Computing homological information based on directed graphs within discrete objects

Aldo González Lorenzo[1,2], Jean-Luc Mari[1], Alexandra Bac[1]
Pedro Real[2]

[1]Aix-Marseille University, CNRS, LSIS UMR 7296 (France)

[2]University of Seville, Department of Applied Mathematics I (Spain)

September 22, 2014

Outline

Outline

Outline

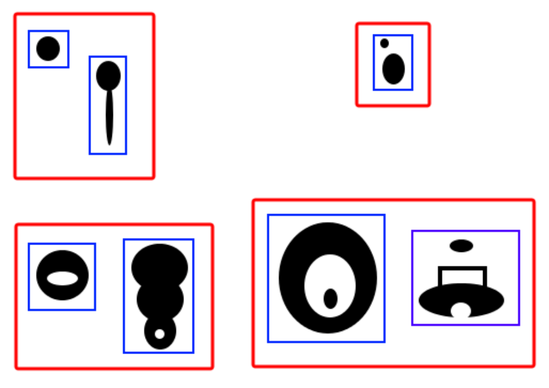
Outline

Objective: to analyse 2D images or 3D volumes, to find properties for establishing equivalences.

Approach: homology. Branch of topology considering "holes".

Example

We can use the number of holes in order to classify images.



Two objects *equivalents* will be in the same class, and two objects in different classes will be *non-equivalents*.

Example

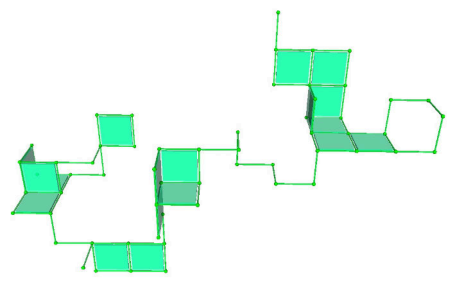


Figure : One connected component and two 1-dimensional holes.

Example

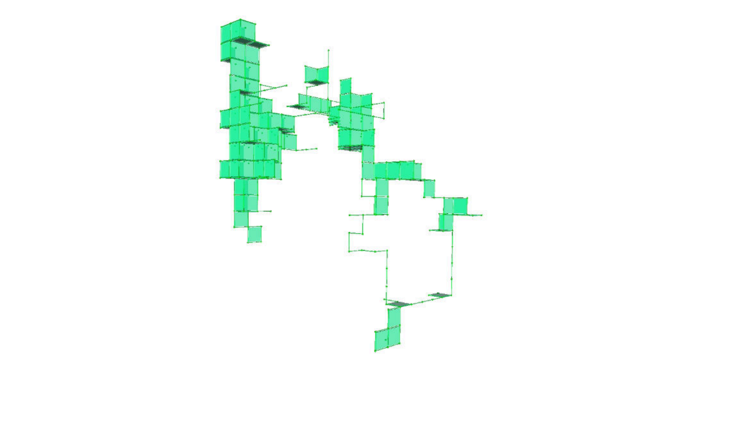


Figure : Here it is more difficult.

How can we do it ?

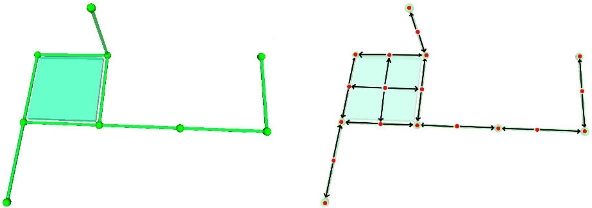
- Step 1: To have a binary image (2D, 3D, etc);
- Step 2: To build a cubical complex from it, chosing one adjacency relationship;
- Step 3: To compute the homology of this complex.

We will see all this in detail.

Outline

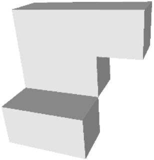
- In order to compute homology, we restrict ourselves to one kind of topological space: cubical complexes
- Intuitively, it is an object made of the union of points, lines, squares, cubes, . . ., glued by their boundaries
- In other words: the intersection of two pieces (*cells*) is empty or another piece

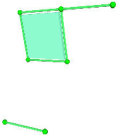
- Every cubical complex has an associated directed graph (*digraph*) called *Hasse diagram*
- Vertices represent the cells
- Arcs go from one cell to the cells in its border



Discrete object → cubical complexes

From each n-dimensional discrete object, we can build a cubical complex w.r.t the $3^n - 1$ or the $2n$-adjacency relationship.





Homology

- Homology is defined on **chain complexes**. It is an algebraic object that formalizes the *boundary* of a cell
- The elements are *chains*, linear combinations of cells
- The boundary of a cell $d(\sigma)$ is the sum of the cells in the boundary of σ with their respective coefficients

There are two important classes of chains:

- $Z = \ker(d)$, the chains whose boundary is 0 (*cycles*)
- $B = im(d)$, the chains that are the boundary of another chain (*boundaries*)

Homology groups consists of the elements of Z/B, this is the cycles that are not boundaries.

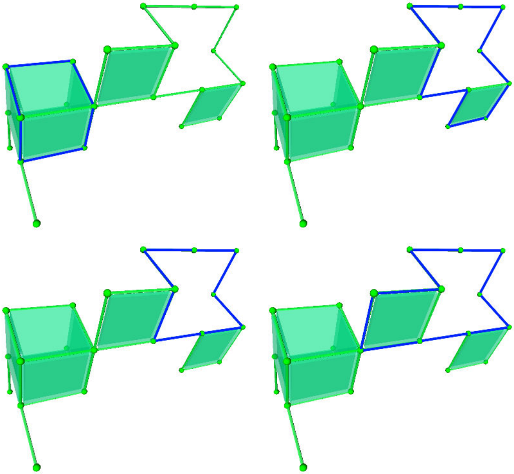


Figure : Some cycles.

- When computing the homology of an object, we want to find a basis (set of *homology generators*) of each homology group
- The size of this basis is the *Betti number* (for dimension $n \leq 3$). It is the number of independent "holes"

Let's remark that this basis is not unique.

Outline

- The Effective Homology theory introduces the concept of *reduction*
- It gives a relation between the original chain complex and another one, equivalent and "smaller"
- Briefly, it consists of three maps between chains (f, g, h) satisfying several properties

Outline

- Introduced by Robin Forman in the 90s
- Discrete version of the Morse theory, well-developed and used for computing the homology in the continuous context
- It gives an upper bound of the Betti numbers without dealing with algebraic objects

Definition

A *Discrete Gradient Vector Field* (DGVF) is a matching on the Hasse diagram of a cell complex such that there are not "directed cycles".

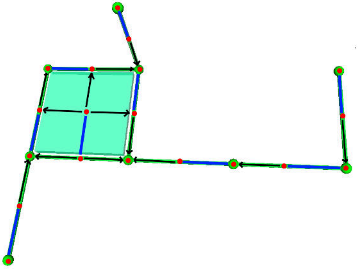


Figure : Left: A matching. Right: Not a Matching

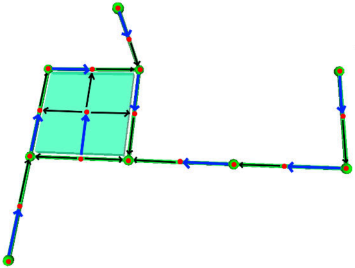


Figure : Left: A DGVF. Right: Not a DGVF

Given a DGVF, a cell is *critical* if it is unmatched.

Theorem

For each $q \geq 0$, the q-th Betti number is less than the number of critical q-cells.

DGVF $\rightarrow$ reduction

We can establish a reduction from a DGVF:

$$\begin{aligned} h(\sigma) &= \sum_{k \geq 0} V(1 - dV)^k(\sigma) & &= V(\sigma) + h(1 - dV)(\sigma) \\ f(\sigma) &= (1 - dh - hd)(\sigma) & &= f(1 - dV)(\sigma) \\ g(\sigma) &= \sigma \end{aligned}$$

where

$$V(\sigma) = \begin{cases} \langle d(\tau), \sigma \rangle \cdot \tau, & (\sigma, \tau) \text{ belongs to the matching} \\ 0, & \text{if not} \end{cases}$$

($\langle d(\tau), \sigma \rangle$ being the coefficient of the cell σ in the chain $d(\tau)$)

Outline

1. Discrete object $\rightarrow$ cubical complex;

1. Discrete object $\rightarrow$ cubical complex;
2. Initial DGVF;

1. Discrete object $\rightarrow$ cubical complex;
2. Initial DGVF;
3. Iterative correction: at each time, we erase 2 critical cells;

1. Discrete object $\rightarrow$ cubical complex;
2. Initial DGVF;
3. Iterative correction: at each time, we erase 2 critical cells;
4. Homology generators: we obtain them by the reduction.

Step 1: the complex

From a **discrete object** (voxels set), we build the **cubical complex** w.r.t the 2n-adjacency

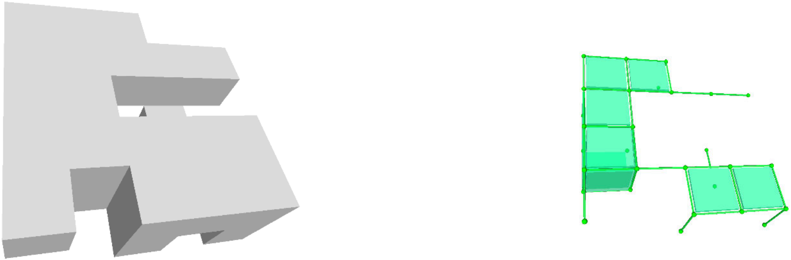
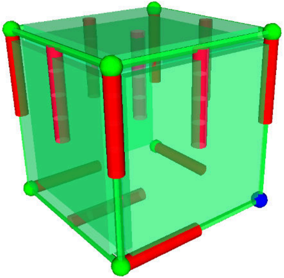


Figure : Left: Discrete object. Right: Its associated cubical complex

Step 2: initial DGVF

- We establish any DGVF
- There are several methods. We use the parallel method



- Typically, there are too many critical cells

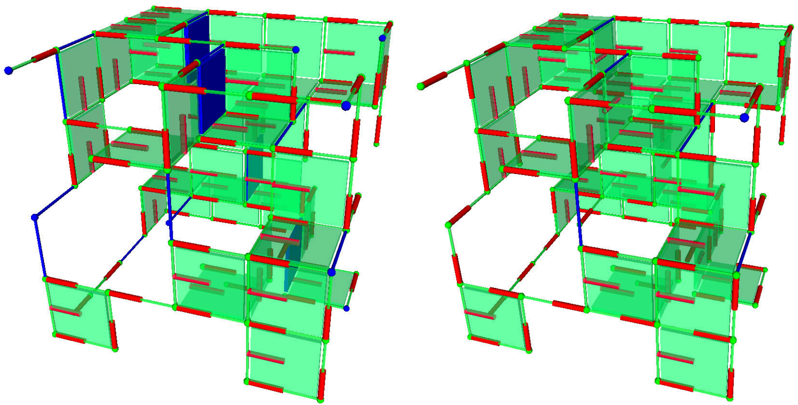


Figure : Left: optimal DGVF. Right: not an optimal DGVF

Step 3: iterative correction

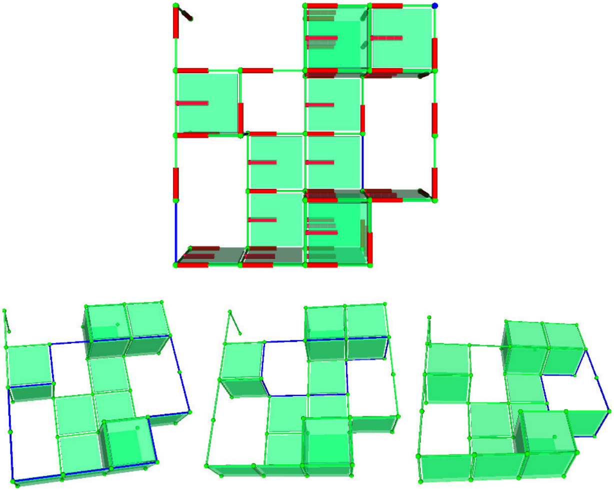
At each iteration,

- We choose a critical cell σ;
- We compute $fd(\sigma)$. We choose a critical cell τ found during this computation;
- We reverse the path from σ to τ.

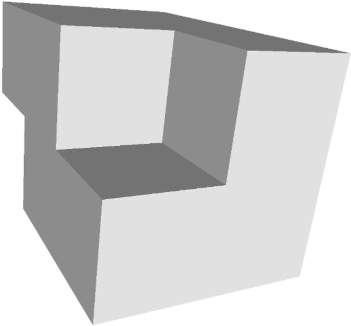
At the end, the number of critical cells equals the Betti numbers

Step 4: homology generators

We only have to compute $f = 1 - dh - hd$ over the critical cells

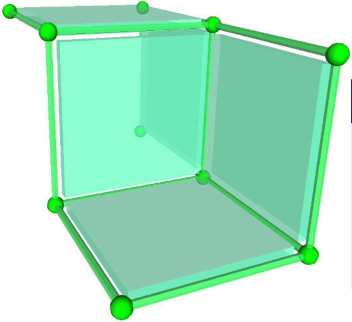


Outline



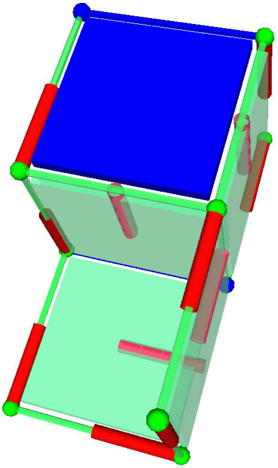
Discrete object

10 voxels



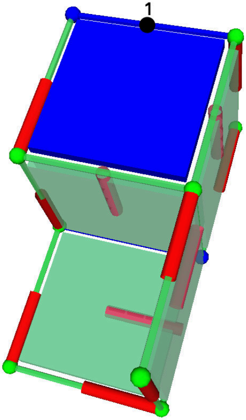
Cubical complex

- 10 0-cells
- 14 1-cells
- 5 2-cells



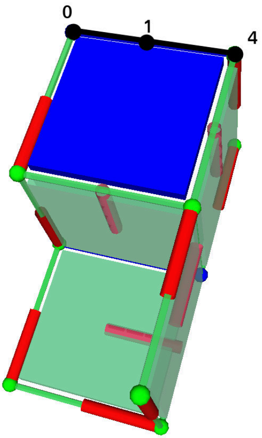
Initial DGVF

- 2 critical 0-cells
- 2 critical 1-cells
- 1 critical 2-cells



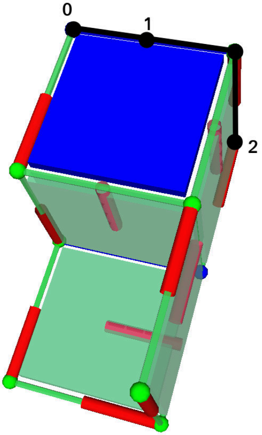
Correction of the critical cell **1**

We compute $fd(1) =$



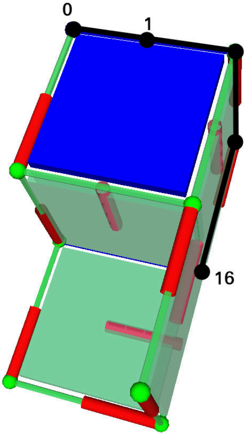
Correction of the critical cell **1**

We compute $fd(1) =$
$f(-0+4) = -f(0) + f(4)$



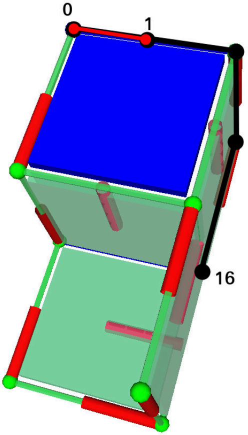
Correction of the critical cell **1**

We compute $fd(1) =$
$= -f(0) + f(2)$



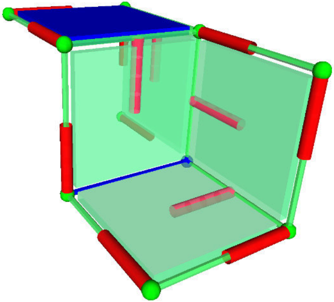
Correction of the critical cell **1**

We compute $fd(1) =$
$= -f(0) + f(16)$



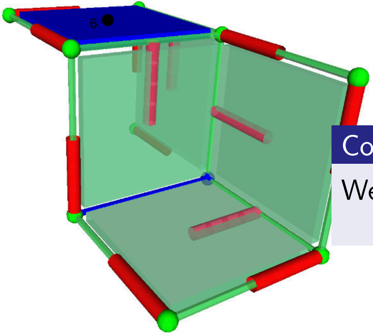
Correction of the critical cell **1**

We compute $fd(1) =$
$= -f(0) + f(16)$
We reverse the path from **1** to **0**



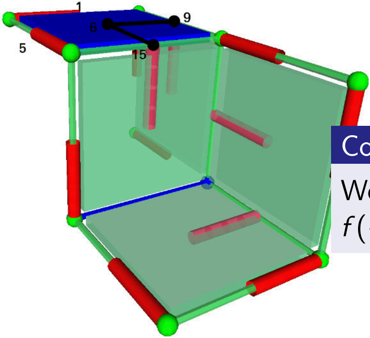
Correction of the critical cell **1**

We compute $fd(1) =$
$= -f(0) + f(16)$
We reverse the path from **1** to **0**



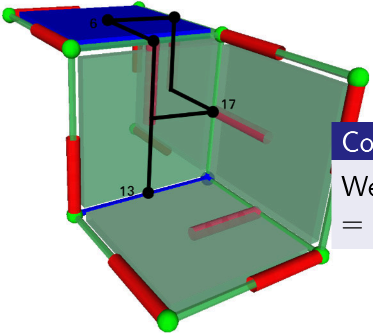
Correction of the critical cell 6

We compute $fd(6) =$



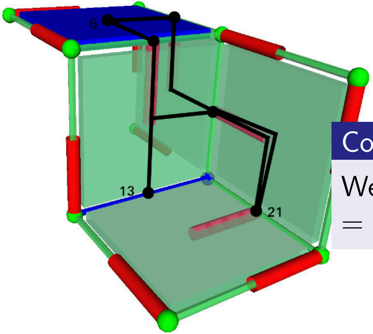
Correction of the critical cell **6**

We compute $fd(6) =$
$f(-1+5-9+15) = -f(9)+f(15)$



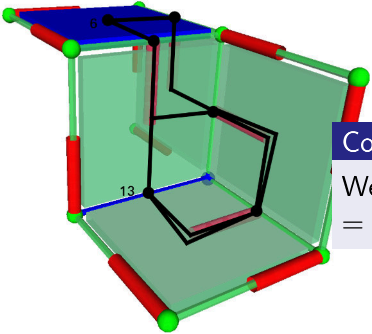
Correction of the critical cell **6**

We compute $fd(6) =$
$= -f(17) + f(17) - f(13)$



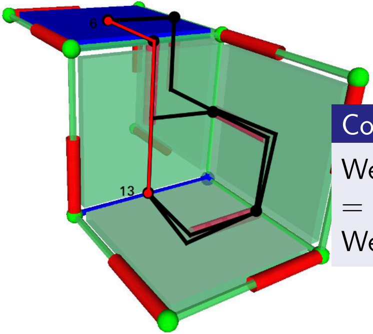
Correction of the critical cell 6

We compute $fd(6) =$
$= -f(21) + f(21) - f(13)$



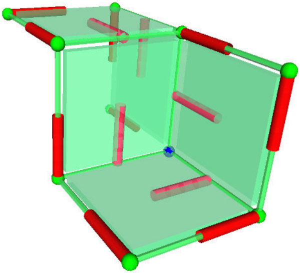
Correction of the critical cell 6

We compute $fd(6) =$
$= -f(13) + f(13) - f(13) = f(13)$



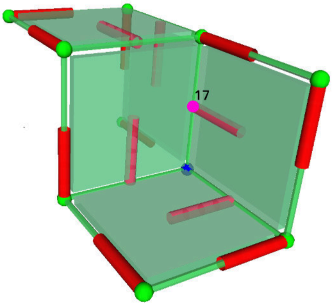
Correction of the critical cell **6**

We compute $fd(6) =$
$= -f(13) + f(13) - f(13) = f(13)$
We reverse the path from **6** to **13**



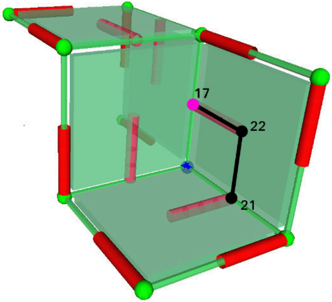
Correction of the critical cell **6**

We compute $fd(6) =$
$= -f(13) + f(13) - f(13) = f(13)$
We reverse the path from **6** to **13**



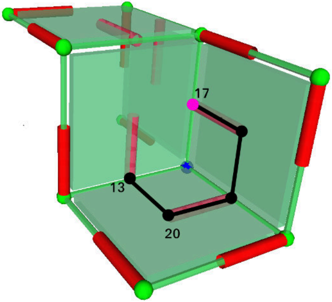
Computing h on the *confluence vertices*

We compute $h(17) =$



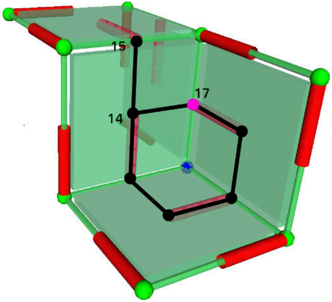
Computing *h* on the *confluence vertices*

We compute $h(17) =$
$V(17) + h(1 - dV)(17) = -22 + h(21)$



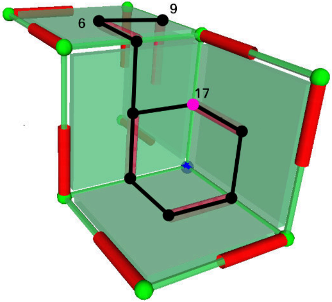
Computing h on the *confluence vertices*

We compute $h(17) =$
$= -22 - 20 - h(13)$



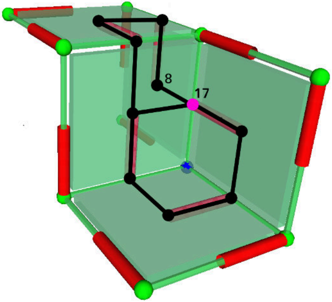
Computing *h* on the *confluence vertices*

We compute $h(17) =$
$= -22 - 20 - 14 - h(15) - h(17)$



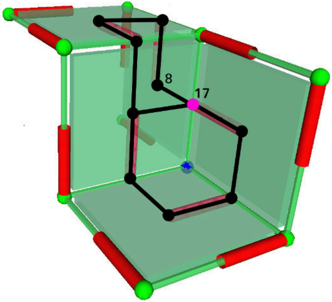
Computing h on the *confluence vertices*

We compute $h(17) =$
$= -22 - 20 - 14 - 6 - h(9) - h(17)$



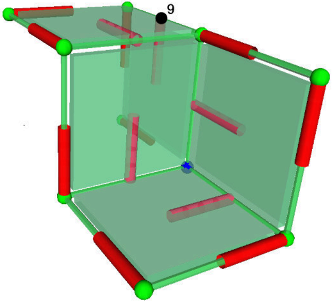
Computing h on the *confluence vertices*

We compute $h(17) =$
$= -22 - 20 - 14 - 6 + 8 + h(17) - h(17)$



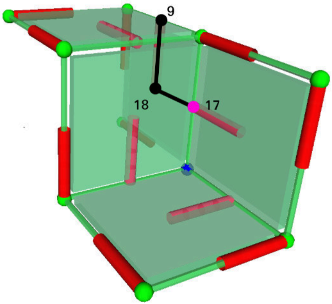
Computing h on the *confluence vertices*

We compute $h(17) =$
$= -22 - 20 - 14 - 6 + 8 + h(17) - h(17)$
We will substitute
$h(17) = -22 - 20 - 14 - 6 + 8$



Example

We compute $h(9) =$



Example

We compute $h(9) =$

$$= -18 + h(17)$$

$$= -18 - 22 - 20 - 14 - 6 + 8$$

- Absolute control of the homological information (thanks to the reduction);
- Non-redundant representation of the reduction;
- Integer coefficients, any dimension.

Future works

- Minimize the number of confluence vertices;
- Beautiful generators.

Thanks. Questions?