

Geometric & Graphics Programming Lab: Lecture 14

Alberto Paoluzzi

November 18, 2016

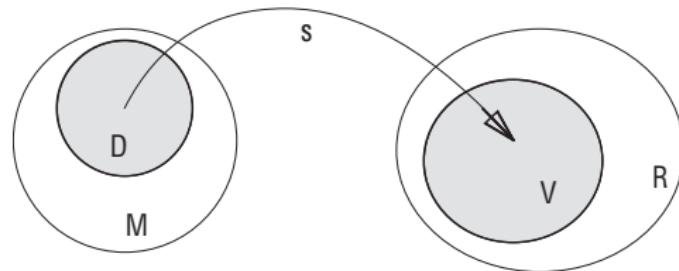
Outline: Cell complexes (LAR)

- 1 Solid Modeling
- 2 Linear Algebraic Representation (LAR)
- 3 LAR: computer representation
- 4 Operations

Solid Modeling

Representation scheme: definition

mapping $s : M \rightarrow R$ from a space M of mathematical models to a space R of computer representations



- ① The M set contains the mathematical models of the class of solid objects the scheme aims to represent
- ② The R set contains the symbolic representations, i.e. the proper data structures, built according to a suitable grammar

A. Requicha, [Representations for Rigid Solids: Theory, Methods, and Systems, ACM Comput. Surv.](#), 1980.

V. Shapiro, [Solid Modeling](#), In [Handbook of Computer Aided Geometric Design](#), 2001

Representation schemes

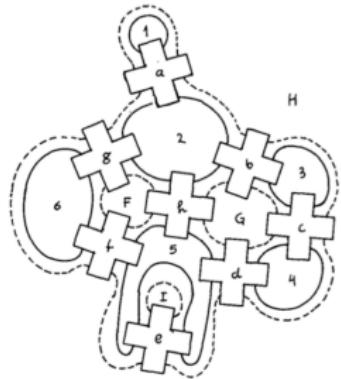
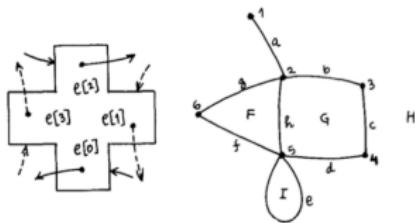
Most of such papers introduce or discuss one or more representation schemes . . .

- 1 Requicha, ACM Comput. Surv., 1980, [?]
- 2 Requicha & Voelcker, PEP TM-25, 1977, [?]
- 3 Rossignac & Requicha, Comput. Aided Des., 1991, [?]
- 4 Bowyer, SVLIS, 1994, [?]
- 5 Baumgart, Stan-CS-320, 1972, [?]
- 6 Braid, Commun. ACM, 1975, [?]
- 7 Dobkin & Laszlo, ACM SCG, 1987, [?]
- 8 Guibas & Stolfi, ACM Trans. Graph., 1985, [?]
- 9 Woo, IEEE Comp. Graph. & Appl., 1985, [?]
- 10 Yamaguchi & Kimura, Comp. Graph. & Appl., 1995, [?]
- 11 Gursoz & Choi & Prinz, Geom.Mod., 1990, [?]
- 12 S.S.Lee & K.Lee, ACM SMA, 2001, [?]
- 13 Rossignac & O'Connor, IFIP WG 5.2, 1988, [?]
- 14 Weiler, IEEE Comp. Graph. & Appl., 1985, [?]
- 15 Silva, Rochester, PEP TM-36, 1981, [?]
- 16 Shapiro, Cornell Ph.D Th., 1991, [?]
- 17 Paoluzzi et al., ACM Trans. Graph., 1993, [?]
- 18 Pratt & Anderson, ICAP, 1994, [?]
- 19 Bowyer, Djinn, 1995, [?]
- 20 Gomes et al., ACM SMA, 1999, [?]
- 21 Raghothama & Shapiro, ACM Trans. Graph., 1998, [?]
- 22 Shapiro & Vossler, ACM SMA, 1995, [?]
- 23 Hoffmann & Kim, Comput. Aided Des., 2001, [?]
- 24 Raghothama & Shapiro, ACM SMA, 1999, [?]
- 25 DiCarlo et al., IEEE TASE, 2008, [?]
- 26 Bajaj et al., CAD&A, 2006, [?]
- 27 Pascucci et al., ACM SMA, 1995, [?]
- 28 Paoluzzi et al., ACM Trans. Graph., 1995, [?]
- 29 Paoluzzi et al., Comput. Aided Des., 1989, [?]
- 30 Ala, IEEE Comput. Graph. Appl., 1992, [?]

and much more . . .

Representation scheme: Quad-Edge data structure

(Guibas & Stolfi, ACM Transactions on Graphics, 1985)



- (a) Edge record showing Next links.
- (b) A subdivision of the sphere.
- (c) Data structure for the subdivision

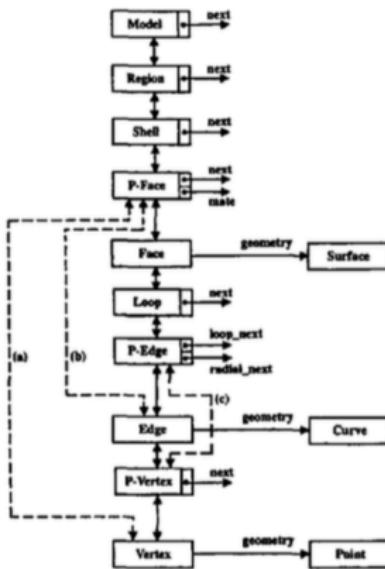
Primitives for the Manipulation
of General Subdivisions and
the Computation of Voronoi
Diagrams

largely used in computational
geometry algorithms and in
geometric libraries

Representation scheme: Partial-Entity data structure

(Sang Hun Lee & Kunwoo Lee, ACM Solid Modeling, 2001)

Compact Non-Manifold Boundary Representation Based on Partial Topological Entities



```

class Entity {
    int _id;
    Attribute *_attribute;
};

class Model : public Entity {
    Model *_next; // next model
    Region *_region; // list of regions
};

class Region : public Entity {
    Region *_next; // link field of the region list of a model
    Model *_model; // parent model
    Shell *_shell; // peripheral shell
};

class Shell : public Entity {
    Shell *_next; // next void shell
    Region *_region; // parent region
    Pface *_pface; // partial face
};

class Pface : public Entity { // partial face (p-face) class
    Pface *_next; // next p-face
    Shell *_shell; // parent shell
    Entity *_child; // child entity: a face, an edge, or a vertex
    Orient *_orient; // orientation flag w.r.t. the face normal
    Pface *_mate; // mate p-face
};

class Face : public Entity {
    Pface *_pface; // one of two incident p-faces
    Loop *_loop; // peripheral loop
    Surface *_geometry; // surface
};

class Loop : public Entity { // next hole loop
    Loop *_next; // next hole loop
    Face *_face; // parent face
    Pedge *_pedge; // a p-edge in a loop
};

class Pedge : public Entity { // partial edge (p-edge) class
    Pedge *_loop; // parent loop
    Entity *_child; // child entity: an edge or a p-vertex
    Orient *_orient; // orientation flag w.r.t. the edge direction
    Pvertex *_pvertices; // start p-vertex
    Pedge *_looped_prev; // previous p-edge in the loop cycle
    Pedge *_looped_next; // next p-edge in the loop cycle
    Pedge *_radial_prev; // previous p-edge in the radial cycle
    Pedge *_radial_next; // next p-edge in the radial cycle
};

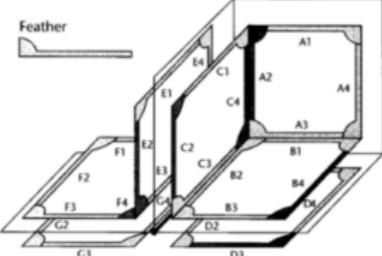
class Pvertex : public Entity { // partial vertex (p-vertex) class
    Pvertex *_next; // another p-vertex associated with _vertex
    Entity *_parent; // parent entity: an edge or a p-edge
    Vertex *_vertex; // mother vertex
};

class Vertex : public Entity { // partial vertex (p-vertex) class
    Entity *_parent; // parent entity: a p-vertex or a p-face
    Point *_geometry; // position
};

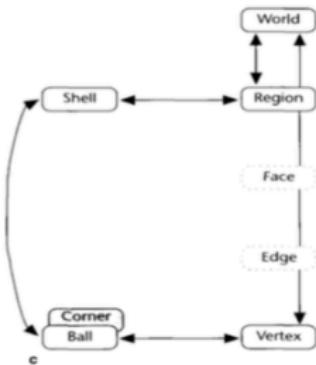
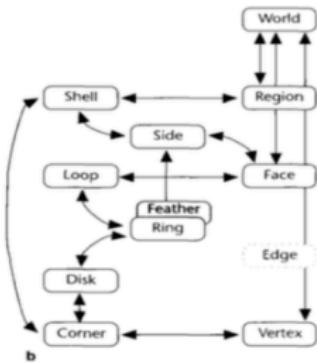
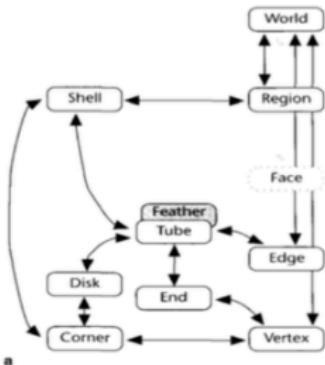
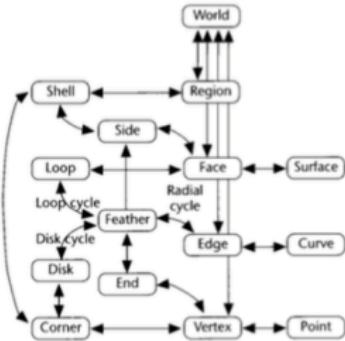
```

Representation scheme: Coupling Entities data structure

(Yamaguchi & Kimura, IEEE Computer Graphics and Applications, 1995)



Fan	$B_4 \& D_3$	$:D_3 = FM(B_4), B_4 = FM(D_3)$
Blade	$C_2 \& E_2$	$:E_2 = BM(C_2), C_2 = BM(E_2)$
Wedge	$A_2 \& C_4$	$:C_4 = WM(A_2), A_2 = WM(C_4)$
Loop cycle	$A_1 >> A_2 >> A_3 >> A_4$	$:A_2 = CCL(A_1), A_3 = CCL(A_2)$
	$A_4 >> A_3 >> A_2 >> A_1$	$:A_1 = CL(A_2), A_2 = CL(A_3)$
Radial cycle	$D_2 >> C_3 >> F_4$	$:C_3 = CCR(D_2), F_4 = CCR(C_3)$
	$F_4 >> C_3 >> D_2$	$:D_2 = CR(C_3), C_3 = CR(F_4)$
Disk cycle	$A_3 >> C_4 >> B_2$	$:C_4 = CCD(A_3), B_2 = CCD(C_4)$
	$B_2 >> C_4 >> A_3$	$:A_3 = CD(C_4), C_4 = CD(B_2)$



Linear Algebraic Representation (LAR)

LAR: Mathematical models

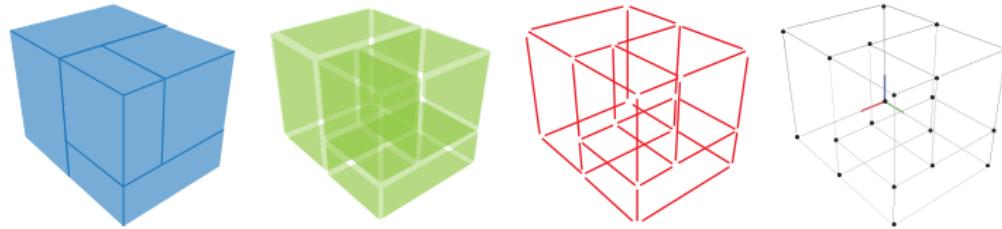
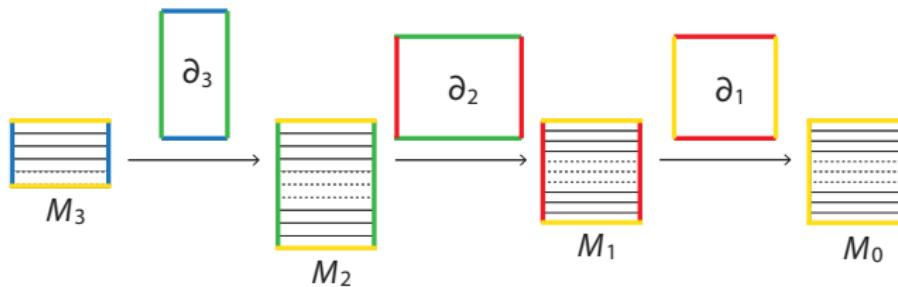
Space of CW-complexes (cellular complexes)

Chain complex (of chain spaces)

Sequence of linear spaces (over \mathbb{Z}_2) of d -cell subsets

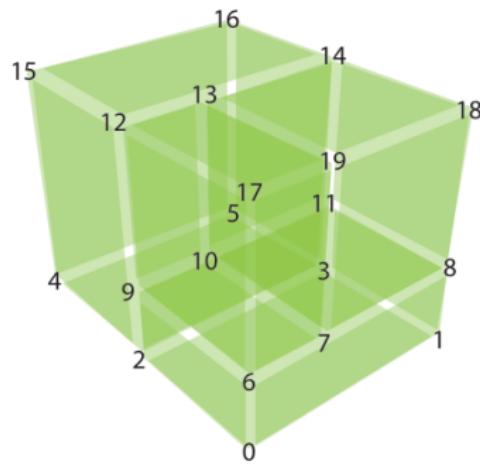
A **chain complex C** is a complex of **chain spaces** and **boundary maps**:

Unit d -chains (single d -cell subsets), give the **standard bases** (M_d rows) of d -chain spaces



Characteristic matrices of d -chain spaces

Matrix representation of the basis — d -cells as subsets of vertices



$$M_3 = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$M_2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

LAR: computer representation

Sparse matrix representations

Basic representations

A few basic representation of topology are used in LARCC.

They include some common sparse matrix representations:

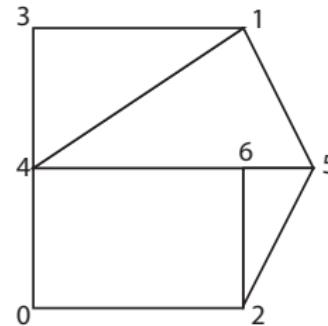
- CSR (Compressed Sparse Row),
- CSC (Compressed Sparse Column),
- COO (Coordinate Representation),
- and BRC (Binary Row Compressed)

Data definition

- We can identify each cell with its set of vertices.
- Thus, to define S , we start from the vertex set V , and specify the vertex subsets which correspond to the cells, and the rank of each cell.
- The partial order is induced by set inclusion.

Characteristic matrix $M_2 : C_0 \rightarrow C_2$

$$M_2 = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$



BRC (Binary Row Compressed) representation

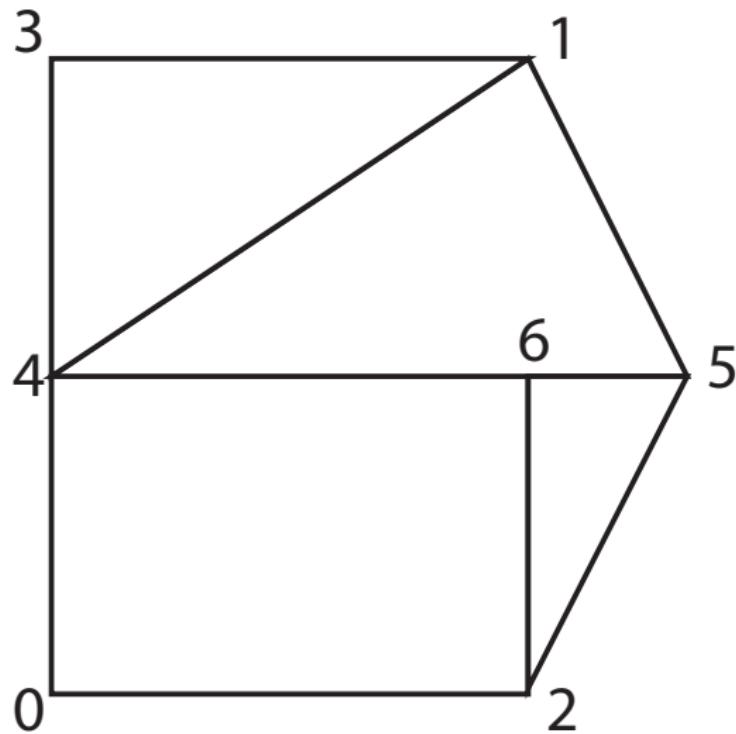
List of list of integer

typically used for input of cell complexes

Let $A = (a_{i,j} \in \{0, 1\})$ be a characteristic matrix.

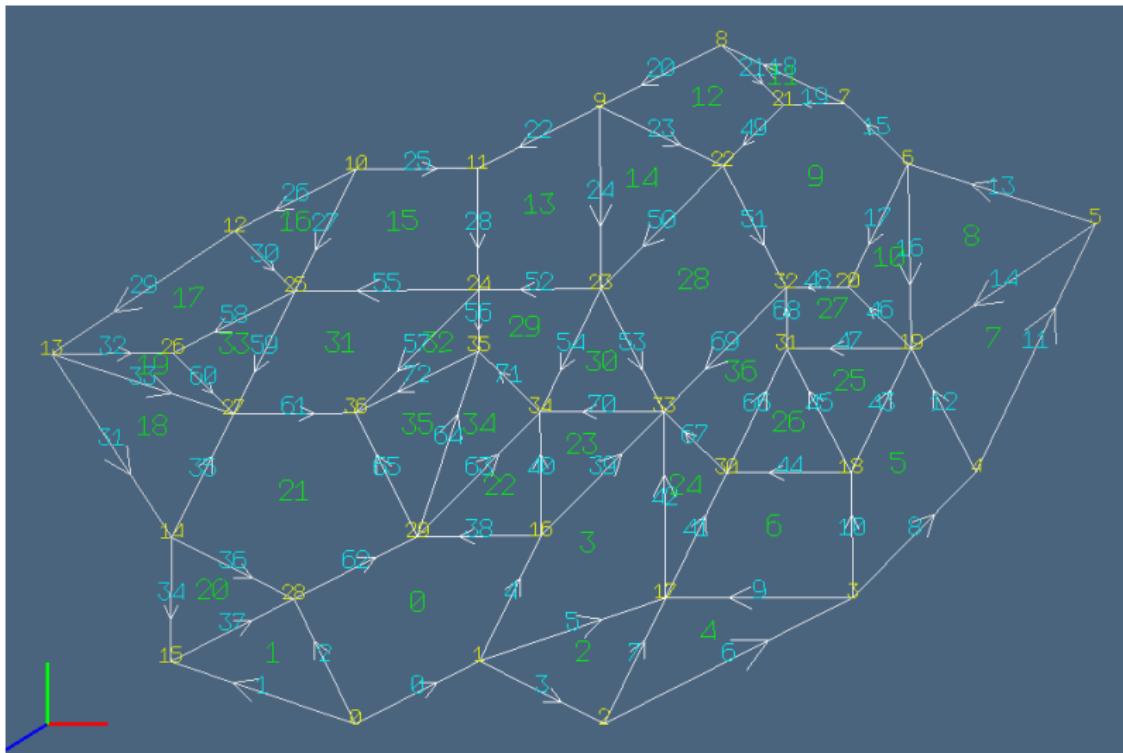
$$A = \left(\begin{array}{ccccccccc} 0, 1, 0, 0, 0, 0, 0, 1, 0, 0 \\ 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 \\ 1, 0, 0, 1, 0, 0, 0, 0, 0, 1 \\ 1, 0, 0, 0, 0, 0, 1, 0, 0, 0 \\ 0, 0, 0, 0, 0, 1, 1, 1, 0, 0 \\ 0, 0, 1, 0, 1, 0, 0, 0, 1, 0 \\ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \\ 0, 1, 0, 0, 0, 0, 0, 1, 0, 1 \\ 0, 0, 0, 1, 0, 0, 0, 0, 1, 0 \\ 0, 1, 1, 0, 1, 0, 0, 0, 0, 0 \end{array} \right) \mapsto \text{BRC}(A) = [[1, 7], [2], [0, 3, 9], [0, 6], [5, 6, 7], [2, 4, 8], [], [1, 7, 9], [3, 8], [1, 2, 4]]$$

The idea: generalisation of abstract simplicial sets



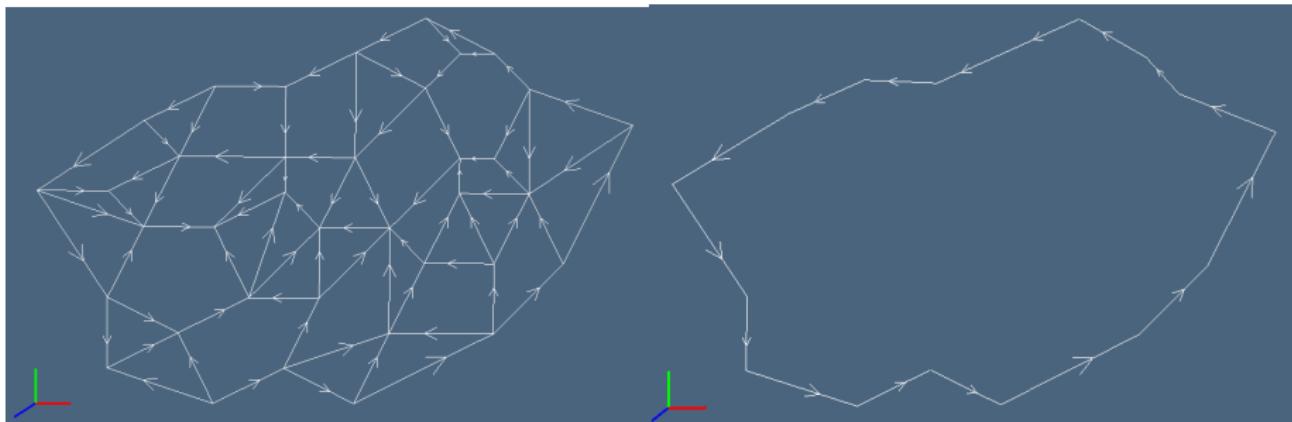
n -cells by
their vertices:

$[[0, 2, 4, 6],$
 $[1, 4, 5, 6],$
 $[1, 5, 6],$
 $[2, 5, 6]]$



The bare minimum data

with full awareness of topology



```
V = [[5,0],[7,1],[9,0],[13,2],[15,4],[17,8],[14,9],[13,10],[11,11],[9,10],[5,9],[7,
```

```
9],[3,8],[0,6],[2,3],[2,1],[8,3],[10,2],[13,4],[14,6],[13,7],[12,10],[11,9],[9,7],
```

```
[7,7],[4,7],[2,6],[3,5],[4,2],[6,3],[11,4],[12,6],[12,7],[10,5],[8,5],[7,6],[5,5]]
```

```
FV = [[0,1,16,28,29],[0,15,28],[1,2,17],[1,16,17,33],[2,3,17],[3,4,18,19],[3,17,18,30],[4,5,19],[5,6,19],[6,7,20,21,22,32],[6,19,20],[7,8,21],[8,9,21,22],[9,11,23,24],[9,22,23],[10,11,24,25],[10,12,25],[12,13,25,26],[13,14,27],[13,26,27],[14,15,28],[14,27,28,29,36],[16,29,34],[16,33,34],[17,30,33],[18,19,31],[18,30,31],[19,20,31,32],[22,23,32,33],[23,24,34,35],[23,33,34],[24,25,27,36],[24,35,36],[25,26,27],[29,34,35],[29,35,36],[30,31,32,33]]
```

Compressed Sparse Row (CSR) matrix storage

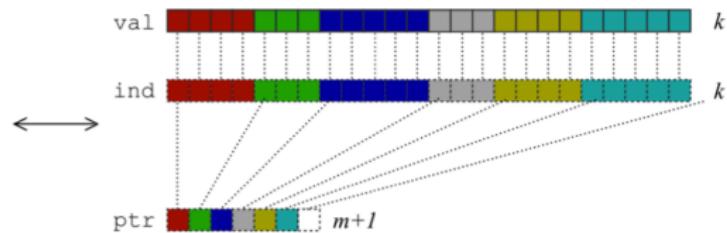
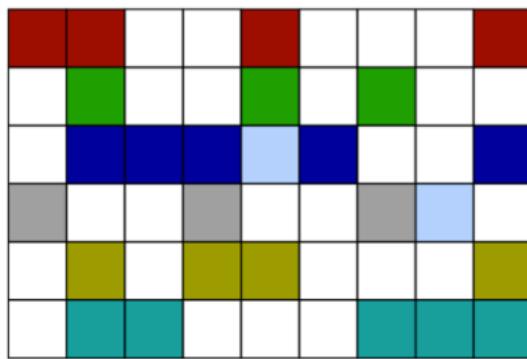
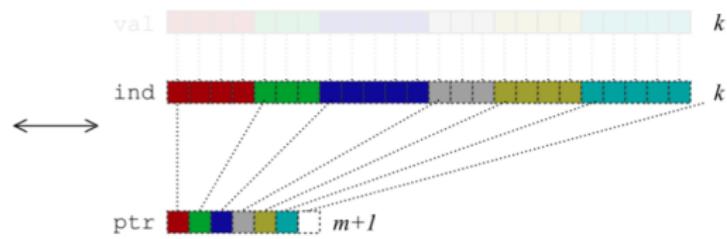
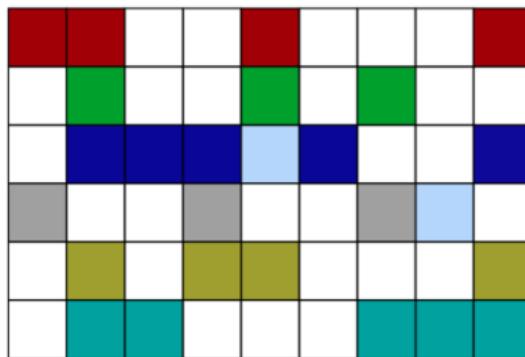


image from

Samuel Williams, Leonid Oliker, Richard Vuduc, John Shalf, Katherine Yelick, and James Demmel, [Optimization of sparse matrix-vector multiplication on emerging multicore platforms](#), Proceedings of the 2007 ACM/IEEE conference on Supercomputing (New York, NY, USA), SC '07, ACM, 2007, pp. 38:1–38:12.

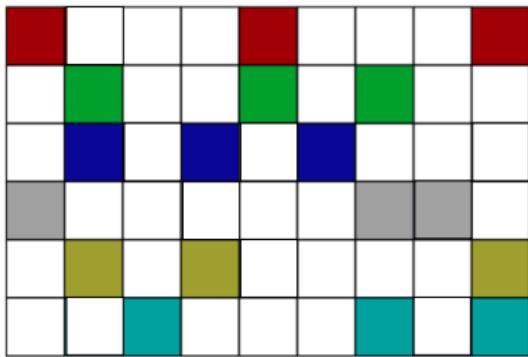
CSR storage of a **BINARY** matrix



of course, **non-zero values (all ones)** do not require storage

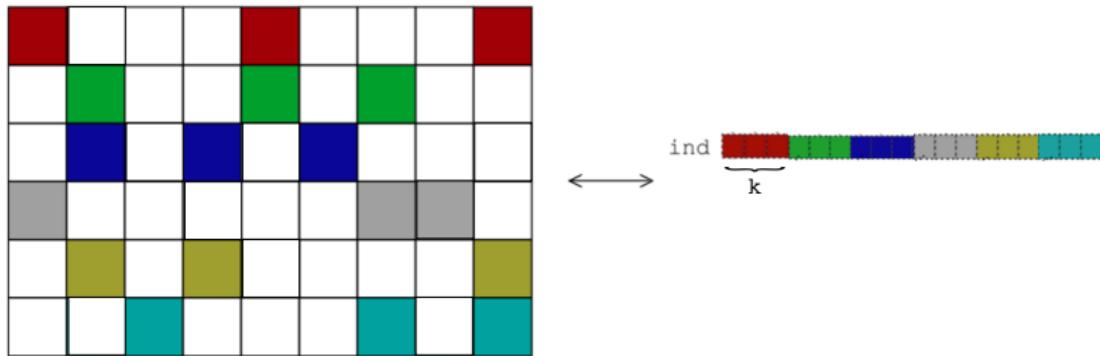
CSR storage of a binary matrix M_d such that

$$M_d \mathbf{1} = \mathbf{k}$$



CSR storage of a binary matrix M_d such that

$$M_d \mathbf{1} = \mathbf{k}$$



important cases:

- regular simplicial d -complexes: $k = d + 1$
 - regular cuboidal d -complexes: $k = 2^d$

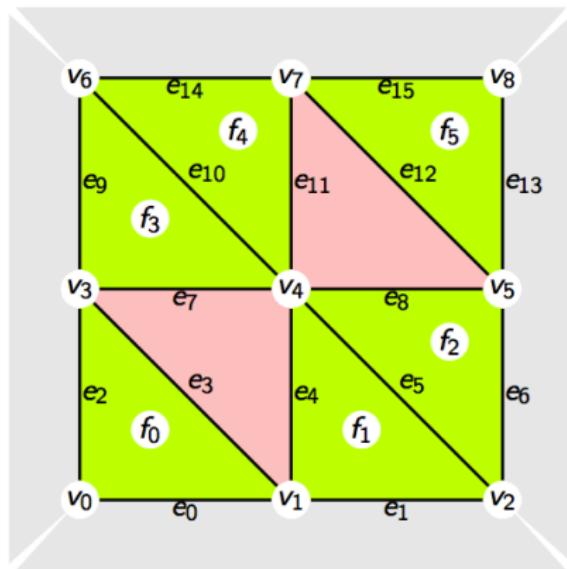
Operations

Facet extraction (1/5)

use the characteristic matrix M_d of a cellular partition Λ_d of \mathbb{E}^d , empty cells included

$$2D \Rightarrow d = 2$$

$$M_2 = \left(\begin{array}{ccccccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right)$$

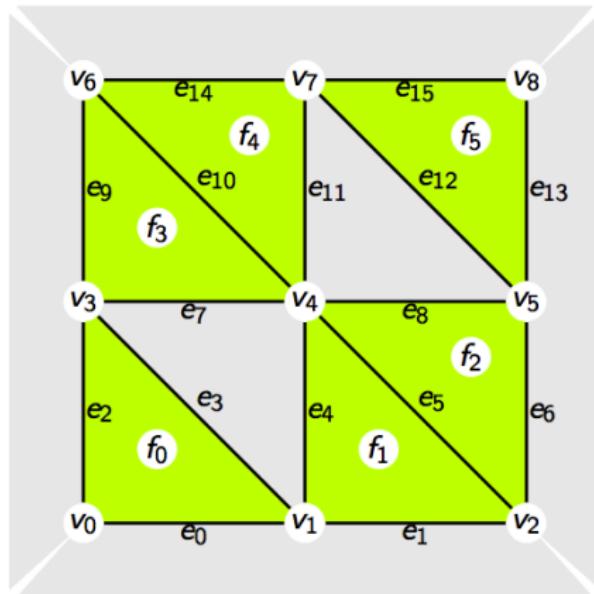


Facet extraction (2/5)

use the characteristic matrix M_d of a cellular partition Λ_d of \mathbb{E}^d , empty cells included

$$2D \Rightarrow d = 2$$

$$M_2 = \left(\begin{array}{ccccccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right)$$



Facet extraction (3/5)

use the characteristic matrix M_d of a cellular partition Λ_d of \mathbb{E}^d , empty cells included

$$A = M_2 M_2^t = \left(\begin{array}{cccccccccc} 3 & 1 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 2 & 2 & 0 \\ 3 & & 2 & 1 & 1 & 0 & 2 & 1 & 0 & 0 & 2 & 1 \\ & 3 & 1 & 1 & 1 & 1 & 2 & 0 & 0 & 1 & 1 & 2 \\ & & 3 & 2 & 0 & 0 & 0 & 1 & 2 & 2 & 2 & 1 \\ & & & 3 & 1 & 0 & 0 & 2 & 1 & 1 & 2 & 2 \\ & & & & 3 & 0 & 2 & 2 & 0 & 0 & 2 & 1 \\ & & & & & 3 & 1 & 0 & 1 & 1 & 0 & 0 \\ & & & & & & 3 & 1 & 0 & 0 & 1 & 1 \\ & & & & & & & 3 & 1 & 0 & 0 & 1 \\ & & & & & & & & 3 & 1 & 0 \\ & & & & & & & & & 3 & 1 \\ & & & & & & & & & & 3 \end{array} \right)$$

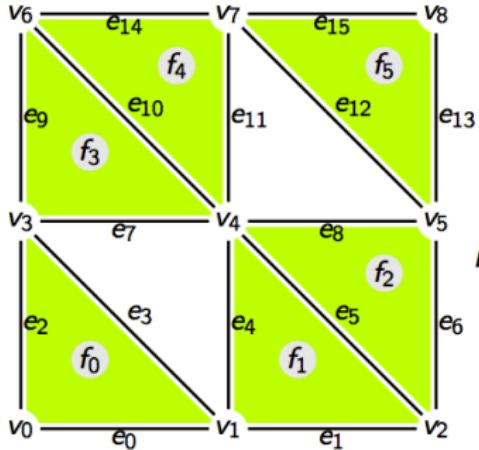
sym

$$\#(\lambda_d^i \cap \lambda_d^j) = A(i,j) \geq d \quad (i \neq j) \Rightarrow \exists \lambda_{d-1} = M_d(i) \wedge M_d(j)$$

Facet extraction (4/5)

use the characteristic matrix M_d of a cellular partition Λ_d of \mathbb{E}^d , empty cells included

$$A(0, 6) = 2 \Rightarrow (\text{110100000}) \wedge (\text{010110000}) = (\text{010100000}) \Rightarrow e = (v_1, v_3)$$

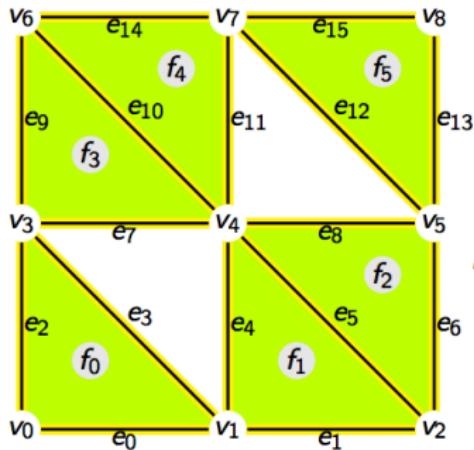


$$M_2 = \left(\begin{array}{c|c} \text{110100000} \\ \text{011010000} \\ \text{001011000} \\ \text{000110100} \\ \text{000010110} \\ \text{000001011} \\ \text{010110000} \\ \text{000011010} \\ \text{111000000} \\ \text{001001001} \\ \text{000000111} \\ \text{100100100} \end{array} \right) \Rightarrow M_1 = \left(\begin{array}{c|c} \text{110000000} \\ \text{100100000} \\ \text{011000000} \\ \text{010100000} \\ \text{010010000} \\ \text{001010000} \\ \text{001001000} \\ \text{000110000} \\ \text{000100100} \\ \text{000011000} \\ \text{000010100} \\ \text{000010010} \\ \text{000001010} \\ \text{000001001} \\ \text{000000110} \\ \text{000000011} \end{array} \right)$$

Facet extraction (5/5)

use the characteristic matrix M_d of a cellular partition Λ_d of \mathbb{E}^d , empty cells included

$$A(0, 6) = 2 \Rightarrow (\textcolor{blue}{110100000}) \wedge (\textcolor{gray}{010110000}) = (\textcolor{yellow}{010100000}) \Rightarrow e = (v_1, v_3)$$



$$M_2 = \left(\begin{array}{cccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right) \Rightarrow M_1 = \left(\begin{array}{cccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right)$$

Simplicial extrusion (1/3)

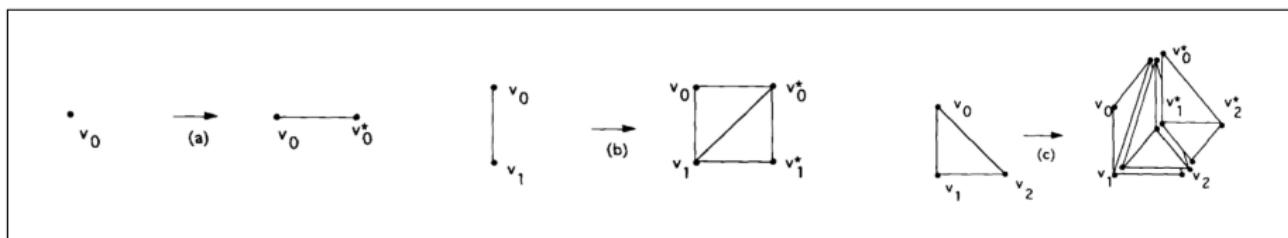
Optimal combinatorial algorithm

[Ferrucci & Paoluzzi, CAD, 1991]

for each d-simplex

$$\sigma^d = (v_0, v_1, \dots, v_d)$$

in the input complex $S(d)$, we generate combinatorially a chain of $d + 1$ **coherently-oriented** simplexes of dimension $d + 1$:



So we have, with $|\gamma^{d+1}| = \sigma^d \times I$, and $I = [0, 1]$:

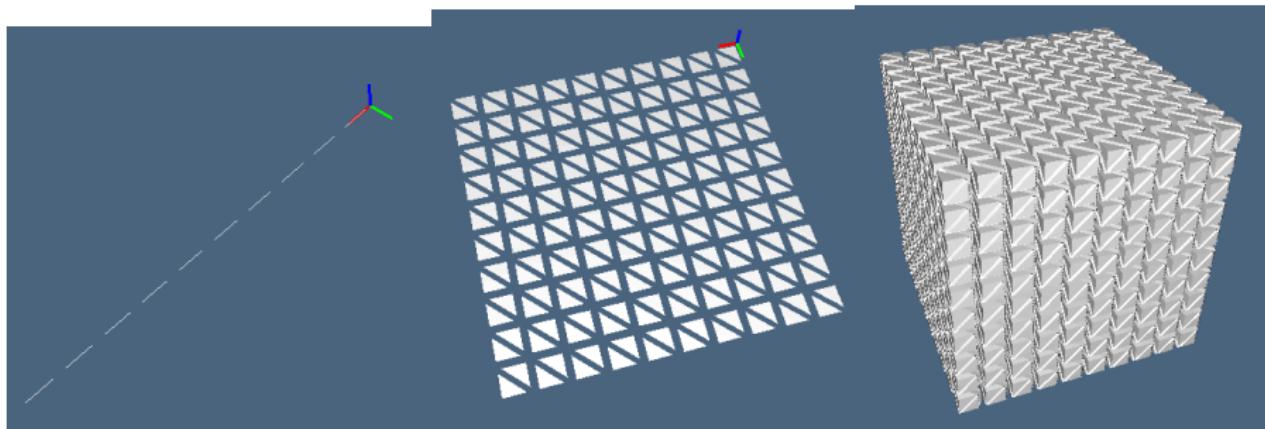
$$\gamma^{d+1} = \sum_{k=0}^d (-1)^{kd} \langle v_k, \dots, v_d, v_0^*, \dots, v_k^* \rangle$$

with $v_k \in \sigma^d \times \{0\}$ and $v_k^* \in \sigma^d \times \{1\}$,

Simplicial extrusion (2/3)

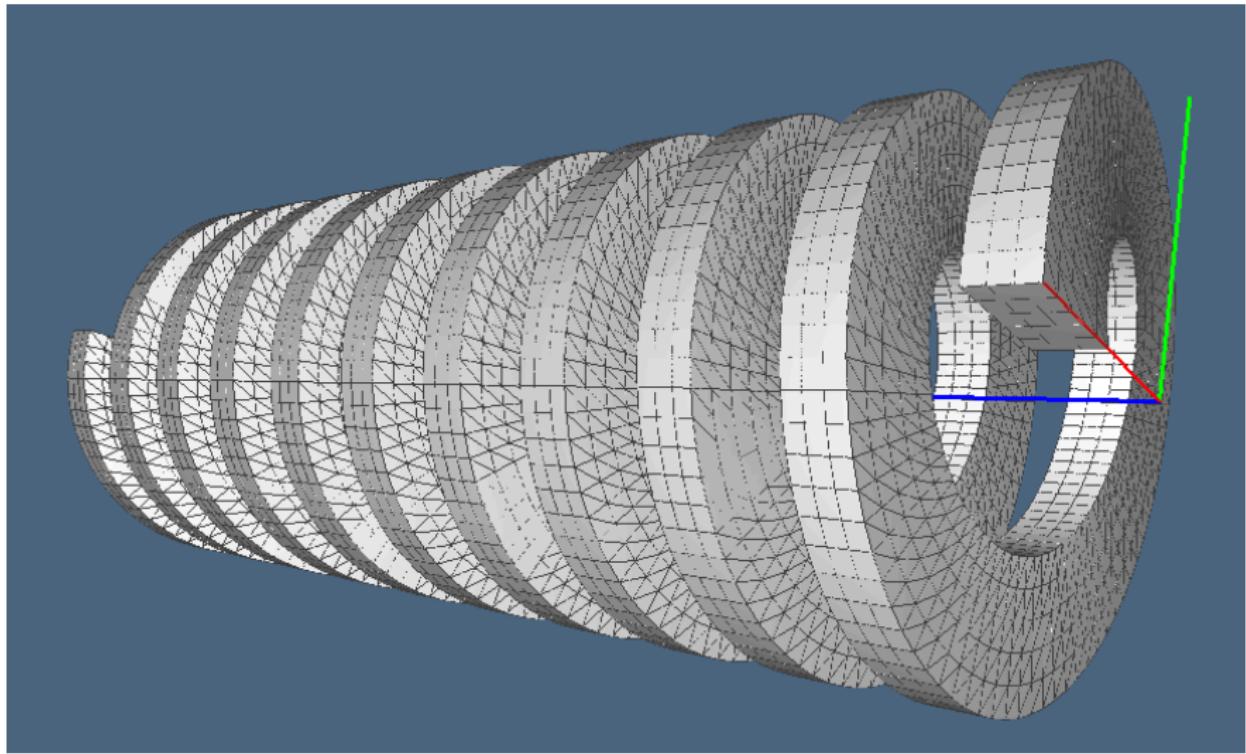
Let \mathfrak{S} be the set of simplicial c.c.c., and $S_d, S_1 \in \mathfrak{S}$:

$$S_d \times S_1 =: S_{d+1} \in \mathfrak{S}$$



```
model1 = larExtrude( VOID, 10*[1] )
model2 = larExtrude( model1, 10*[1] )
model3 = larExtrude( model2, 10*[1] )
```

Simplicial extrusion (3/3)



Cartesian product of complexes (1/4)

Let \mathfrak{C} be the set of c.c.c's. $X, Y \in \mathfrak{C} \Rightarrow X \times Y \in \mathfrak{C}$ [Basak, Geom. dedicata, 2010]

Let

$$X = S(m) \in \mathfrak{S}, \quad \text{and} \quad Y = S(n) \in \mathfrak{S};$$

Then

$$X \times Y = S(X \times Y) \in \mathfrak{S}$$

with

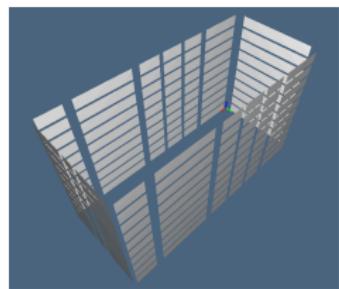
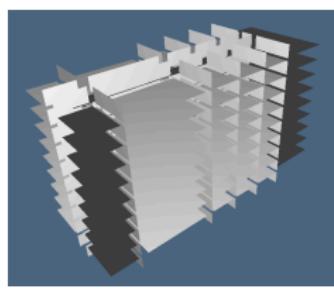
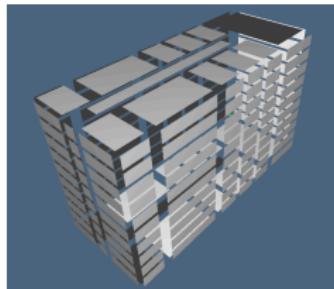
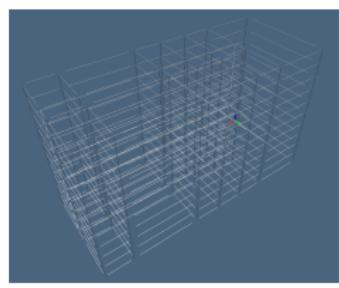
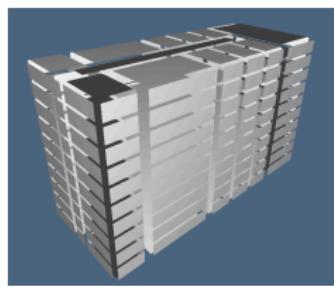
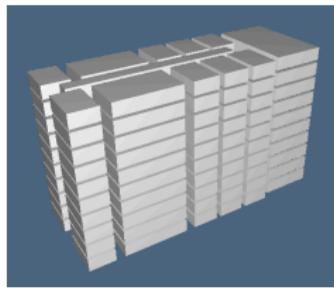
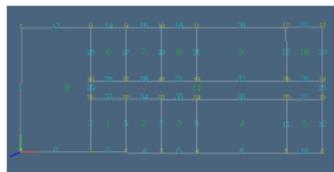
$$x \times y = (x_1y_1, \dots, x_1y_n, \dots, x_my_1, \dots, x_my_n) \in S(m+n).$$

Cartesian product of complexes (2/4)

Implementation

```
def larModelProduct(twoModels):
    (V, cells1), (W, cells2) = twoModels
    vertices = collections.OrderedDict(); k = 0
    for v in V:
        for w in W:
            id = tuple(v+w)
            if not vertices.has_key(id):
                vertices[id] = k
                k += 1
    cells = [ [vertices[tuple(V[v] + W[w])] for v in c1 for w in c2 ]
              for c1 in cells1 for c2 in cells2]
    model = [list(v) for v in vertices.keys()], cells
    return model
```

Cartesian product of complexes (4/4)



Larlib Examples ()

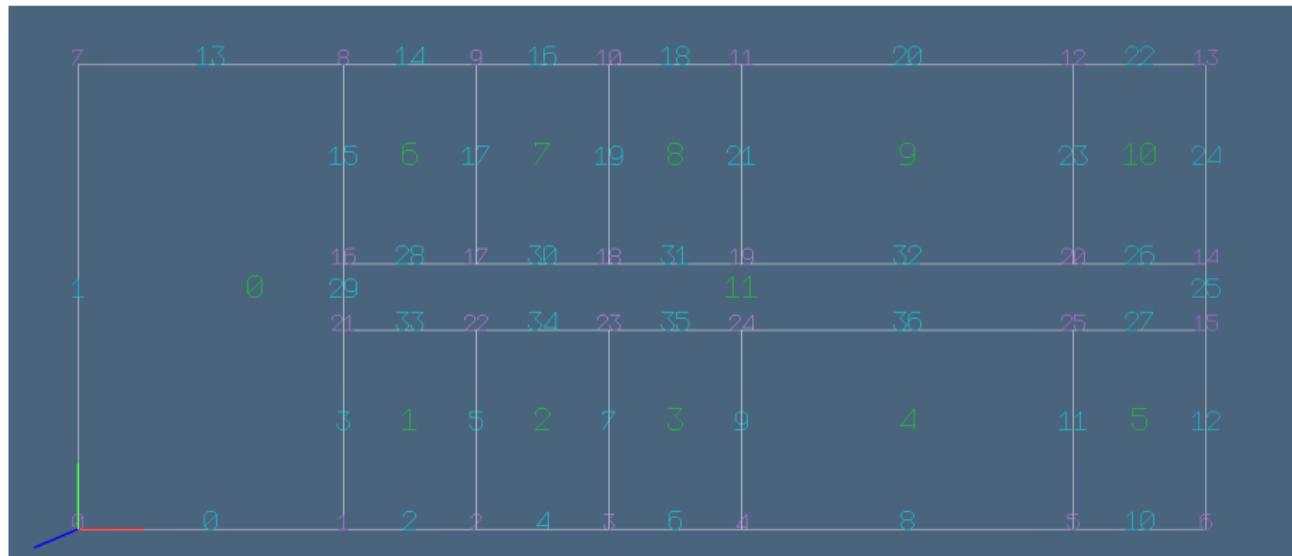
```
from larlib import *

V = [[0,0],[4,0],[6,0],[8,0],[10,0],[15,0],[17,0],[0,7],[4,7],[6,7],[8,7],[15,7],[17,7],[17,4],[17,3],[4,4],[6,4],[8,4],[10,4],[15,4],[4,3],[6,3],[8,10],[10,3],[15,3]]

FV = [[0,1,7,8,16,21],[1,2,21,22],[2,3,22,23],[3,4,23,24],[4,5,24,25],[5,6,25,26],[6,7,26,27],[7,8,27,28],[8,9,28,29],[9,10,29,30],[10,11,30,31],[11,12,31,32],[12,13,32,33],[13,14,33,34],[14,15,34,35],[15,16,35,36],[16,17,36,37],[17,18,37,38],[18,19,38,39],[19,20,39,40],[20,21,40,41],[21,22,41,42],[22,23,42,43],[23,24,43,44],[24,25,44,45],[25,26,45,46],[26,27,46,47],[27,28,47,48],[28,29,48,49],[29,30,49,50],[30,31,50,51],[31,32,51,52],[32,33,52,53],[33,34,53,54],[34,35,54,55],[35,36,55,56],[36,37,56,57],[37,38,57,58],[38,39,58,59],[39,40,59,60],[40,41,60,61],[41,42,61,62],[42,43,62,63],[43,44,63,64],[44,45,64,65],[45,46,65,66],[46,47,66,67],[47,48,67,68],[48,49,68,69],[49,50,69,70],[50,51,70,71],[51,52,71,72],[52,53,72,73],[53,54,73,74],[54,55,74,75],[55,56,75,76],[56,57,76,77],[57,58,77,78],[58,59,78,79],[59,60,79,80],[60,61,80,81],[61,62,81,82],[62,63,82,83],[63,64,83,84],[64,65,84,85],[65,66,85,86],[66,67,86,87],[67,68,87,88],[68,69,88,89],[69,70,89,90],[70,71,90,91],[71,72,91,92],[72,73,92,93],[73,74,93,94],[74,75,94,95],[75,76,95,96],[76,77,96,97],[77,78,97,98],[78,79,98,99],[79,80,99,100],[80,81,100,101],[81,82,101,102],[82,83,102,103],[83,84,103,104],[84,85,104,105],[85,86,105,106],[86,87,106,107],[87,88,107,108],[88,89,108,109],[89,90,109,110],[90,91,110,111],[91,92,111,112],[92,93,112,113],[93,94,113,114],[94,95,114,115],[95,96,115,116],[96,97,116,117],[97,98,117,118],[98,99,118,119],[99,100,119,120],[100,101,119,121],[101,102,121,122],[102,103,122,123],[103,104,123,124],[104,105,124,125],[105,106,125,126],[106,107,126,127],[107,108,127,128],[108,109,128,129],[109,110,129,130],[110,111,130,131],[111,112,131,132],[112,113,132,133],[113,114,133,134],[114,115,134,135],[115,116,135,136],[116,117,136,137],[117,118,137,138],[118,119,138,139],[119,120,139,140],[120,121,140,141],[121,122,141,142],[122,123,142,143],[123,124,143,144],[124,125,144,145],[125,126,145,146],[126,127,146,147],[127,128,147,148],[128,129,148,149],[129,130,149,150],[130,131,150,151],[131,132,151,152],[132,133,152,153],[133,134,153,154],[134,135,154,155],[135,136,155,156],[136,137,156,157],[137,138,157,158],[138,139,158,159],[139,140,159,160],[140,141,160,161],[141,142,161,162],[142,143,162,163],[143,144,163,164],[144,145,164,165],[145,146,165,166],[146,147,166,167],[147,148,167,168],[148,149,168,169],[149,150,169,170],[150,151,170,171],[151,152,171,172],[152,153,172,173],[153,154,173,174],[154,155,174,175],[155,156,175,176],[156,157,176,177],[157,158,177,178],[158,159,178,179],[159,160,179,180],[160,161,180,181],[161,162,181,182],[162,163,182,183],[163,164,183,184],[164,165,184,185],[165,166,185,186],[166,167,186,187],[167,168,187,188],[168,169,188,189],[169,170,189,190],[170,171,190,191],[171,172,191,192],[172,173,192,193],[173,174,193,194],[174,175,194,195],[175,176,195,196],[176,177,196,197],[177,178,197,198],[178,179,198,199],[179,180,199,200],[180,181,199,201],[181,182,201,202],[182,183,202,203],[183,184,203,204],[184,185,204,205],[185,186,205,206],[186,187,206,207],[187,188,207,208],[188,189,208,209],[189,190,209,210],[190,191,209,211],[191,192,211,212],[192,193,212,213],[193,194,213,214],[194,195,214,215],[195,196,215,216],[196,197,216,217],[197,198,217,218],[198,199,218,219],[199,200,219,220],[200,201,219,221],[201,202,221,222],[202,203,222,223],[203,204,223,224],[204,205,224,225],[205,206,225,226],[206,207,226,227],[207,208,227,228],[208,209,228,229],[209,210,229,230],[210,211,229,231],[211,212,231,232],[212,213,232,233],[213,214,233,234],[214,215,234,235],[215,216,235,236],[216,217,236,237],[217,218,237,238],[218,219,238,239],[219,220,239,240],[220,221,239,241],[221,222,241,242],[222,223,242,243],[223,224,243,244],[224,225,244,245],[225,226,245,246],[226,227,246,247],[227,228,247,248],[228,229,248,249],[229,230,249,250],[230,231,249,251],[231,232,251,252],[232,233,252,253],[233,234,253,254],[234,235,254,255],[235,236,255,256],[236,237,256,257],[237,238,257,258],[238,239,258,259],[239,240,259,260],[240,241,259,261],[241,242,261,262],[242,243,262,263],[243,244,263,264],[244,245,264,265],[245,246,265,266],[246,247,266,267],[247,248,267,268],[248,249,268,269],[249,250,269,270],[250,251,269,271],[251,252,271,272],[252,253,272,273],[253,254,273,274],[254,255,274,275],[255,256,275,276],[256,257,276,277],[257,258,277,278],[258,259,278,279],[259,260,279,280],[260,261,279,281],[261,262,281,282],[262,263,282,283],[263,264,283,284],[264,265,284,285],[265,266,285,286],[266,267,286,287],[267,268,287,288],[268,269,288,289],[269,270,289,290],[270,271,289,291],[271,272,291,292],[272,273,292,293],[273,274,293,294],[274,275,294,295],[275,276,295,296],[276,277,296,297],[277,278,297,298],[278,279,298,299],[279,280,299,300],[280,281,299,301],[281,282,301,302],[282,283,302,303],[283,284,303,304],[284,285,304,305],[285,286,305,306],[286,287,306,307],[287,288,307,308],[288,289,308,309],[289,290,309,310],[290,291,309,311],[291,292,311,312],[292,293,312,313],[293,294,313,314],[294,295,314,315],[295,296,315,316],[296,297,316,317],[297,298,317,318],[298,299,318,319],[299,300,319,320],[300,301,319,321],[301,302,321,322],[302,303,322,323],[303,304,323,324],[304,305,324,325],[305,306,325,326],[306,307,326,327],[307,308,327,328],[308,309,328,329],[309,310,329,330],[310,311,329,331],[311,312,331,332],[312,313,332,333],[313,314,333,334],[314,315,334,335],[315,316,335,336],[316,317,336,337],[317,318,337,338],[318,319,338,339],[319,320,339,340],[320,321,339,341],[321,322,341,342],[322,323,342,343],[323,324,343,344],[324,325,344,345],[325,326,345,346],[326,327,346,347],[327,328,347,348],[328,329,348,349],[329,330,349,350],[330,331,349,351],[331,332,351,352],[332,333,352,353],[333,334,353,354],[334,335,354,355],[335,336,355,356],[336,337,356,357],[337,338,357,358],[338,339,358,359],[339,340,359,360],[340,341,359,361],[341,342,361,362],[342,343,362,363],[343,344,363,364],[344,345,364,365],[345,346,365,366],[346,347,366,367],[347,348,367,368],[348,349,368,369],[349,350,369,370],[350,351,369,371],[351,352,371,372],[352,353,372,373],[353,354,373,374],[354,355,374,375],[355,356,375,376],[356,357,376,377],[357,358,377,378],[358,359,378,379],[359,360,379,380],[360,361,379,381],[361,362,381,382],[362,363,382,383],[363,364,383,384],[364,365,384,385],[365,366,385,386],[366,367,386,387],[367,368,387,388],[368,369,388,389],[369,370,389,390],[370,371,389,391],[371,372,391,392],[372,373,392,393],[373,374,393,394],[374,375,394,395],[375,376,395,396],[376,377,396,397],[377,378,397,398],[378,379,398,399],[379,380,399,400],[380,381,399,401],[381,382,401,402],[382,383,402,403],[383,384,403,404],[384,385,404,405],[385,386,405,406],[386,387,406,407],[387,388,407,408],[388,389,408,409],[389,390,409,410],[390,391,409,411],[391,392,411,412],[392,393,412,413],[393,394,413,414],[394,395,414,415],[395,396,415,416],[396,397,416,417],[397,398,417,418],[398,399,418,419],[399,400,419,420],[400,401,419,421],[401,402,421,422],[402,403,422,423],[403,404,423,424],[404,405,424,425],[405,406,425,426],[406,407,426,427],[407,408,427,428],[408,409,428,429],[409,410,429,430],[410,411,429,431],[411,412,431,432],[412,413,432,433],[413,414,433,434],[414,415,434,435],[415,416,435,436],[416,417,436,437],[417,418,437,438],[418,419,438,439],[419,420,439,440],[420,421,439,441],[421,422,441,442],[422,423,442,443],[423,424,443,444],[424,425,444,445],[425,426,445,446],[426,427,446,447],[427,428,447,448],[428,429,448,449],[429,430,449,450],[430,431,449,451],[431,432,451,452],[432,433,452,453],[433,434,453,454],[434,435,454,455],[435,436,455,456],[436,437,456,457],[437,438,457,458],[438,439,458,459],[439,440,459,460],[440,441,459,461],[441,442,461,462],[442,443,462,463],[443,444,463,464],[444,445,464,465],[445,446,465,466],[446,447,466,467],[447,448,467,468],[448,449,468,469],[449,450,469,470],[450,451,469,471],[451,452,471,472],[452,453,472,473],[453,454,473,474],[454,455,474,475],[455,456,475,476],[456,457,476,477],[457,458,477,478],[458,459,478,479],[459,460,479,480],[460,461,479,481],[461,462,481,482],[462,463,482,483],[463,464,483,484],[464,465,484,485],[465,466,485,486],[466,467,486,487],[467,468,487,488],[468,469,488,489],[469,470,489,490],[470,471,489,491],[471,472,491,492],[472,473,492,493],[473,474,493,494],[474,475,494,495],[475,476,495,496],[476,477,496,497],[477,478,497,498],[478,479,498,499],[479,480,499,500],[480,481,499,501],[481,482,501,502],[482,483,502,503],[483,484,503,504],[484,485,504,505],[485,486,505,506],[486,487,506,507],[487,488,507,508],[488,489,508,509],[489,490,509,510],[490,491,509,511],[491,492,511,512],[492,493,512,513],[493,494,513,514],[494,495,514,515],[495,496,515,516],[496,497,516,517],[497,498,517,518],[498,499,518,519],[499,500,519,520],[500,501,519,521],[501,502,521,522],[502,503,522,523],[503,504,523,524],[504,505,524,525],[505,506,525,526],[506,507,526,527],[507,508,527,528],[508,509,528,529],[509,510,529,530],[510,511,529,531],[511,512,531,532],[512,513,532,533],[513,514,533,534],[514,515,534,535],[515,516,535,536],[516,517,536,537],[517,518,537,538],[518,519,538,539],[519,520,539,540],[520,521,539,541],[521,522,541,542],[522,523,542,543],[523,524,543,544],[524,525,544,545],[525,526,545,546],[526,527,546,547],[527,528,547,548],[528,529,548,549],[529,530,549,550],[530,531,549,551],[531,532,551,552],[532,533,552,553],[533,534,553,554],[534,535,554,555],[535,536,555,556],[536,537,556,557],[537,538,557,558],[538,539,558,559],[539,540,559,560],[540,541,559,561],[541,542,561,562],[542,543,562,563],[543,544,563,564],[544,545,564,565],[545,546,565,566],[546,547,566,567],[547,548,567,568],[548,549,568,569],[549,550,569,570],[550,551,569,571],[551,552,571,572],[552,553,572,573],[553,554,573,574],[554,555,574,575],[555,556,575,576],[556,557,576,577],[557,558,577,578],[558,559,578,579],[559,560,579,580],[560,561,579,581],[561,562,581,582],[562,563,582,583],[563,564,583,584],[564,565,584,585],[565,566,585,586],[566,567,586,587],[567,568,587,588],[568,569,588,589],[569,570,589,590],[570,571,589,591],[571,572,591,592],[572,573,592,593],[573,574,593,594],[574,575,594,595],[575,576,595,596],[576,577,596,597],[577,578,597,598],[578,579,598,599],[579,580,599,600],[580,581,599,601],[581,582,601,602],[582,583,602,603],[583,584,603,604],[584,585,604,605],[585,586,605,606],[586,587,606,607],[587,588,607,608],[588,589,608,609],[589,590,609,610],[590,591,609,611],[591,592,611,612],[592,593,612,613],[593,594,613,614],[594,595,614,615],[595,596,615,616],[596,597,616,617],[597,598,617,618],[598,599,618,619],[599,600,619,620],[600,601,619,621],[601,602,621,622],[602,603,622,623],[603,604,623,624],[604,605,624,625],[605,606,625,626],[606,607,626,627],[607,608,627,628],[608,609,628,629],[609,610,629,630],[610,611,629,631],[611,612,631,632],[612,613,632,633],[613,614,633,634],[614,615,634,635],[615,616,635,636],[616,617,636,637],[617,618,637,638],[618,619,638,639],[619,620,639,640],[620,621,639,641],[621,622,641,642],[622,623,642,643],[623,624,643,644],[624,625,644,645],[625,626,645,646],[626,627,646,647],[627,628,647,648],[628,629,648,649],[629,630,649,650],[630,631,649,651],[631,632,651,652],[632,633,652,653],[633,634,653,654],[634,635,654,655],[635,636,655,656],[636,637,656,657],[637,638,657,658],[638,639,658,659],[639,640,659,660],[640,641,659,661],[641,642,661,662],[642,643,662,663],[643,644,663,664],[644,645,664,665],[645,646,665,666],[646,647,666,667],[647,648,667,668],[648,649,668,669],[649,650,669,670],[650,651,669,671],[651,652,671,672],[652,653,672,673],[653,654,673,674],[654,655,674,675],[655,656,675,676],[656,657,676,677],[657,658,677,678],[658,659,678,679],[659,660,679,680],[660,661,679,681],[661,662,681,682],[662,663,682,683],[663,664,683,684],[664,665,684,685],[665,666,685,686],[666,667,686,687],[667,668,687,688],[668,669,688,689],[669,670,689,690],[670,671,689,691],[671,672,691,692],[672,673,692,693],[673,674,693,694],[674,675,694,695],[675,676,695,696],[676,677,696,697],[677,678,697,698],[678,679,698,699],[679,680,699,700],[680,681,699,701],[681,682,701,702],[682,683,702,703],[683,684,703,704],[684,685,704,705],[685,686,705,706],[686,687,706,707],[687,688,707,708],[688,689,708,709],[689,690,709,710],[690,691,709,711],[691,692,711,712],[692,693,712,713],[693,694,713,714],[694,695,714,715],[695,696,715,716],[696,697,716,717],[697,698,717,718],[698,699,718,719],[699,700,719,720],[700,701,719,721],[701,702,721,722],[702,703,722,723],[703,704,723,724],[704,705,724,725],[705,706,725,726],[706,707,726,727],[707,708,727,728],[708,709,728,729],[709,710,729,730],[710,711,729,731],[711,712,731,732],[712,713,732,733],[713,714,733,734],[714,715,734,735],[715,716,735,736],[716,717,736,737],[717,718,737,738],[718,719,738,739],[719,720,739,740],[720,721,739,741],[721,722,741,742],[722,723,742,743],[723,724,743,744],[724,725,744,745],[725,726,745,746],[726,727,746,747],[727,728,747,748],[728,729,748,749],[729,730,749,750],[730,731,749,751],[731,732,751,752],[732,733,752,753],[733,734,753,754],[734,735,754,755],[735,736,755,756],[736,737,756,757],[737,738,757,758],[738,739,758,759],[739,740,759,760],[740,741,759,761],[741,742,761,762],[742,743,762,763],[743,744,763,764],[744,745,764,765],[745,746,765,766],[746,747,766,767],[747,748,767,768],[748,749,768,769],[749,750,769,770],[750,751,769,771],[751,752,771,772],[752,753,772,773],[753,754,773,774],[754,755,774,775],[755,756,775,776],[756,757,776,777],[757,758,777,778],[758,759,778,779],[759,760,779,780],[760,761,779,781],[761,762,781,782],[762,763,782,783],[763,764,783,784],[764,765,784,785],[765,766,785,786],[766,767,786,787],[767,768,787,788],[768,769,788,789],[769,770,789,790],[770,771,789,791],[771,772,791,792],[772,773,792,793],[773,774,793,794],[774,775,794,795],[775,776,795,796],[776,777,796,797],[777,778,797,798],[778,779,798,799],[779,780,799,800}]]
```

Larlib Examples (1/8)

```
submodel = SKEL_1(STRUCT(MKPOLS((V,FV))))  
V, EV = larFacets((V,FV),2,4)  
VV = AA(LIST)(range(len(V)))  
VIEW(larModelNumbering(1,1,1)(V,[VV,EV,FV[:-4]],submodel,3))
```



Larlib Examples (2/8)

```
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V,FV[:-4]))+MKPOLS((V,EV))+AA(MK)(V)))
```

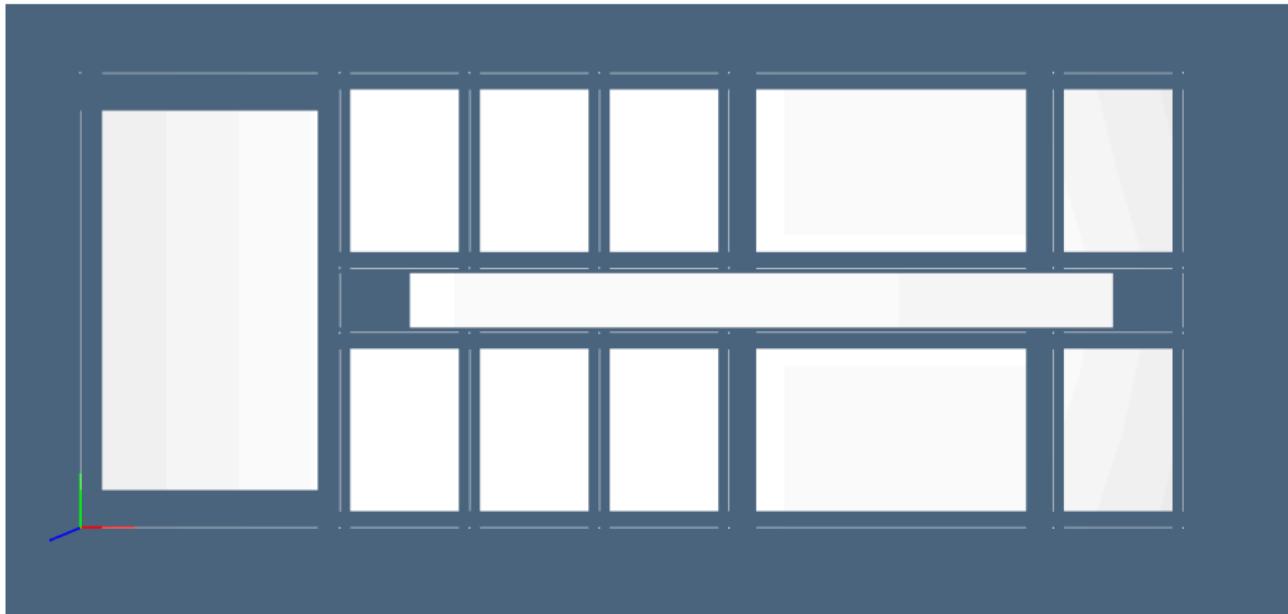


Figure 2:

Larlib Examples (3/8)

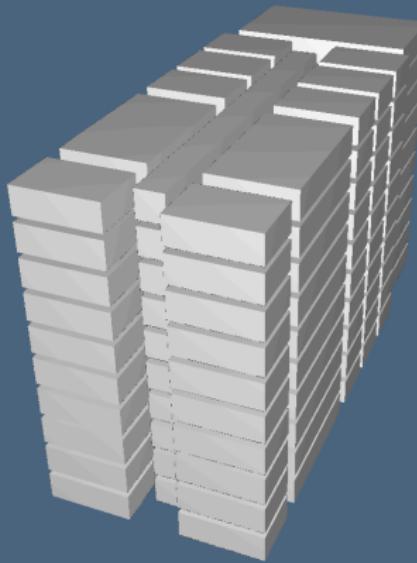
```
BE = boundaryCells(FV[:-4],EV)
VIEW(EXPLODE(1.2,1.2,1)(MKPOL((V,[EV[e] for e in BE]))))
```



Figure 3:

Larlib Examples (4/8)

```
C1 = larCuboids([10])
V3,CV = larModelProduct([(V,FV[:-4]),C1])
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,CV))))
```



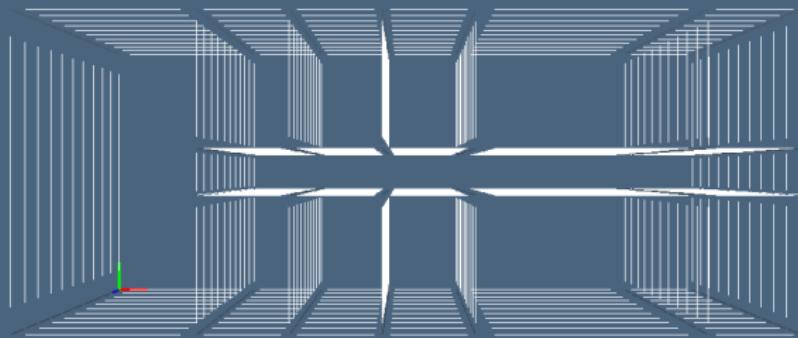
Larlib Examples (5/8)

```
full3D = CV + exteriorCells((V3,CV))
V3,FV3 = larFacets((V3,full3D),3,6)
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,FV3))))
```



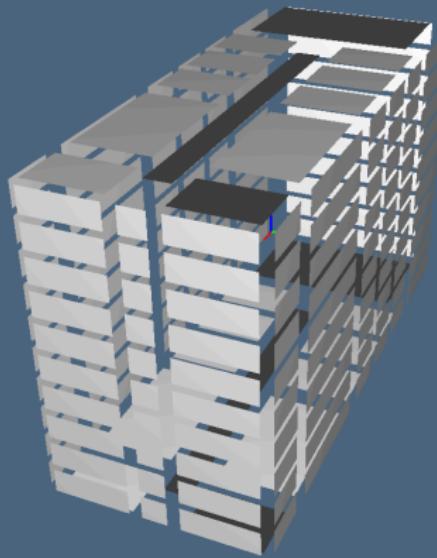
Larlib Examples (6/8)

```
V3,EV3 = larFacets((V3,FV3),2)
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,EV3))))
```



Larlib Examples (7/8)

```
VV3 = AA(LIST)(range(len(V3)))
boundary3D = boundaryCells(CV,FV3)
VIEW(EXPLODE(1.25,1.25,1.25)(MKPOL((V3,[FV3[f] for f in boundary3D]))))
```



Larlib Examples (8/8)

```
V3,F3 = larFacets((V3,CV))
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,F3))))
```

