniform representation of finite and *regular* CW-complexes over subdivided d-manifolds.

More general set-theoretic and topological operators were provided by [13] to represent inhomogeneous objects, building upon the *Selective Geometric Complex* (SGC), a superset of *CW-complexes* allowing for cells non homeomorphic to open balls, proposed to handle dimension-independent models of point sets with internal structures and incomplete boundaries. A dimension-independent generalisation of simplicial schemes, and various operators and algorithms were discussed by [11]. Paper [14] reviewed the main ideas and foundations of solid modeling in engineering. In particular, he remarked that solid modeling was conceived as a *universal technology* for developing engineering languages and systems with guaranteed geometric validity.

Today, information and communication technologies are changing at a furious pace, and new challenges are posed by old and new application fields, producing highly demanding requirements. Despite the tremendous amount of research done and the progress made, the most used industrial softwares still follow the basic approach established twenty years ago, centered on boundary representations and non-manifold data structures. Without preprocessing, b-reps are handled sequentially by "boundary traversal" algorithms that depend on specific pointer structures, and this is even worth for higher-dimensional cell complexes, such as 3D unstructured meshes.

Only recently the field started moving away from boundary representations and akin off-line meshing, towards more general cellular decompositions [8,15] and direct integration between geometry and physics [5,6,7]. Similarly, the iso-geometric analysis by [4] requires 3D meshes, embedded using three-variate NURBs, so "allowing models to be designed, tested and adjusted in one integrated stage".

2. LAR scheme

In this section we introduce the *Linear Algebraic Representation* (LAR) scheme. The aim is to provide a representation that could support (at least in principle) all topological queries and constructions that may be asked of the model mesh. We show that the characteristic matrices of a chain complex, i.e. the binary matrices that encode the incidence of d-cells with 0-cells, provide a convenient tool for computing boundary and coboundary operators and answering queries concerning the topological relations between cells. In particular, d-chains represent any possible subset of d-cells, d-cochains represent any possible field over the chains, and boundary and coboundary operators provide the computational tools needed by the discrete version of the generalized Stokes theorem for integration of fields over d-dimensional domains.

Summary. This paper builds over the concepts introduced in [6] and [7], where to find motivations and definitions. In the present paper we discuss a powerful practical realization of the ideas introduced there, after some years of careful experiments and implementations.

In the mod 2 cellular complexes used here, d-chains are *sets* of d-cells; the standard basis of the $\mathbb{Z}_2$-linear space of d-chains C_d is provided by *singletons* of d-cells; each d-cell is represented by a map $C_d \to \mathbb{Z}_2\, C_0$, i.e. by a *row* of a binary characteristic matrix M_d. Of course, every d-chain in C_d may be generated by a $(\mathbb{Z}_2)$-linear combination of M_d rows.

Characteristic matrices are *very* sparse for actual chain complexes used in applications, and can be represented in a standard CSR (Compressed Sparse Row) format. The product and transposition of such CSR matrices, needed to compute the boundary, adjacency and incidence operators between such linear spaces are efficient, since linear with the size of the output.

2.1. *Representation scheme*

In this section we define a simple and efficient computer representation of chain complexes.

2.1.1. *Representation (general case)*

Let a finite cellular complex $\Lambda(X)$ be given, such that $X = \Lambda_0 \cup \cdots \cup \Lambda_d$.

Let also $M_p \in \mathbb{Z}_2^{m \times n}$ $(0 \leq p \leq d)$ be binary characteristic matrices, with number of rows equal to the number of p-cells, and number of columns equal to the number of 0-cells:

$$m = k_p = \sharp\Lambda_p, \qquad n = k_0 = \sharp\Lambda_0$$

Each $m_{ij} \in M_p$ tells us whether (the vertex) $\nu_j \in \Lambda_0$ is contained on the boundary of the cell $\lambda_i \in \Lambda_p$ or not.

LAR scheme. It is a map from mathematical models of solids to their computer representations:

$$\mathrm{LAR} : \mathbf{M} \to \mathbf{R}; \quad \mathbb{C}\mathrm{h}(\Lambda(X)) \mapsto (M_p)_{p=1}^{d}$$

Math models. The LAR domain is the set $\mathbf{M}$ of *chain complexes* $\mathbb{C}$h supported by a finite cellular complex $\Lambda(X)$.

Computer representations. The set $\mathbf{R}$ of d-tuples of binary *compressed sparse row* (CSR) [1] matrices is the LAR codomain, where d is the dimension of the space X.

[1] Compressed Sparse Row (CSR) format, for which efficient implementations on high-performance hardware exist. See [3] and [10].

Validity test of representations. A representation $(M_p)_{p=1}^d$ is *valid* if and only if

$$[\partial_{p-1}][\partial_p]\mathbf{1}_{k_p} = \mathbf{0}_{k_{p-2}} \pmod 2, \qquad 1 \le p \le d. \quad (1)$$

where $k_h = \sharp\Lambda_h$, and the matrices of operators ∂_h ($h = p, p-1$) are constructed by properly filtering the matrices $M_{h-1}\, M_h^t$, as shown in Section 3.2.

Remarks. The interesting point is that, for a given cellular d-complex $\Lambda(X)$, all the M_p matrices ($1 \le p \le d$) have the same number of columns, and can be operated by standard matrix transposition and multiplication. This fact provides a convenient tool for computing boundary and coboundary operators and topological relations between cells.

2.1.2. *Reduced LAR (Special but common important case)*

Two additional assumptions. A d-complex is *regular* when each cell is contained in a d-cell. Let assume:

- LAR contains both the d-cells of a regular decomposition of a d-space and of its complement;
- intersection condition: any two cells intersect on a connected cell.

Claim: Highest dimensional CSR matrix is sufficient for valid LAR. For a regular and polytopal d-complex the only M_d matrix suffices to compute the topology of a space and its (co)boundary operators. Therefore, CSR(M_d) is enough to fully characterize the chain complex induced by $\Lambda(X)$, and to produce a *valid* LAR. For a given decomposition, no ambiguity may arise. The algorithm to compute the matrix M_{p-1} from M_p ($1 \le p \le d$) is given in Section 3.1. The only requirement for validity of M_d is the usual one for cell-complexes, i.e. that the intersection of any two d-cells (considered as sets of 0-cells) is a *single connected cell*.

Compressed LAR. If regular d-complexes are additionally characterized from d-cells with the same number of vertices, a specially compressed version of the CSR representation may be used, since every row of M_d contains the same number of non-zero elements, and therefore it is not necessary to store the index to the beginning of the storage of the column indices of each row.

It is worth noting that the above happens for the important and common case of simplicial and cuboidal d-complexes, where the number of ones per row of the characteristic matrix is equal to $d+1$ and to 2^d, respectively. Of course, we do not need to store the values of non-zero elements in M_d, since it is a binary matrix.

Examples. Finally, we note that a reduced and compressed LAR is very compact. For example, the boundary representation of a 3-manifold has space $|FV| = 2|E| + |V|$, where $|E|$ and $|V|$ are the numbers of the boundary edges and the boundary vertices, respectively. We remember that the Baumgart's *winged-edge* solid representation has space $8|E| + |V| + |F|$. It is worth noting that the reduced LAR exactly coincides with the bulk of common representations in Computer Graphics — for instance, the OBJ and PLY file formats.

3. Mappings and Algorithms

3.1. *Computing $(d-1)$-chains from d-chains*

A characteristic matrix M_d contains, by rows, the generators of the linear space C_d, that coincide with the unit d-chains, i.e. the singletons consisting of a cell in the partition $\Lambda_d(X)$. In this section we show how to compute the matrix M_{d-1} starting from M_d. In other words, we give a map $\phi : M_d \mapsto M_{d-1}$.

Cell partition of the full space. Start from the characteristic matrix M_d of a cellular partition $\Lambda_d = \Lambda_d(X) \cup \Lambda_d(Y \backslash X)$ where Y is the embedding space of X. In other words, we assume in this case

$$\Lambda_d = \Lambda_d(X) \cup \Lambda_d(\complement X).$$

Computation of the adjacency matrix. The adjacency relation between d-cells is the subset of cell pairs that share some vertex (0-cell). The adjacency matrix A_d can be computed by standard matrix multiplication between the characteristic matrix $M_d(\Lambda_d)$ and its traspose:

$$A_d = M_d\, M_d^t$$

$A_d = (a_{i,j})$ is integer-valued and symmetric. Each $a_{i,j}$ gives the *number* of 0-cells shared between the cells $\lambda_i, \lambda_j \in \Lambda_d$.

Computation of facets. When the set intersection of two cells $\lambda_i, \lambda_j \in \Lambda_d$ contains a number of vertices greater or equal to the dimension d of cells, they share a common facet $\mu \in \Lambda_{d-1}$, that can be computed as component-wise integer product (or binary intersection) of rows i and j of M_d. The generated characteristic vector gives a row of the output matrix M_{d-1}. In symbols:

$$a_{i,j} = \sharp(\lambda_d^i \cap \lambda_d^j)$$
$$a_{i,j} \ge d \quad (i \ne j) \quad \Rightarrow \quad \exists\, M_d(i) \wedge M_d(j) = \mu \in \Lambda_{d-1}$$

Example. An example of computation of the M_1 characteristic matrix starting from M_2 is given in Section 4.2 for the regular polytopal complex shown in Figure 1b.

For instance, the edge $e_2 = (v_0, v_6)$ is generated from the term $a_{0,1} = 2$ of the cell adjacency matrix $A_2 = M_2 M_2^t$ given in (3). The edge e_2 is computed by the entrywise product (intersection) of rows $M_2(0)$ and $M_2(1)$:

$$(1\,1\,0\,0\,0\,0\,1\,1) (1\,0\,1\,0\,1\,0\,1\,0) = (1\,0\,0\,0\,0\,0\,1\,0)$$

$M_2(0)$ and $M_2(1)$ are related to the 2-cells f_0 and f_1 of Figure 1b.

Efficiency of computation. This computation is efficient with CSR matrices, because zero elements are not stored, so that each row of CSR(A_d) actually stores only the numbers of vertices shared by a d-cell with all the incident ones. The component-wise product of two binary CSR rows i and j of M_d is efficient, and at most takes time $O(\sharp\lambda_i + \sharp\lambda_j)$.

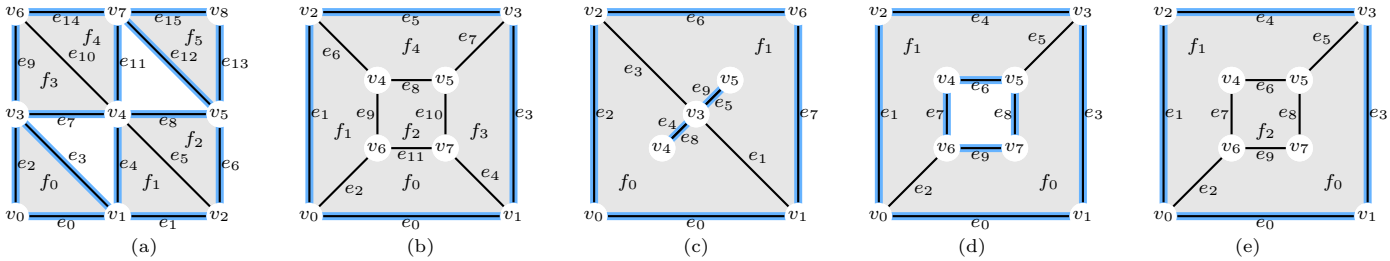

Fig. 1. Cellular 2-complexes and their boundaries (blue): (a) non-manifold simplicial complex; (b) cuboidal complex; (c) complex with internal boundaries; (d,e) complexes with non-convex 2-cells.

3.2. *Boundary/coboundary matrix*

Consider the *incidence operator* $\mathcal{I}_{p-1}^{p} : C_p \to C_{p-1}$ and its matrix

$$[\mathcal{I}_{p-1}^{p}] := M_{p-1}^{p} = M_{p-1}\, M_p^t.$$

The entry $M_{p-1}^{p}(i,j)$ stores the value of the application of the $(p-1)$-cochain μ_{p-1}^{i} on the p-chain λ_p^j:

$$M_{p-1}^{p}(i,j) = \langle \mu_{p-1}^{i}, \lambda_p^j \rangle = \mu_{p-1}^{i}(\lambda_p^j)$$

By standard multiplication in $\mathbb{Z}$ (*not* $\mathbb{Z}_2$), we get

$$\mu_{p-1}^{i}(\lambda_p^j) = \sum_{h=0}^{k_0-1} \left(M_{p-1}(i,h)\right)\left(M_p(j,h)\right) = \sharp(\mu_{p-1}^{i} \cap \lambda_p^j),$$

thus computing the *number of vertices of the intersection*, i.e. of the common face, between $\mu_{p-1}^{i} \subset \Lambda_0$ and $\lambda_p^j \subset \Lambda_0$.

This common face coincides with μ_{p-1}^{i} if and only if

$$\sharp(\mu_{p-1}^{i} \cap \lambda_p^j) = \sharp\mu_{p-1}^{i}.$$

In such a case, we have

$$\mu_{p-1}^{i} \wedge \lambda_p^j = \mu_{p-1}^{i} \in \partial\lambda_p^j.$$

3.2.1. *Boundary computation algorithm*

Therefore, as a computational procedure to calculate the matrix of the *unoriented boundary* operator, we have the following algorithm:

(i) Compute $M_{p-1}^{p} := M_{p-1}M_p^t$ by standard matrix product of (sparse) matrices.
(ii) For each $0 \le i \le k_{p-1} - 1$,
  (a) compute the number of stored elements in row i of CSR(M_{p-1}):

$$k = \sum_h M_{p-1}(i,h) =: \sharp\mu_{p-1}^{i},$$

  (b) and, for each $0 \le j \le k_p - 1$, set:

$$[\,\partial_p\,](i,j) = \begin{array}{ll} 1 & \text{if} \quad M_{p-1}^{p}(i,j) = k; \\ 0 & \text{otherwise.} \end{array}$$

Example Here we compute the $[\partial_2]$ boundary matrix for the regular polytopal complex shown in Figure 1b, starting from characteristic matrices M_2 and M_1 given in Section 4.2. According to step (i) of the algorithm above, we get

$$M_1^2 = M_1\, M_2^t$$

Then, we filter the matrix M_1^2 according to the criterion given in step (ii), producing the matrix of the boundary operator $\partial_2 : C_2 \to C_1$:

Of course, $[\partial_2]$ contains by row the (characteristic vectors of) generators of the C_1 linear space, i.e. the edges of the complex, indexed by faces, and contains by columns the (characteristic vectors of) generators of the C_2 linear space, i.e. the faces of the complex, indexed by edges, as it is possible to check in Figure 1b.

$$[\partial_2]^t = \begin{pmatrix} 1&0&1&0&1&0&0&0&0&0&0&1 \\ 0&1&1&0&0&0&1&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&1&1&1&1 \\ 0&0&0&1&1&0&0&1&0&0&1&0 \\ 0&0&0&0&0&1&1&1&1&0&0&0 \\ 1&1&0&1&0&1&0&0&0&0&0&0 \end{pmatrix} \tag{2}$$

3.3. *Boundary queries*

The knowledge of a d-boundary matrix is very useful, since it encodes the boundaries of *all possible chains*, i.e. of *all possible subsets* of d-cells.

It is also computationally efficient, since the calculation of $[\mu] = [\partial_d][\lambda]$, where $[\mu], [\lambda]$ are the characteristic vectors of chains $\mu \in C_{d-1}$ and $\lambda \in C_d$, is actually performed using only the non-zero terms stored in their CSR matrices.

In particular, the boundary of the partition $\Lambda_d(X) \cup \Lambda_d(\complement X)$, of a space and its complement, is performed by setting the characteristic vector $[\Lambda_d]$ with $\sharp\Lambda_d(X)$ ones and $\sharp\Lambda_d(\complement X)$ zeros.

Example: boundary of the total chain. Here we discuss the computation of the boundary of the regular polytopal complex of Figure 1b. In this case we set

$$[\Lambda_2] = [1,1,1,1,1,0]^t$$

and therefore we get for the boundary 1-chain:

$$[\delta_2][\Lambda_2] = [1,1,0,1,0,1,0,0,0,0,0,0]^t \pmod 2$$

Writing the boundary 1-chain of Figure 1b as a set of 1-cells (edges), we have $\{e_0, e_1, e_3, e_5\}$.

Example: boundary of another chain. If we remove from $\Lambda_2(X)$ the central cell f_4, and insert it in the exterior space

partition $\Lambda_2(\complement X)$, we set

$$[\Lambda_2] = [\Lambda_2(X) \cup \Lambda_2(\complement X)] = [1,1,1,1,0,0]^t$$

and get

$$[\delta_2][\Lambda_2] = [1,1,0,1,0,1,0,0,1,1,1,1]^t \pmod 2$$

giving the boundary chain $\{e_0, e_1, e_3, e_5, e_8, e_8, e_{10}, e_{11}\}$.

3.4. *Product of cellular complexes*

Let $A = \Lambda(X)$ and $B = \Lambda(Y)$ be cell complexes of dimension m and n, respectively. Then, $C = A \times B = \Lambda(X \times Y)$ is a cell complex of dimension $m+n$, with cells the products $\lambda^h \times \lambda^k$, with $\lambda^h \in \Lambda_m(X)$ and $\lambda^k \in \Lambda_n(Y)$.

Assuming $X \subset \mathbb{E}^c$ and $Y \subset \mathbb{E}^d$, it is $X \times Y \subset \mathbb{E}^{c+d}$.

If $V(X)$ and $V(Y)$ are the column matrices of positions of the X and Y vertices, then we have

$$V(X \times Y) = V(X) \otimes V(Y)^t.$$

The implementation in the LAR scheme is very simple, reducing to a double loop on cells and vertices.

3.5. *Star operator*

The star of a cell λ in a cellular complex $\Lambda(X)$, denoted $\mathrm{St}\,\lambda$, is the set of all cells in Λ that have at least a face in λ. In algebraic terms, it is a linear endomorphism of

$$\mathcal{C} = \oplus_p C_p$$

such that

$$\mathrm{St} : \mathcal{C} \to \mathcal{C}.$$

In practice, it is easy to compute. To get the star of an elementary chain $\{\lambda\} \in C_p$, just consider its unit covector representation $[\lambda]^t$, multiply it by all the matrices $M_{p,q} := M_p\, M_q^t$, and make the union of the resulting sets of cells.

Application: distributed models An important application of the star operator is the following. Let suppose to allocate a *big geometric model* across a distributed computational environment. In order to execute any distributed simulation, the (image of the) star of each submodel (as a chain) must be computed, and allocated on both sides of the pair of processes sharing a portion of the submodel boundary.

Stars are also used for enforcing continuity of mappings between the complexes, as well as in physics and in mesh processing. E.g. Laplace operators operate on open neighborhoods, and can be easily generated by this operator.

4. LAR examples

We discuss a few elementary examples, in order to demonstrate the simple computations required by LAR. In all of our examples, `FV` and `EV` give a CSR representation of the binary matrices M_2 and M_1, respectively.

4.1. *Non-manifold simplicial complex*

Let's consider the $\Lambda(X)$ decomposition in Figure 1a, i.e. the simplicial complex $\Lambda = \Lambda_0 \cup \Lambda_1 \cup \Lambda_2$, with $k_0, k_1, k_2 = 9, 16, 6$, that is locally *non-manifold*.

```
FV = [[0,1,3],[1,2,4],[2,4,5],[3,4,6],[4,6,7],[5,7,8]]
```

The array `FV` $= \mathrm{CSR}(M_2)$ encodes the linear space $C_2(\Lambda)$ of 2-chains, generated by the matrix rows, through a standard basis (isomorphic to the 2-cells). The 2-cells are given as linear combinations of the standard basis of 0-chains (isomorphic to the 0-cells). This information is provided for the linear space $C_1(\Lambda)$ of 1-chains by the array `EV` $= \mathrm{CSR}(M_1)$. Note that `FV` is the only needed input: for a simplicial complex, the array `EV` may be computed in time linear with the output size, by using either a well-known combinatorial formula [11], or the method given in Section 3.1.

4.2. *Convex-cell complex*

The cellular complex shown in Figure 1b is composed by five quads, described symbolically by quadruples of vertex indices (`FV`), and is embedded in $\mathbb{E}^2$ by assigning a pair of coordinates to position vectors of vertices (`V`).

```
V = [[0.,0.],[1.,0.],[2.,0.],[0.,1.],[1.,1.],[2.,1.],
     [0.,2.],[1.,2.],[2.,2.]]
FV = [[0,1,6,7],[0,2,4,6],[4,5,6,7],[1,3,5,7],[2,3,4,5]]
```

In order to automatically extract the edge information with matrix-based methods, we must append to `FV` the description of one or more external 2-cells, in this case denoted by the quadruple `[0,1,2,3]`, so getting a revised

```
FV = [[0,1,6,7],[0,2,4,6],[4,5,6,7],[1,3,5,7],[2,3,4,5],
      [0,1,2,3]]
```

Using the technique introduced in Section 3.1, we first convert the input `FV` into the (binary) characteristic matrix M_2 such that $\mathrm{CSR}(M_2) =$ `FV`:

$$M_2 = \begin{pmatrix} 1&1&0&0&0&0&1&1 \\ 1&0&1&0&1&0&1&0 \\ 0&0&0&0&1&1&1&1 \\ 0&1&0&1&0&1&0&1 \\ 0&0&1&1&1&1&0&0 \\ 1&1&1&1&0&0&0&0 \end{pmatrix}, \quad A_2 = M_2\, M_2^t = \begin{pmatrix} 4&2&2&2&0&2 \\ 2&4&2&0&2&2 \\ 2&2&4&2&2&0 \\ 2&0&2&4&2&2 \\ 0&2&2&2&4&2 \\ 2&2&0&2&2&4 \end{pmatrix} \tag{3}$$

and for each item $a_{ij} \in A_2$ $(i < j)$ such that $a_{ij} \geq d = 2$, compute the binary intersection between the rows i and j of M_2, and put the binary result as a row of the characteristic matrix of the 1-cells, named M_1. Therefore we get:

$$M_1^t = \begin{pmatrix} 1&1&1&0&0&0&0&0&0&0&0&0 \\ 1&0&0&1&1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&1&1&0&0&0&0&0 \\ 0&0&0&1&0&1&0&1&0&0&0&0 \\ 0&0&0&0&0&0&1&0&1&1&0&0 \\ 0&0&0&0&0&0&0&1&1&0&1&0 \\ 0&0&1&0&0&0&0&0&0&1&0&1 \\ 0&0&0&0&1&0&0&0&0&0&1&1 \end{pmatrix},$$

from which we derive `EV` $= \mathrm{CSR}(M_1)$:

```
EV = [[0,1],[0,2],[0,6],[1,3],[1,7],[2,3],[2,4],[3,5],
      [4,5],[4,6],[5,7],[6,7]]
```

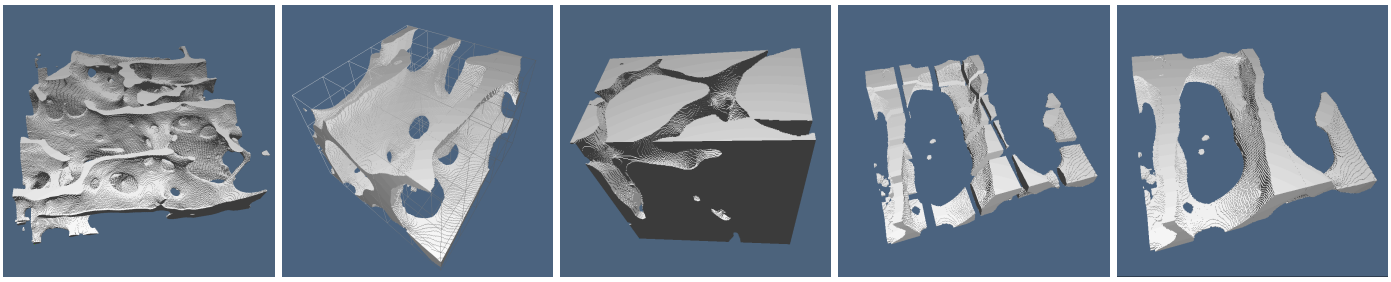

Fig. 2. Models extracted from 3D images of spongy bone: (a,b) from a portion of $250 \times 250 \times 150$ pixel; (c) a portion of exterior space; (d) stream of chunks generated by work-units using a 25MB [∂_3] matrix in constant memory in the OpenCL implementation; (e) partial model.

4.3. *Models from images*

LAR is being developed as a web service in a web-based computational environment, using some Khronos's APIs (OpenCL and WebCL) for industry standard heterogeneous computing. We are using this algebraic representation to help dealing with biomedical applications, which require fast performances with big geometric data. [2] For instance, we handle density values in medical images as a scalar field (cochain) over a cubical cellular complex, and use LAR for topologically correct 3D image segmentation, and for (enumerative) solid model extraction from image, to be subsequently made smooth. A nice characteristic of this approach is that the whole image is partitioned into a set of cochains associated to various field intervals, including the interstitial space, so providing a well-defined meshing of both the features and their outer space.

5. Conclusions

In this paper we introduced LAR, a simple algebraic representation for cellular complexes, using a CSR (Compressed Sparse Row) form for characteristic matrices of linear spaces of (co)chains. LAR allows to compute and query any model topology through sparse matrix multiplication, and supports all topological incidence structures, including enumerative (images), decompositive (meshes) and boundary (CAD) representations. It is dimension-independent, not restricted to regular complexes, and allows for internal/external structures, including cracks and fibers of lower dimensions. Furthermore, LAR enjoys a neat mathematical format—being based on *chains*, the domains of discrete integration, and *cochains*, the discrete prototype of differential forms, so naturally integrating the geometric shape with the supported physical properties. In conclusion, LAR seems to provide some very promising directions, in particular towards manycore/heterogeneous computing and web computations with distributed models and parallel algorithms, and offers a sound basis for a standardization work in this field.

[2] In particular, LAR is being used for model extraction from 3D images within the framework of the IEEE-SA Project P3333.2 - Standard for "Three-Dimensional Model Creation Using Unprocessed 3D Medical Data".

References

[1] B. G. Baumgart, Winged edge polyhedron representation., Tech. Rep. Stan-CS-320, Stanford, CA, USA (1972).

[2] E. Brisson, Representing geometric structures in d dimensions: topology and order, in: Proc. of the 5-th Annual Symposium on Computational Geometry, SCG '89, Acm, New York, NY, USA, 1989, pp. 218–227.

[3] A. Buluç, J. R. Gilbert, Parallel sparse matrix-matrix multiplication and indexing: Implementation and experiments, SIAM J. of Scientific Computing (SISC) 34 (4) (2012) 170 – 191.

[4] J. Cottrell, T. Hughes, Y. Bazilevs, Isogeometric analysis: toward integration of CAD and FEA, Wiley, 2009.

[5] M. Desbrun, A. N. Hirani, M. Leok, Jerrold, E. Marsden, Discrete exterior calculus, Tech. rep. (2003).

[6] A. DiCarlo, F. Milicchio, A. Paoluzzi, V. Shapiro, Solid and physical modeling with chain complexes, in: Proceedings of the 2007 ACM symposium on Solid and physical modeling, SPM '07, Acm, New York, NY, USA, 2007, pp. 73–84.

[7] A. DiCarlo, F. Milicchio, A. Paoluzzi, V. Shapiro, Chain-based representations for solid and physical modeling, Automation Science and Engineering, IEEE Trans. on 6 (3) (2009) 454 –467.

[8] X. Feng, Y. Wang, Y. Weng, Y. Tong, Compact combinatorial maps in 3d, Graphical Moddels 75 (3) (2012) 149–156.

[9] L. Guibas, J. Stolfi, Primitives for the manipulation of general subdivisions and the computation of voronoi, ACM Trans. Graph. 4 (2) (1985) 74–123.

[10] A. Lokhmotov, Implementing sparse matrix-vector product in OpenCL, in: OpenCL tutorial, 6th Int. Conference on High Performance and Embedded Architectures and Compilers (HiPEAC'11), 2012.

[11] A. Paoluzzi, F. Bernardini, C. Cattani, V. Ferrucci, Dimension-independent modeling with simplicial complexes, ACM Trans. Graph. 12 (1) (1993) 56–102.

[12] A. G. Requicha, Representations for rigid solids: Theory, methods, and systems, ACM Comp. Surv. 12 (4) (1980) 437–464.

[13] J. R. Rossignac, M. A. O'Connor, SGC: a dimension-independent model for pointsets with internal structures and incomplete boundaries, in: Geometric modeling for product engineering, North-Holland, 1990.

[14] V. Shapiro, Solid modeling, in: G. Farin, J. Hoschek, S. Kim (eds.), Handbook of Computer Aided Geometric Design, chap. 20, Elsevier Science, 2002, pp. 473–518.

[15] K. Wang, X. Li, B. Li, H. Xu, H. Qin, Restricted trivariate polycube splines for volumetric data modeling, IEEE Trans. Vis. Comput. Graph. 18 (5) (2012) 703–716.

[16] K. J. Weiler, The radial edge structure: A topological representation for non-manifold geometric modelling, in: M. Wozny, H. McLaughlin, J. Encarnacao (eds.), Geometric modelling for CAD applications, North-Holland, Amsterdam, 1988, pp. 3–12.