# Applications of Forman's discrete Morse theory to topology visualization and mesh compression

THOMAS LEWINER, HÉLIO LOPES AND GEOVAN TAVARES

Department of Mathematics — Pontifícia Universidade Católica — Rio de Janeiro — Brazil
{tomlew, lopes, tavares}@mat.puc--rio.br.

**Abstract.** Morse theory is a powerful tool for investigating the topology of smooth manifolds. It has been widely used by the computational topology, computer graphics and geometric modeling communities to devise topology based algorithms and data structures. Forman introduced a discrete version of this theory, which is purely combinatorial.

This work aims to build, visualize and apply the basic elements of Forman's discrete Morse theory. It intends to use some of those concepts to visually study the topology of an object. As a basis, an algorithmic construction of optimal Forman's discrete gradient vector fields is provided. This construction is then used to topologically analyze mesh compression schemes, such as Edgebreaker and Grow&Fold. In particular, this paper proves that the complexity class of the strategy optimization of Grow&Fold is MAX–SNP hard.

**Keywords:** ***Discrete Mathematics. Hypergraphs. Data compaction and compression. Computer Graphics. Computer–aided design. Morse Theory. Forman Theory. Vector Field Visualization. Computational Topology.***

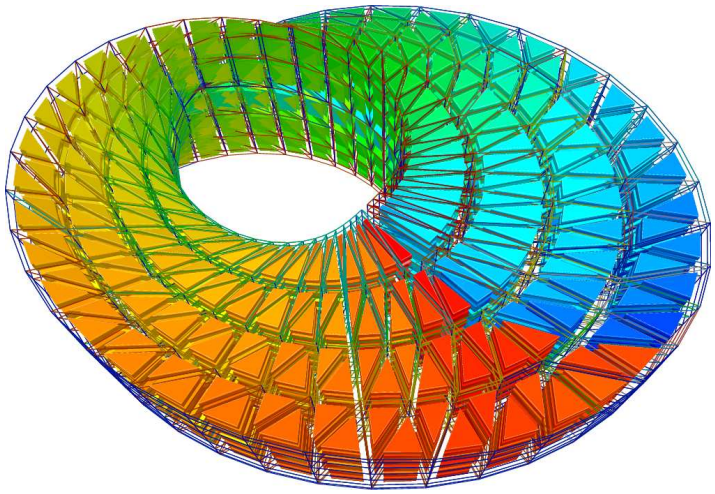


**Figure 1:** *The energy of a discrete gradient vector field on a 3–manifold cell complex (projection on $\mathbb{R}^3$).*

## 1 Introduction

Morse theory [16] is a fundamental tool for investigating the topology of smooth manifolds. Many applications of this theory to computer graphics have been devised [18, 8]. In the new field of computational topology [4, 2], Morse theory has been used to design topology based algorithms and data structures [13]. This work focuses on a similar tool for discrete structures such as cell complexes (see Figure 1), based on Forman's extension of Morse theory [5, 6].

In this work, we aim to visually investigate topological aspects of a geometric or abstract model. This paper also develops to mesh compression. To do so, we first introduce an algorithm to construct an optimal discrete gradient vector field on general, $n$–dimensional discrete structures, where optimality entails having the minimum possible number of critical elements. This algorithm illustrates the structure of a discrete gradient vector field [10]. Second, we provide a construction of such discrete gradient vector fields on triangulated surfaces using compression strategy such as the Edgebreaker [17]. Finally, we use Forman's theory together with this construction to analyze the Grow&Fold algorithm [19] for solid mesh compression.

The paper is organized as follows. In section 2 *Forman's discrete Morse theory*, we introduce some basics of Forman's theory. In section 3 *Hypergraphs and Hyperforests* we define some concepts of hypergraph theory, which are slightly dif-

Preprint MAT. 18/02, communicated on Dec. $3^{rd}$, 2002 to the Department of Mathematics, Pontifícia Universidade Católica — Rio de Janeiro, Brazil. The corresponding work was published in Transactions on Visualization and Computer Graphics, volume 10, number 5, pp. 499–508. IEEE Press, 2004.

ferent from the classical ones [1]. In section 4 *Algorithm*, we introduce our algorithm to build gradient fields, trying to reach optimality. Reaching the optimum in the general case is MAX–SNP hard [10], since it reduces to the vertex cover problem. A MAX–SNP hard problem is an NP–Hard problem for which any polynomial approximation can be arbitrarily far from the optimal. However, our algorithm builds optimal discrete gradient vector fields for the case of 2–manifolds [11], and almost always achieves the minimum number of critical cells in quadratic time for the general case (non–manifold and higher dimensions) [12]. We will illustrate some applications to topology visualization in the section 5 *Topology Visualization*. Finally we apply this theoretical framework to mesh compression in section 6 *Applications to mesh compression*.

## 2 Forman's discrete Morse theory

Classical Morse theory [16] is usually considered as a bridge between geometry and topology. In particular, it justifies the intuition that an object with a complex topology must have a complex geometry. More formally, for any smooth function defined on a smooth manifold, the number and the nature of its critical points (i.e. the points where the gradient vanishes) are strongly related to the topology of the manifold. Usually, the smooth function is devised from the geometry of the embedding of the manifold, as the vertical projection of Figure 2.

Robin Forman extended this theory to discrete structures, namely cell complexes (see Section 2(a)). The resulting theory is more powerful and rigorous than a simple discretization. In particular, a cell complex with a discrete gradient vector field $V$ is simple homotopy equivalent to a complex composed of only the critical cells of $V$. However, this theory requires some further concepts that will be briefly introduced in this section. For a complete presentation of Forman's theory, see [5, 6].

### (a) Cell Complexes

A cell complex is, roughly speaking, a generalization of the structures used to represent solid models: it is a consistent collection of cells (vertices, edges, faces…). A complete introduction to cell complexes can be found in [15].

A *cell* $\alpha^{(p)}$ of dimension $p$ is a set homeomorphic to the open $p$–ball $\{x \in \mathbb{R}^p : \|x\| < 1\}$. When the dimension $p$ of the cell is obvious, we will denote $\alpha$ instead of $\alpha^{(p)}$.

A *cell complex* $K$ is built by starting off with a discrete collection of 0–cells (vertices) called $K^0$, then attaching 1–cells (edges) to $K^0$ along their boundaries, obtaining $K^1$, then attaching 2–cells (faces) to $K^1$ along their boundaries, and so on, obtaining spaces $K^n$ for each $n$.

A cell complex will be said to be finite when it is built out of a finite number of cells. In this work, we will consider only finite (and therefore regular) cell complexes.

A $p$–cell $\alpha^{(p)}$ is called a sub–face of a $q$–cell $\beta^{(q)}$ $(p < q)$ if $\alpha \subset closure\,(\beta)$. If $q = p + 1$, we will use the notation $\alpha^{(p)} \prec \beta^{(q)}$, and say that $\alpha$ and $\beta$ are *incident*. We call the boundary of a $p$–cell the union of cells with lower dimension incident to it.

### (b) Smooth interpretation of discrete Morse theory

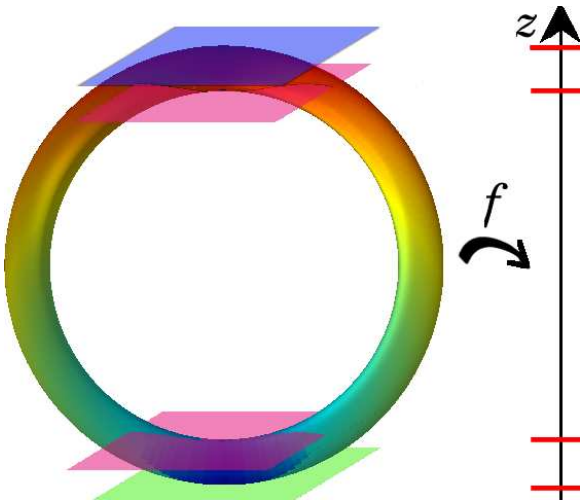



**Figure 2:** *A smooth manifold (torus) with a Morse function ($f(x, y, z) = z$) defined on it. There are 4 critical points: where the tangent plane is horizontal ($\nabla f = 0$).*

In this section, we will try to introduce discrete Morse theory to the reader familiar with the classical notions of vector fields and its critical points. Consider a smooth manifold $M$ of dimension $n$ with a smooth function $f$ defined on it, such as the torus of Figure 2. A point $x$ of $f$ is critical if the gradient of $f$ vanishes at $x$: $\nabla f\,(x) = 0$. When second derivative $H_f\,(x)$ is non–degenerated, a critical point can be:

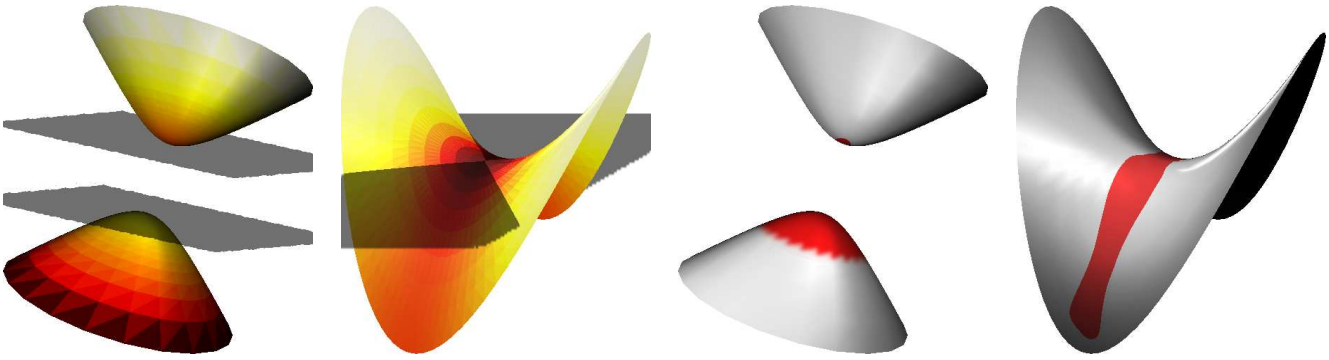
- a local minimum: $H_f\,(x)$ has only positive eigenvalues (see Figure 3(a)).
- a local maximum: $H_f\,(x)$ has only negative eigenvalues (see Figure 3(a)).
- a $k$–saddle: $H_f\,(x)$ has $k$ negative eigenvalues and $(n - k)$ positive ones (see Figure 3(b)).

A small region around a critical point will be deformed by the flow of $-\nabla f$ (in a similar way to the trajectory of water drops on $M$ under the force $-\nabla f$). This deformation depends of the nature of the critical point:

- around a local minimum, the region retracts to a single point, i.e. a 0–cell (see Figure 3(c)).
- around a local maximum, the region will cover a small region of $M$ (a $n$–cell) (see Figure 3(c)).
- around a $k$–saddle, the region will retract in some directions, and expand in the other ones, deforming to a "$k$–curve", i.e. a $k$–cell (see Figure 3(d)).

Those small deformed regions will be the cells of the discretization of $M$ (see Section 2(f)).

However in computer graphics, the problem is often formulated as the opposite one: given a discrete object, how to



(a) Local minimum and maximum (b) Saddle (c) Retraction near the minimum, expansion near the maximum (d) The region around the saddle tends to a curve

**Figure 3:** *Deformations around different kinds of critical points.*

define its smooth analog. In the case of Morse theory, a gradient field indicates how small regions, i.e. cells, will merge during the deformation induced by its flow. A discrete gradient vector field can be seen as the indication of those local merges. Successive merges cannot loop, and thus they form a kind of tree, rooted at the critical cell. Our algorithm is based on this observation, although the kind of tree is a quite complex object in the general case (see section 3 *Hypergraphs and Hyperforests*).

### (c) Discrete gradient vector fields

Forman's theory relies on admissible functions on a cell complex, or equivalently their gradient vector field. Here, we will introduce the theory from the second point of view. Nevertheless, our construction of a vector field also builds the corresponding discrete Morse function. We will define those notions in the following paragraphs. The analogy of the discrete vector field with the flow mentioned above is based on the merges of small regions, i.e. cells, under the flow derived from a smooth Morse. Those local merge operations can be represented by pairings, which is the general definition of discrete vector fields:

**Definition 1 (Combinatorial vector field)** *A* combinatorial vector field $V$*, defined on a cell complex* $K$*, is a disjoint collection of pairs* $\{\alpha^{(p)}, \beta^{(p+1)}\}$ *of incident cells:* $\alpha^{(p)} \prec \beta^{(p+1)}$.
*For such pairs,* $V(\alpha) = \beta$ *and* $V(\beta) = 0$*. If a cell* $\sigma$ *does not belong to any pair, then* $V(\sigma) = 0$.

We will represent this pairing by an arrow from $\alpha^{(p)}$ to $\beta^{(p+1)}$.

A *closed* $V$*–path* is an alternate sequence of $p$– and ($p$+1)–cells $\alpha_0, \beta_0, \ldots, \alpha_1, \beta_1, \alpha_{r+1} = \alpha_0$, with $r > 0$, satisfying:

$$V(\alpha_i^{(p)}) = \beta_i^{(p+1)} \quad \text{and} \quad \beta_i^{(p+1)} \succ \alpha_{i+1}^{(p)} \neq \alpha_i^{(p)}.$$

**Definition 2 (Discrete gradient vector field)** *A combinatorial vector field* $V$ *will be called a* discrete gradient vector field *if it contains no closed* $V$*–path.*

### (d) A simple example

In the example of Figure 4, the discrete gradient vector field $V$ is represented by arrows, from a cell of the complex to its image by $V$: from an edge to a face, and from a vertex to an edge. There is, among others, an opened $V$–path starting at the bottom–right vertex, and ending at the topmost one, and another one starting at the bottom green edge, and ending at the top horizontal red edge.

The corresponding Hasse diagram (Figure 5) represents every cell by one node. The faces (2–cells) are aligned on top rank, the edges (1–cells) on the middle one and the vertices (0–cells) on the bottom rank. A link between two nodes symbolizes that the corresponding cells are incident. We linked by an orange line paired cells.

### (e) Critical cells and optimality

Morse proved that the topology of a manifold is related to its critical elements. Forman gave an analogous result, with the following definition for the critical cells.

**Definition 3 (Critical cells)** *A cell* $\alpha$ *is* critical *if it is not paired with any other cell, i.e.:*

$$V(\alpha) = 0 \qquad \text{and} \qquad \alpha \notin Im(V)$$

On Figure 4 and Figure 5, critical cells are drawn in red: there is one critical vertex (topmost one) and one critical edge (top horizontal edge).

**Definition 4 (Optimality)** *A discrete gradient vector field is* optimal *if it has the minimum possible number of critical cells.*

The number of critical cells is not a topological invariant, since it depends on the discrete gradient vector field defined. For example, an empty discrete vector field (i.e. no cells are paired) would have all its cells critical. However, the minimal possible number of critical cells is a topological invariant for $n$–manifolds, $n < 4$ [12]. Reaching the optimum in the general case is MAX–SNP hard [10]. However, our algorithm builds optimal gradient vector field for the case of

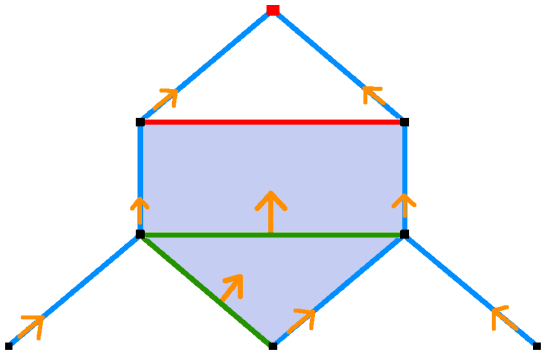


**Figure 4:** *A cell complex with its discrete gradient field.*

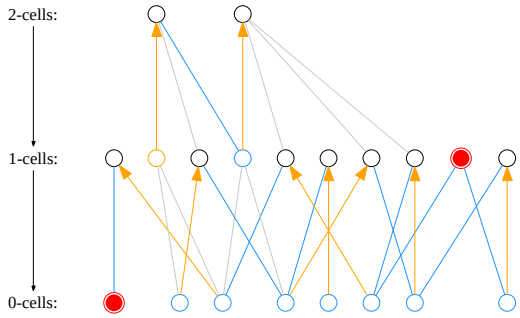
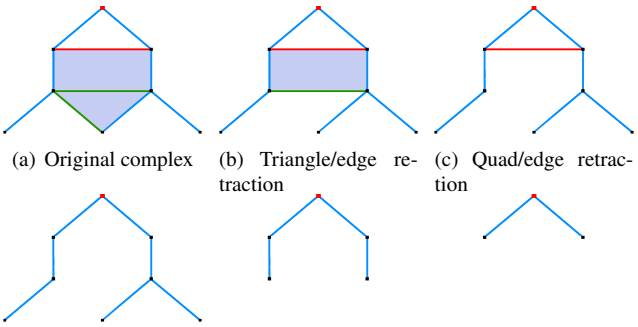



**Figure 5:** *The Hasse diagram of Figure 4.*

2–manifolds [11] and almost always results in the minimum number of critical cells in quadratic time for the general case [12].

### (f) Homotopy properties

Forman proved that a cell complex with a discrete gradient vector field $V$ is *simple homotopy equivalent* [3] to a complex built with exactly one cell for each critical element of $V$. This corresponds to cutting a differentiable manifold at different heights, as in classical Morse theory [16], or to the deformation described in Section 2(b).



(a) Original complex (b) Triangle/edge retraction (c) Quad/edge retraction

(d) Critical edge removal (e) Vertex/edge retraction (f) Last steps until the critical vertex

**Figure 6:** *The collapse scheme of the example of Figure 4.*

In the example of Figure 4, there is one critical vertex and one critical edge: the corresponding complex has the homotopy type of a circle. This simple homotopy equivalence is built out of a sequence of collapses, where each step collapses a pair of cells $\{\alpha, V(\alpha)\}$, as illustrated on Figure 6. The ordering of the collapses form a particular kind of tree that will be introduced in the next section.

## 3 Hypergraphs and Hyperforests

In the triangulation of a solid, an edge is generally incident to more than two faces. Therefore, the dual graph whose nodes are the triangles of the triangulation, and whose links join triangles that share an edge would not be a simple graph: such links can have more than two end nodes. We will thus need to generalize the notion of graphs to *hypergraphs*. A complete introduction to hypergraphs can be found in [1].

### (a) Oriented hypergraphs

**Definition 5 (Hypergraph)** *A* hypergraph *is a pair* $(N, L)$. *$N$ is the set of nodes. Each element of the family $L$ is a family of nodes, and is called* hyperlinks.

The elements of $L$ are families, which means that a hyperlink can be incident more than once to a node. They can also be empty. We will classify hyperlinks into the *regular hyperlinks* (or shortly, *links*), which join two distinct nodes as in ordinary graphs, the *loops* which are incident to only one node, and the *non–regular hyperlinks*, which either are multiply incident to one node or join three or more nodes.

We will give a hypergraph an orientation by distinguishing one node of each hyperlink as its *source*.

**Definition 6 (Regular components)** *The* regular components *of a hypergraph $(N, L)$ are the connected components of the graph $(N, R)$, where $R$ is the set of the regular hyperlinks of $L$.*

### (b) Hyperforests

**Definition 7 (Hypercircuit)** *An* oriented hypercircuit *in a hypergraph is a sequence of distinct nodes $n_0, n_1, \ldots, n_{r+1}$ such that $n_{r+1} = n_0$ and, for all $0 \leq i \leq r$, $n_i$ is the source of a hyperlink leading to $n_{i+1}$.*

**Definition 8 (Hyperforest)** *We will say that an oriented hypergraph is a* hyperforest *if each node is the source of at most one hyperlink, and if it does not contain any hypercircuit.*

For example, Figure 7 a hyperforest extracted from the triangle/edge hypergraph of a double cube.

**Proposition 9** *Let $HF$ be a hyperforest, and $R$ one of its regular components.*

*(i) $R$ is a tree.*

*(ii) There is at most one node in $R$ which is the source of either a loop or a non–regular hyperlink.*

The demonstration of this proposition can be found in [12].

The corresponding work was published in Transactions on Visualization and Computer Graphics, volume 10, number 5, pp. 499–508. IEEE Press, 2004.

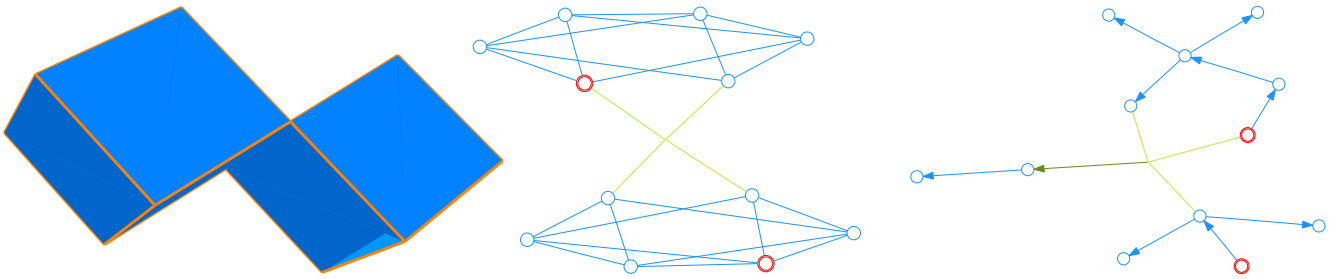


**Figure 7:** *A double cube made with 12 squared–faces, 23 edges and 14 vertices. Its dual hypergraph has 12 nodes and 23 links, and admits a hyperforest with 10 hyperlinks.*

## 4 Algorithm

This section introduces an algorithm to define a discrete gradient vector field for a given cell complex.

The algorithm is optimal for surfaces [11], in the sense that it minimizes the number of critical cells. But the general case (non–manifold, higher dimension) has been proved to be MAX–SNP hard.

For the sake of clarity, the description of the algorithm will begin by the simplest part, although it is the last one of the process.

### (a) Outline

Let us consider a finite cell complex $K$ of dimension $n$. The algorithm consists in the following two steps:

1. We first select some ($n$-1)–cells of $K$ guaranteeing that the hypergraph whose nodes are all the $n$–cells of $K$ and whose hyperlinks represent those selected ($n$-1)–cells will be a hyperforest. We then define the vector field for the selected cells by orienting this hyperforest (see Section 4(c)).
   The unselected ($n$-1)–cells of $K$ together with the $p$–cells, for $p < n - 1$, form again a complex $K^{n-1}$, of dimension at most ($n$-1). We repeat this selection on $K^{n-1}$ until the unselected cells form a complex $K^1$ of dimension 1, i.e. a graph.

2. We finally build the vector field on that graph $K^1$ by orienting it. This step is the simplest one, and will be introduced first in Section 4(b).

The algorithm optimality relies on the hyperlink selection, which is explained in section 4(d) *Selecting cells of the hypergraph*. The complexity of this selection is quadratic in the worst case. The other parts of the algorithm have a linear complexity.

Working again on the example of Figure 4, we see on Figure 8 and Figure 9 the two steps of the algorithm. During the first step, the vector field is defined on a dual tree containing all faces. The unpaired vertices and edges form another cell complex, actually a graph. During the second and last step, the vector field is defined on this graph.

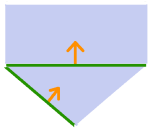


**Figure 8:** *First step: selecting faces and edges to form a hyperforest.*

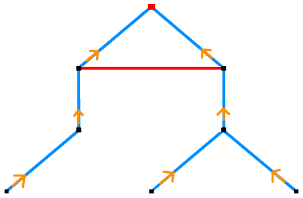


**Figure 9:** *Last step: processing the remaining vertex/edge graph.*

### (b) Last step: construction on graphs

We know from the topology of a graph that any graph is homotopy equivalent to a node with loops. To match the homotopy property of Section 2(f), we have to pair every node except one, leaving unpaired one edge per loop.

To do so, we build a spanning tree [9] of each connected component of the graph. All the links which are not in the tree will remain unpaired, and will thus be critical. We then choose a root node $r$ for the tree, which will also be left unpaired. We pair every link of the spanning tree to its incident node further to the root. We have thus paired all the nodes and links of the spanning tree, except the root $r$.

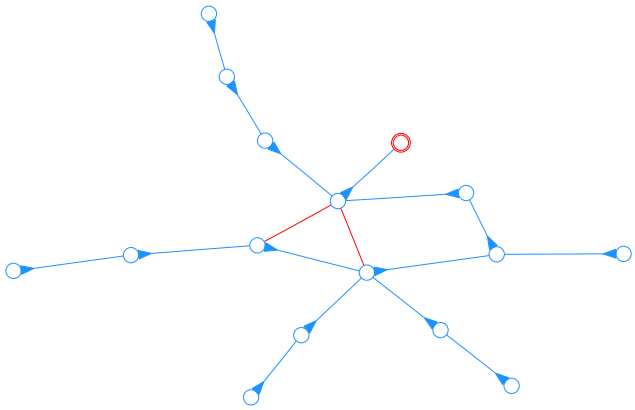


**Figure 10:** *Last step: the graph remaining after processing a homological sphere.*

As the tree contains no cycle, the resulting vector field is

admissible as a discrete gradient vector field. On Figure 10, we can visualize the vector field on a graph resulting of the process of a model of Poincaré's homological sphere [7]. There are two cycles in this graph. In each cycle there is an edge that remained unpaired and, consequently, critical. The root node of the graph also remained unpaired, and thus critical.

**(c) First steps: construction on hyperforests**

The spanning tree of the last step had its nodes representing the vertices of $K$ and its links representing some edges of $K$. The following is an extension of the last procedure to hyperforest. We will now consider a hyperforest $(N, L)$ extracted from $K$. Its nodes will represent the $p$–cells of $K$. A hyperlink representing a ($p$-1)–cell $\sigma$ of $K$ will join the nodes corresponding to the $p$–cells incident to $\sigma$. On the contrary of the previous case, the vector field will pair a node to a hyperlink.

The next section presents a procedure to select the hyperlinks in such a way that $(N, L)$ will be a hyperforest. This selection, and the induced orientation introduce below, are based on the equivalent of a critical cell for a hyperforest [12]:

**Definition 10 (Critical component)** *A regular component of a hyperforest will be called* critical *if none of its nodes is the source of either a loop or a non–regular hyperlink.*

We will proceed regular component per regular component. As $(N, L)$ does not have any hypercircuit, there is at least one regular component $R_0$ which is critical or incident to a loop. We will then process $R_0$ and, for $i > 0$, the regular components $R_i$ connected to $R_{i-1}$ by a non–regular hyperlink.

If $R_i$ is a critical component, its root $r$ will be an arbitrary node. Otherwise, its root $r$ will be the source of the loop or non–regular hyperlink incident to $R_i$. This root $r$ is uniquely defined by proposition 9. As $R_i$ is an ordinary tree, we will pair each of its links with its incident node closer to the root.

If the component is critical, $r$ will remain unpaired, and thus critical. Otherwise, we pair $r$ with its incident loop or non–regular hyperlink.

As the hyperforest does not contain any hypercircuit, the resulting pairing will correspond to an admissible discrete gradient vector field.

For example on Figure 11, the tree is processed from the leaves to the root, which is critical (in red).

**(d) Selecting cells of the hypergraph**

As the problem of selecting ($n$-1)–cells to form an optimal discrete gradient vector field is MAX–SNP hard, we had to use some heuristics: our selection algorithm uses a greedy method. This simple approach gives almost always optimal results [12]. As for the construction of a spanning tree, we maintain a union/find structure [20] that assigns to each cell its component number.

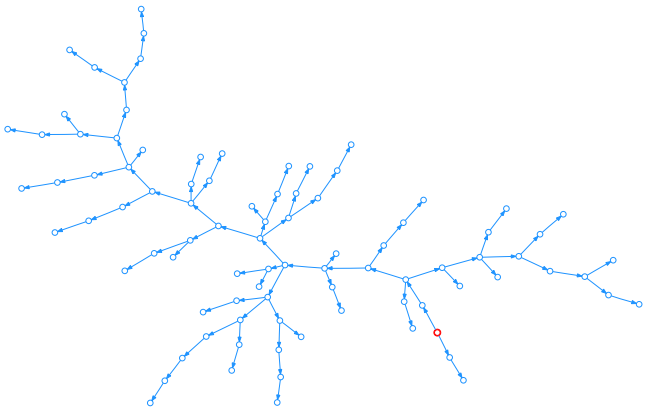


**Figure 11:** *First step: a spanning tree tetrahedron/triangle extracted from a homological sphere.*

We aim to construct a hyperforest, with all the cells of dimension $n$, and one hyperlink per selected cell of dimension ($n$-1).

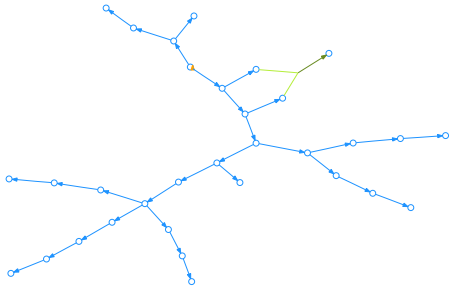


**Figure 12:** *Detail of a hyperlink insertion in the hyperforest appearing with a solid torus model.*

Adding the hyperlink of the left side of Figure 12 allows us to pair it with the node on the left. More generally, there will be less critical (unpaired) nodes if we can add more hyperlinks to the hyperforest. Therefore, in order to reach optimality, we will try to select the maximum number of hyperlinks. Our selection algorithm proceeds in 3 steps:

1. (*spanning tree of the regular components.*) First, for each regular hyperlink, we test whether it joins one or two components. In the latter case, we select it for the hyperforest and update the union/find structure. At the end of this step, the connected components of the tree are the regular components of the final hyperforest.

2. (*loop addition.*) Then we process every loop of the hypergraph. We add to the hyperforest at most one loop per regular component (according to proposition 9).

3. (*non–regular hyperlink heuristic.*) Finally, we process the non–regular hyperlinks. For each non–regular hyperlink $lk$ left, we test if it is incident to a critical component. We require the selected critical components to share only one node of $lk$. In that case, we add $lk$ to the

hyperforest, removing the critical status of the component.

As long as there are critical components, we have to test each non–regular hyperlink until we can add it or discard it because it is multiply incident to each incident critical component. This is the bottleneck point where the algorithm is, in the worst case, quadratic in the number of non–regular hyperlink encountered.

## 5 Topology Visualization

### (a) Visualizing the gradient field of a geometric model

Forman's discrete gradient vector field can be directly visualized on a geometrical object, by coloring each cell $\alpha$ according to its energy, which can be seen as the integral of the gradient field. This energy, which Forman defines as the discrete Morse function, corresponds to the ordering mentioned in Section 2(f), and to the time for the flow to reach the cell $\alpha$, starting at a critical point. It is computed as the depth of $\alpha$ in the hyperforest. This energy can be visualized with a color scale, as on Figure 1, where we visualize the projection of a 3–manifold embedded in $\mathbb{R}^4$, which is obtained by the Cartesian product of a Möbius strip with a circle.

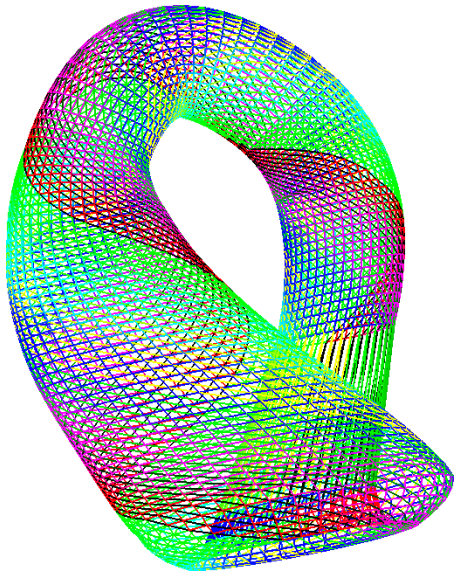
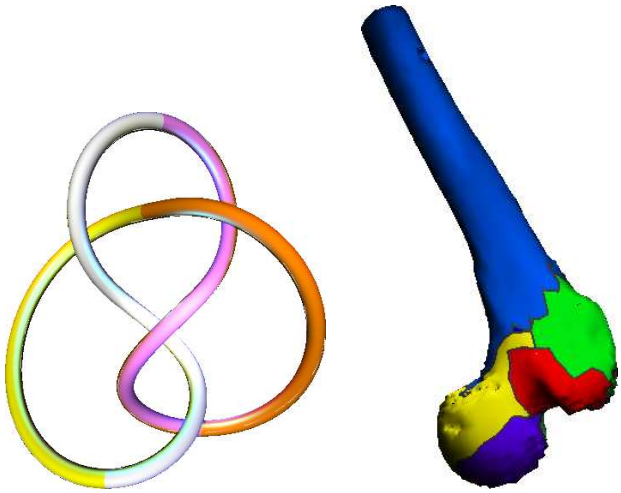


**Figure 13:** *The energy of a discrete gradient vector field on the edges/vertices of a Klein bottle model.*

For example on Figure 13, we drew the edges of a Klein bottle model with self–intersection. Edges drawn with the same color have the same energy. Looking at the green edges for example, we see clearly a Möbius strip spiraling along the bottle. This discrete gradient vector field gives an intuitive sense of the non–orientability of the Klein bottle.

### (b) Coherent quad patch decomposition

We can build a discrete gradient vector field imposing some geometrical constraints. The cell selection described in section 4(d) *Selecting cells of the hypergraph* is a greedy procedure. Thus, the cells can be selected with a geometrical cost in order to create minimal spanning trees. For example, considering the mean curvature as the cost function, the connected components of the hyperforest will tend to be globally as flat as possible.



(a) Quad decomposition of a figure eight knot, using a projection

(b) Quad decomposition of a bone model, using mean curvature

**Figure 14:** *Quad patch decomposition of surfaces.*

Moreover, we can pre–define the critical cells, and maintain them critical during the hyperforest creation: when adding a link or hyperlink, we can prevent union operations between components both containing a pre–defined critical cell. Combining those two techniques, we obtain a Smale–like decomposition. For surfaces, this decomposition is guaranteed to span exactly four critical cells. For example, using the curvature to pre–define the critical cells and to compute the geometrical cost, we can compute a globally flat decomposition of a surface with quads (see Figure 14). This requires an extra $O(n \cdot \log n)$ execution time.

### (c) Visualizing the structure of an abstract complex

Many topological objects appear without a geometric model, or with a model in higher dimensions. Those structures are quite difficult to understand without visualization. Forman's theory points out a topologically consistent way of choosing significant cells. Those cells can be outlined in a Hasse diagram

The Hasse diagram represents every cell by one node. On Figure 15, red nodes represent critical cells. Non–regular hyperlinks of the hyperforests are drawn in green. The cells of same dimension are displayed on the same row. The rows are ordered decreasingly with respect to the dimension.

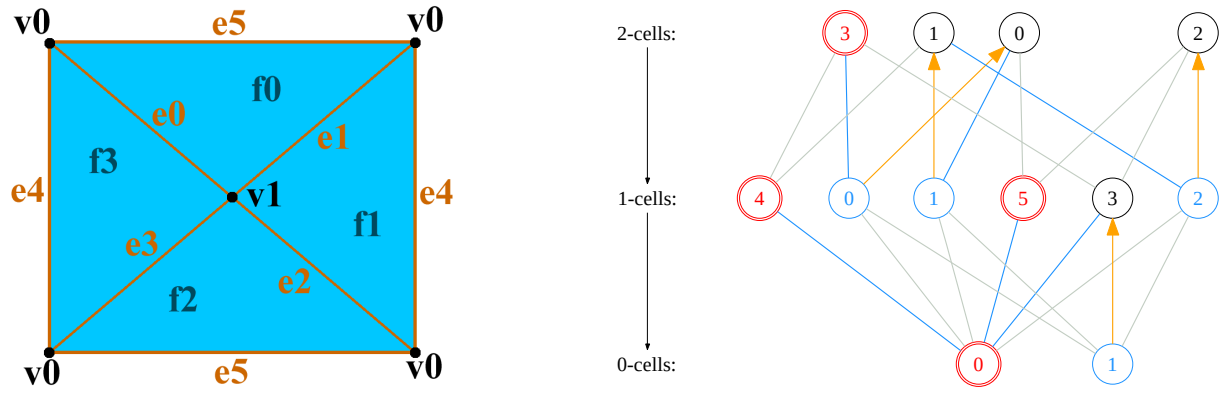



**Figure 15:** *A non–PL torus and its Hasse diagram.*

A link between two nodes symbolizes that the corresponding cells are incident one to the other. We linked by a red line paired cells. Blue lines represent the incidences we selected for the hyperforests and the final graph, as we did in the algorithm (see section 4(d) *Selecting cells of the hypergraph*).

## 6 Applications to mesh compression

Among the different strategies to compress the connectivity of meshes, many of the successful approaches are based on G. Taubin and J. Rossignac's *topological surgery* [21]. The basic idea is to compress the mesh using a tree, and then to encode this tree together with additional elements that will be necessary to reconstruct the mesh. Actually those methods inspired the algorithm we used to build discrete gradient vector fields. Those techniques have excellent performance and also have three main advantages. First, they allow the use of good geometry predictors, since the tree is built in a systematic manner, which allows efficient geometry compression. Second, they permit to efficiently reconstruct the topological adjacencies and incidences of the mesh. Third, they address a cell directly by the order of the tree construction. We will here use our construction of discrete gradient vector field to analyze, with Forman's theory, two of those compression algorithms: Edgebreaker [17] and Grow&Fold [19].

### (a) Edgebreaker

The Edgebreaker, introduced by Rossignac [17], is a compression scheme which encodes triangular surfaces with less than 2 bits per triangle.

***Edgebreaker compression scheme.*** Topologically speaking, it traverses spirally the dual graph of the surface, generating a dual spanning tree. At each step, a decision is made to move from a triangle $Y$ to an adjacent one $X$. To perform this decision, all visited triangles and their incident vertices are marked. Five situations are distinguished depending on the mark of the adjacent triangles and of the top vertex $v$. Those cases are denoted by the letters **C**,**L**,**E**,**R** and **S**. On Figure 16, the arrows starting from the triangle $X$ (in yellow) indicate the direction to the next triangle. Previously visited triangles are filled in gray.

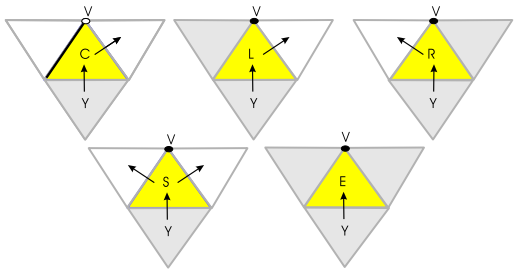



**Figure 16:** *The Edgebreaker encoding.*

***Compression of topological features.*** The simplified compression algorithm for a connected, oriented surface $M$ with genus $g$ and without boundary presented in [14] is a recursive procedure that generates a dual spanning tree. The recursion starts when reaching a triangle of type **S** and compresses the branch adjacent to the right edge of that triange. When the corresponding **E** triangle is reached, the branch traversal is complete and the routine returns from the recursion to pursue the left branch. At this point, two situations are distinguished. If the left triangle has not been visited during the right branch traversal (case of "normal" **S**), we move to the left neighbor and continue our encoding of the left branch. Otherwise (case of a "handle" **S**, denoted by $\mathbf{S}^*$), the left edge of the $\mathbf{S}^*$ triangle is encoded apart from the symbol string and the routine returns. Encountering an **E** that does not match an **S** terminates the compression process.

***Illustration.*** To illustrate the algorithm, consider the surface given by the model for a triangulated torus as shown in Figure 17. Identifying the edges on the opposite sides of the rectangle, one can build a simplicial complex in $\mathbb{R}^3$ whose polyhedron is homeomorphic to the torus.

Figure 17(left) illustrates the Edgebreaker's traversal of a torus. At the end of the algorithm, the code obtained for this surface is **CCCCRCSCRSSRLSEEE**. In this example,

The corresponding work was published in Transactions on Visualization and Computer Graphics, volume 10, number 5, pp. 499–508. IEEE Press, 2004.

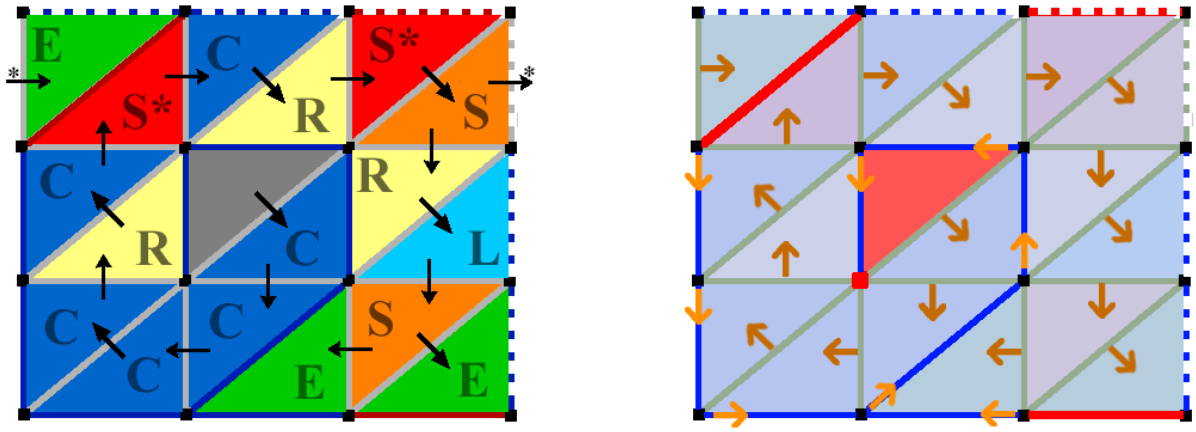



**Figure 17:** *Edgebreaker on a simple torus model: the traversal and the corresponding discrete gradient vector field.*

there are four triangles labeled with **S**. In the string sequence, the last two **S** are normal, since their right and left branches are traversed in the compression algorithm. On the other hand, the left branches of the two first **S** triangles (in red) have not been traversed, since their left triangle have been visited during their right branch traversal. They need to be transmitted apart for the decompression to follow the traversal. Moreover, it has been proved in [14] that the number of $\mathbf{S}^*$ triangles is $2g$.

***The corresponding gradient vector field.*** We can now define a discrete gradient vector field $V$ on a connected surface with genus $g$ and without boundary by using the dual tree and the vertex/edge tree generated by the Edgebreaker. First, we select the initial triangle to be the root of the dual spanning tree, and one of its vertices to be the root of the vertex/edge tree. Then, we build the gradient vector field on those trees following the steps presented in the Section 4(b). Finally, the selected triangle and vertex are the critical cells of dimensions 2 and 0 of $V$. The edges that are not on those two trees remain unpaired. They correspond to the left edges of the $2g$ $\mathbf{S}^*$ triangles. As a consequence, they will be the critical 1–cells of $V$. At the end, we have constructed a gradient vector field using the Edgebreaker that has one critical 0–cell, $2g$ critical 1–cells, and one critical 2–cell. The authors proved in [11] that a discrete gradient vector field defined on $M$ with this number of critical cells is optimal. This construction can be done separately on the different connected components of the surface, and extends naturally to surfaces with boundaryby choosing the first triangle as an infinite cell connecting *all* the boundary loops. The discrete gradient vector field of Figure 17(left) is illustrated on Figure 17(right).

***Conclusion.*** In this section, we saw that the Edgebreaker compression scheme builds an optimal discrete gradient vector field. The critical cells are detected by the Edgebreaker as topological singularities (such as genus) and encoded separately. Actually, this results extend to any compression algorithm based on topological surgery. This gives a simple way to extend those algorithms to handle surfaces with genus, boundary, non–manifold edges... In the next section, we will extend this result to topologically analyze the Grow&Fold tetrahedral mesh compression algorithm.

### (b) Grow and Fold

The Grow&Fold algorithm of A. Szymczak and J. Rossignac [19] is an extension of Edgebreaker to tetrahedral meshes. It allows a representation of a tetrahedral mesh with 7 bits per tetrahedron, and some additional bits for "topological locks" in the tree: the *glue triangles*.

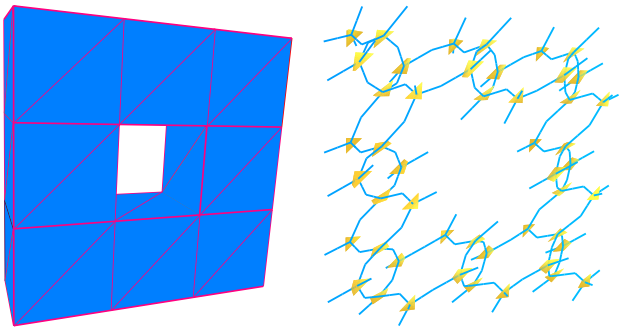


**Figure 18:** *A ring made of tetrahedrons and its dual graph.*

From a topological point of view, the Grow&Fold algorithm creates and encodes a tetrahedron/triangle spanning tree out of the dual graph of the 3–manifold (see Figure 18 and Figure 19). During this traversal, the triangles are classified in 3 categories:

- *door triangles*: links of the dual spanning tree.
- *external triangles*: triangles on the boundary of the mesh.
- *cut triangles*: the remaining ones.

During the decompression, after having rebuilt the spanning tree, each cut triangle is duplicated: there is a copy of it on each of its two incident tetrahedrons. The compression

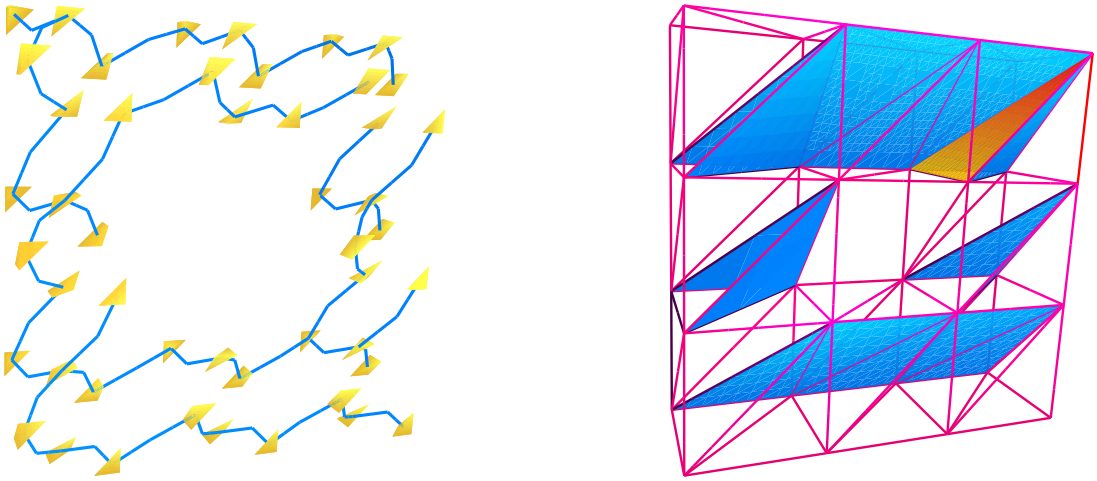


**Figure 19:** *The tetrahedron tree and the cut triangles.*

algorithm could encode this identification by sending pairs of corners opposite to this cut triangle, but this would be too expensive. Most of those identifications can be encoded by *folding* operations, when the two tetrahedrons share a *free edge*: an edge which is shared by the two copies of a cut triangle during the decompression. A folding code is sent when the free edge it touched by the spanning tree. When the cut triangles form a loop, some identifications have to be encoded directly, by sending pairs of opposite tetrahedrons. The corresponding triangle will be called a *glue triangle*.

### (c) The corresponding discrete gradient vector field

We have shown the natural connection between discrete gradient vector fields and the Edgebreaker compression scheme. We will prove here that the folding scheme is equivalent to a discrete gradient vector field, in which each glue triangle corresponds to a critical cell. This will prove that minimizing the number of glue triangles (which are the most expensive to encode) is MAX–SNP hard in the general case. We will build a discrete gradient vector field based on Grow&Fold's strategy.

***Tetrahedron/triangle hyperforest.*** The tetrahedron/triangle spanning tree of Grow&Fold is used *as is* for the hyperforest of the first step of our construction algorithm. It contains all the 3–cells (all the tetrahedrons), and some 2–cells (the door triangles) without creating any hypercircuit. This hyperforest is completed by one external triangle per connected component, in the same way we added loops at step 2 of the selection algorithm.

***Triangle/edge hyperforest.*** The folding scheme corresponds to the second step of our algorithm: it defines a part of the triangle/edge hyperforest. Each folding operation matches a (free) edge with the incident triangle being folded. For each of these matchings, we add a hyperlink representing the free edge to the hyperforest. As each of those folding operations identifies two copies of the cut triangle during decompression, the matching defined above cannot create any hypercircuit. The triangles that remain unmatched are the external and the glue triangles. We match those external triangles with some of the boundary edges through a triangle/edge spanning tree of the boundary. Since each folding sequence contains at most one free edge on the boundary, the resulting triangle/edge hypergraph is a hyperforest (see Figure 12).

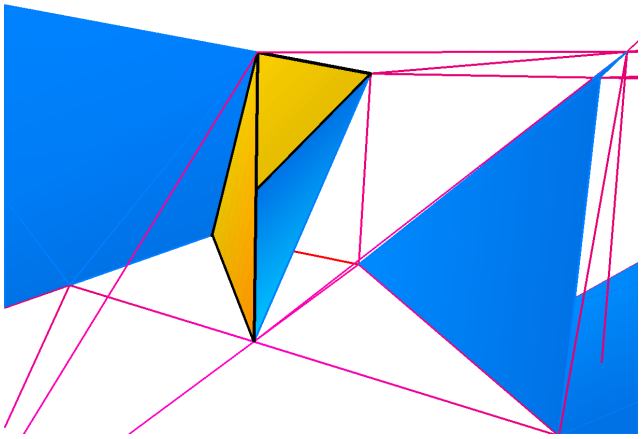


**Figure 20:** *One of the two triangles in orange must be a glue triangle (closer view of the model of Figure 19).*

***Vertex/edge forest.*** The last step of our algorithm constructs a spanning forest of the vertices. Depending on the topology of the initial object, it can happen that some edges would create cycles in this forest, and will thus remain critical.

### (d) On Grow and Fold's optimization

***Glue triangles.*** In the above construction, the glue triangles remained critical. For example, on Figure 20, the two triangles in orange have one common edge and the other

edges are on the boundary. In this folding scheme, such cut triangle has to be a glue triangle, since none of its edges is free. This corresponds to the topological singularity of the model: an annulus (see Figure 18). Minimizing the number of glue triangles will therefore require to minimize the number of critical cells. We already know that this optimization is a MAX–SNP hard problem in the general case.

***Independence from the spanning tree.*** The tetrahedron/triangle spanning tree $T$ of Grow&Fold could be modified, in order to minimize the number of glue triangles, as proposed in [19]: "an efficient algorithm which, by changing the tetrahedron tree, decreases the number of glue triangles". However, the number of glue triangles does not depend on $T$: for any tetrahedron/triangle spanning tree $T$, $T$ defines a sequence of collapses. Therefore, $K \setminus \{root(T)\}$ is homotopy equivalent to $K \setminus T$. Since the minimum number of critical cells is an invariant for 3–manifolds [10], the minimal numbers of critical cells of $K \setminus T$ is independent from $T$, and is equal to the minimal numbers of critical cells of $K \setminus \{root(T)\}$. Therefore, the minimum number of glue triangles, which corresponds to critical cells of a particular discrete gradient vector field, does not depend on $T$.

***Conclusion.*** We have proved that the Grow&Fold's compression actually constructs a discrete gradient vector field. The costly elements, i.e. the glue triangles, correspond to critical cells of the discrete gradient vector field. We deduced that minimizing the number of glue triangles is MAX–SNP hard. Moreover, reaching this minimum cannot be achieved by only modifying the tetrahedron/triangle spanning tree in the general case. However, in practical cases, the greedy algorithm used in [19] is comparable to our methods (with different heuristics) and gives pretty good results.

## 7 Future Works

We intended by this work to illustrate Forman's theory, and to use some of its concepts to visually analyze the topology of an object. We presented an explicit construction of discrete gradient vector fields, whose result is almost always optimal. With this fundamental tool, we provided various ways of using it to visually extract topological information of a given combinatorial structure. We finally used this theory to complete the analysis of Grow&Fold tetrahedral mesh compression algorithm.

We plan to continue this work in three directions. First, we will try to determine the conditions under which the greedy strategy of our algorithm and of Grow&Fold reaches the optimum. Second, we will try to develop graphical tools to capture as much as possible the topology of 3–manifolds, where very hard mathematical problems remain unsolved. Finally, we look forward to produce a topologically consistent morphing based on mapping directly the discrete gradient field between two objects of the same homotopy type.

## References

[1] C. Berge. *Graphes et hypergraphes*. Dunod, Paris, 1970.

[2] M. W. Bern, D. Eppstein et al. Emerging challenges in computational topology, 1999.

[3] M. M. Cohen. *A course in simple homotopy theory*. *Graduate text in Mathematics*. Springer, New York, 1973.

[4] T. K. Dey, H. Edelsbrunner and S. Guha. Computational topology. In B. Chazelle, J. E. Goodman and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, volume 223 of *Contemporary mathematics*, pages 109–143. American Mathematical Society, Providence, 1999.

[5] R. Forman. A discrete Morse theory for cell complexes. In S. T. Yau, editor, *Geometry, Topology and Physics for Raoul Bott*, pages 112–115. International Press, Cambridge, 1995.

[6] R. Forman. Morse theory for cell complexes. *Advances in Mathematics*, 134:90–145, 1998.

[7] M. Hachimori. Simplicial complex library. infoshako.sk.tsukuba.ac.jp/˜hachi.

[8] J. C. Hart. Morse theory for implicit surface modeling. In H.-C. Hege and K. Polthier, editors, *Visualization and Mathematics*, pages 257–268, Heidelberg, 1998. Springer.

[9] J. Hopcroft and R. E. Tarjan. Efficient algorithms for graph manipulation. *Communications of the ACM*, 16:372–378, 1973.

[10] T. Lewiner. Constructing discrete Morse functions. Master's thesis, *Department of Mathematics, PUC–Rio*, Aug. 2002. Advised by Hélio Lopes and Geovan Tavares.

[11] T. Lewiner, H. Lopes and G. Tavares. Optimal discrete Morse functions for 2–manifolds. *Computational Geometry: Theory and Applications*, 26(3):221–233, 2003.

[12] T. Lewiner, H. Lopes and G. Tavares. Towards optimality in discrete Morse theory. *Experimental Mathematics*, 12(3):271–285, 2003.

[13] H. Lopes and G. Tavares. Structural operators for modeling 3–manifolds. In C. Hoffman and W. Bronsvort, editors, *Solid Modeling and Applications*, pages 10–18. ACM, 1997.

[14] H. Lopes, S. Pesco, G. Tavares, M. G. M. Maia and Á. Xavier. Handlebody representation for surfaces and its applications to terrain modeling. *Journal of Shape Modeling*, 9(1), 2003.

[15] A. T. Lundell and S. Weingram. *The topology of CW–complexes*. Van Nostrand Reinhold, New York, 1969.

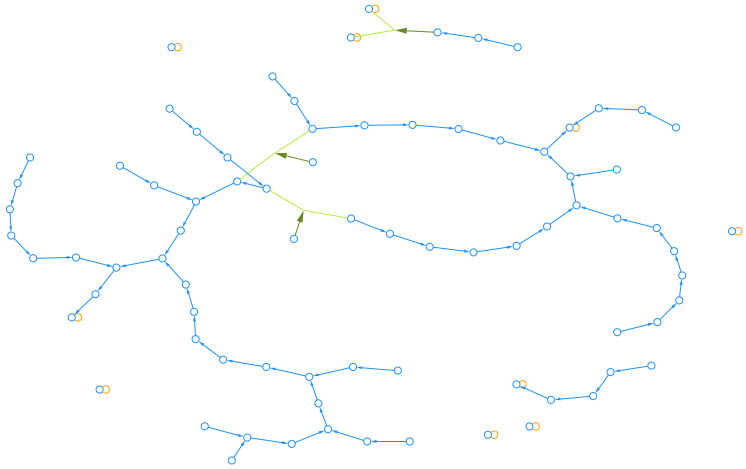


**Figure 21:** *The triangle/edge hyperforest of the example of Figure 19.*

[16] J. W. Milnor. *Morse theory*. Number 51 in *Annals of Mathematics Study*. Princeton University Press, 1963.

[17] J. Rossignac. Edgebreaker: connectivity compression for triangle meshes. *Transactions on Visualization and Computer Graphics*, 5(1):47–61, 1999.

[18] Y. Shinagawa, T. L. Kunii and Y. L. Kergosien. Surface coding based on Morse theory. *Computer Graphics and Applications*, 11:66–78, 1991.

[19] A. Szymczak and J. Rossignac. Grow & Fold: compressing the connectivity of tetrahedral meshes. *Computer–Aided Design*, 32(8/9):527–538, 2000.

[20] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of the ACM*, 22(2):215–225, 1975.

[21] G. Taubin and J. Rossignac. Geometric compression through topological surgery. *Transactions on Graphics*, 17(2):84–115, 1998.

The corresponding work was published in Transactions on Visualization and Computer Graphics, volume 10, number 5, pp. 499–508. IEEE Press, 2004.