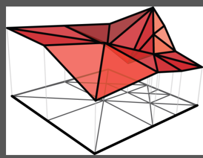
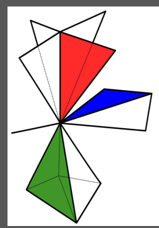




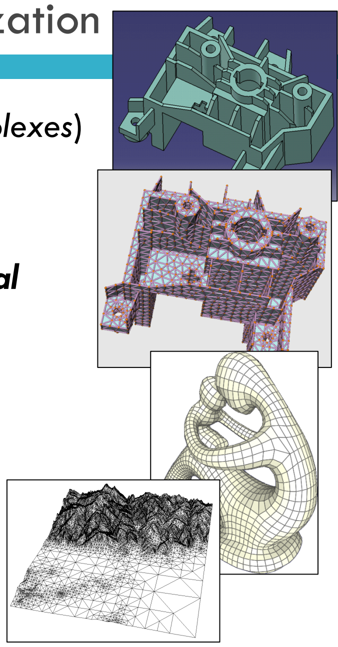
Leila De Floriani,
University of Genova (Italy)

EFFICIENT AND EFFECTIVE REPRESENTATIONS FOR SHAPE MODELING AND ANALYSIS

Introduction: shape discretization

2

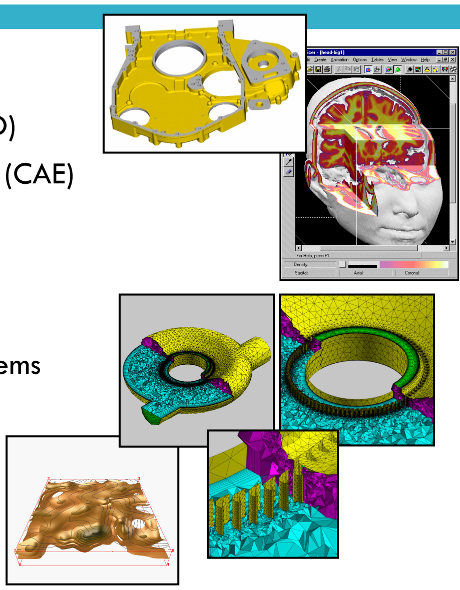
- ***Cellular decompositions*** (*cell complexes*)
- ***Triangle and tetrahedral*** meshes (*simplicial complexes*)
- ***Quad*** and unstructured ***hexahedral*** meshes
- ***Nested*** meshes (e.g, quadtrees, hierarchical triangle meshes)



Introduction: applications

3

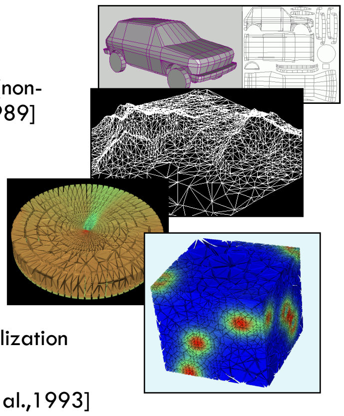
- Computer Graphics
- Computer-Aided Design (CAD)
- Computer-Aided Engineering (CAE)
- Finite Element Analysis
- Animation
- Scientific visualization
- Geographic Information Systems
- Machine learning
- …..



Introduction: history

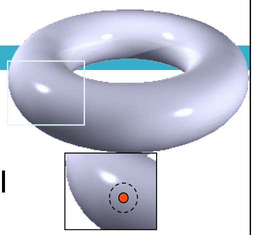
4

- ***Cell complexes:*** basis for boundary models of objects in solid modeling systems
  - first representation: *Winged-Edge data structure* [Baumgardt, 1972]
  - first representation for shapes with singularities (non-manifold): *Radial-Edge data structure* [Weiler, 1989]
- ***Triangle meshes:***
  - basis for terrain modeling [Gold, 1977]
  - finite elements analysis…
- ***Tetrahedral meshes:***
  - representation for volume data in scientific visualization [Cignoni et al., 1994]
  - volumetric representation of objects [Paoluzzi et al.,1993]
  - finite elements analysis…

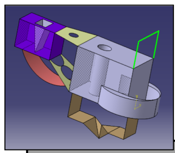


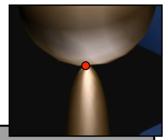
Shape topology

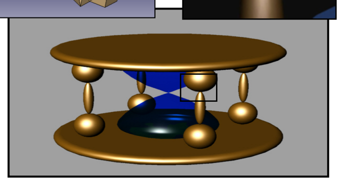
5



- ***Manifold:***
  - neighborhood of each point is a topological open ball (or half-ball)
- ***Non-manifold:***
  - non-manifold joints
  - parts of different dimensionalities

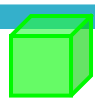






Background: cell complexes

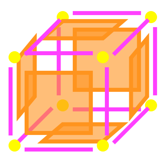
6



A 3-cell

A closed 3-ball

- ***k-dimensional cell (k-cell)*** σ in the n-dimensional Euclidean space E^n: a subset of E^n homeomorphic to a closed k-dimensional ball
- ***d-dimensional cell complex (cell d-complex):*** finite collection of k-cells ($k \leq d$) such that
  - the intersection of any two non-disjoint cells σ and σ' is the union of cells of the complex



Cell 2-complex:
- **0-cells (vertices)**
- **1-cells (edges)**
- **2-cells (faces)**

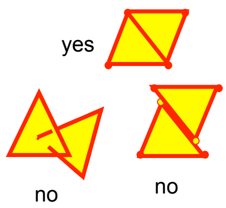
Background: simplicial complexes

7

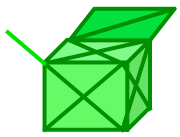
- *k-dimensional Euclidean simplex (k-simplex):*
  - convex hull of k+1 linearly independent points in the n-dimensional Euclidean space E^n (k≤n)

1-simplex (edge)
2-simplex (triangle)
3-simplex (tetrahedron)

- *d-dimensional simplicial complex* finite collection of k-simplexes (k≤d) such that
  - the intersection of any two simplexes, if not empty, is a simplex belonging to both of them



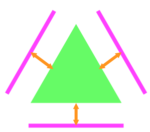
A simplicial 2- complex



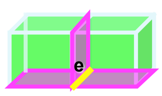
Background: topological relations

8

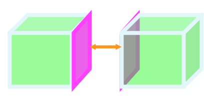
- Ingredients for defining data structures for cell and simplicial complexes:
- *Incidence (boundary and co-boundary)* relations: relations between cells of different dimensions
- *Adjacency* relations: relations between k-cells of the same dimension though (k-1)-dimensional cells



Incidence boundary relation between 2-simplex and its bounding 1-simplexes



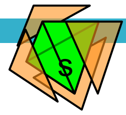
Incidence co-boundary relation between a 1-cell and three 2-cells



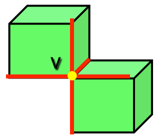
Adjacency relation between two 3-cells sharing a face (2-cell)

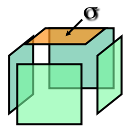
Operations to be supported

9

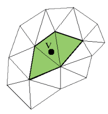


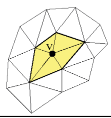
- *Topological connectivity queries* for retrieving
  - cells on the ***boundary*** of a given cell
  - ***co-boundary*** of a cell: cells bounded by a given one
  - cells ***adjacent*** to another one along a lower dimensional cell

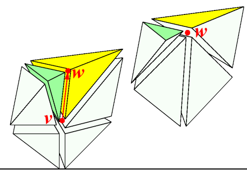




- *Updates*
  - e.g., vertex insertion/deletion; edge contraction; edge swaps; Euler operators; simplex collapse

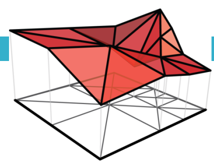




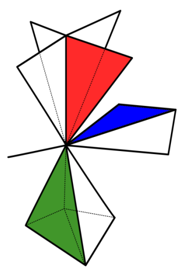


Requirements for data structures

10

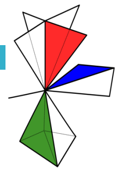


- Compactness
- Efficient support to
  - ***topological connectivity queries***
  - ***updates***
- Flexibility
- Ease of use
- Scalability to manifolds (for non-manifold data structures)

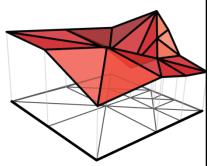


Outline

11

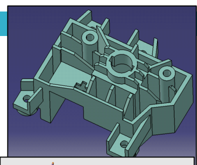


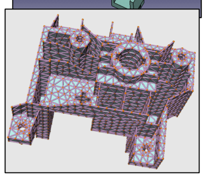
- Taxonomy of data structures
- Review of data structures for manifold complexes
- Approaches to non-manifold shape modeling
- Data structures for simplicial complexes in arbitrary dimensions
- The Mangrove library
- Towards a localized approach: the PR-star octree

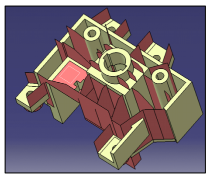


Data structures - Taxonomy

12



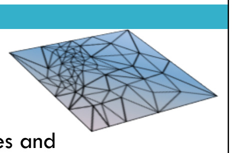




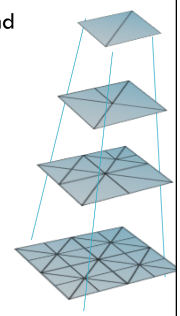
- Discretization of the shape
  - **cell** versus **simplicial** complexes
- Topology of the shape
  - **manifold** versus **non-manifold**
- Dimension of the shape and of its discretization
  - **dimension-specific** data structures
  - **dimension-independent** data structures

Data structures - Taxonomy (2)

13



- *Entities encoded*
  - ***all the cells*** (e.g., in a triangle mesh: all the triangles, edges and vertices)
  - ***a subset of the cells*** (e.g, in a triangle mesh: only triangles and vertices)



- *Granularity*
  - ***flat*** data structures: a single mesh discretizing the shape
  - ***multi-resolution (Level-Of-Detail (LOD))*** data structures:
    - a collection of meshes discretizing the same shape

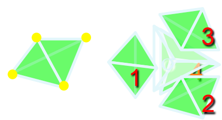
Data structures - Taxonomy (3)

14

- Explicit versus implicit representations

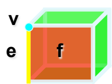
Entities: tetrahedra and vertices

- ***Explicit data structures:***
  - a subset of the cells
  - topological relations among cells (explicit)



Encoded relations:
Tetrahedron-Vertex and
Tetrahedron-Tetrahedron

- ***Implicit data structures:***
  - topological relations are encoded indirectly as tuples of cells in the same relation

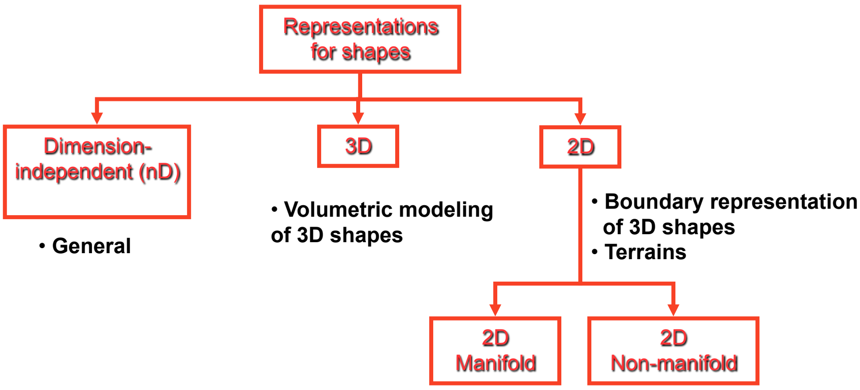


(v, e, f): triple of vertex v, edge e and face f on the surface of a hollow cube

Data Structures: Presentation

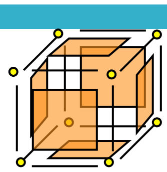
15

- We follow a classification of data structures based on dimension

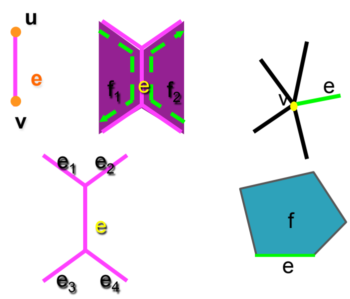


2D Cell complexes - manifold

16

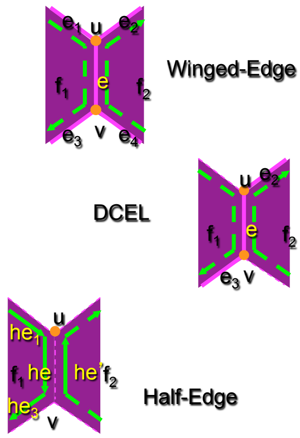


- **Cells:** vertices, edges and faces
  - an **edge** is shared by **at most two** faces
- Most common: ***edge-based representations***
- **Edge** plus its local connectivity:
  - *Edge-Vertex* relation
  - *Edge-Face* relation
  - Partial *Edge-Edge* relation



Partial Edge-Edge Relations

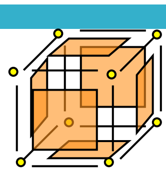
- **Winged-edge:** four edges in $R_{1,1}(e)$
- **DCEL:** two edges in $R_{1,1}(e)$
- **Half-Edge:**
  - **edge e** consists of two half-edges **he** and **he'**
  - **two edges** in $R_{1,1}(e)$ associated with **he**
  - other two associated with **he'**

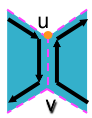


2D cell complexes – manifold (2)

18

- Edge-based representations
  - **Winged-Edge** [Baumgart, 1972]; **Half-Edge** [Mantyla, 1983]; **Quad-edge** [Guibas et al, 1985], etc.
- **Half-edge implementation** (public domain)
  - **Mantyla's book** (1988)
  - **OpenMesh Library** (Computer Graphics Group, RWTH Aachen).
  - **Computational Geometry Algorithms Library (CGAL)**



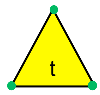


2D Simplicial complexes – manifold

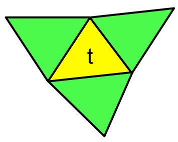
Triangle meshes

19

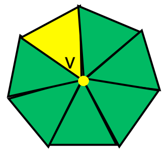
- Edge-based data structures
  - For cell complexes: vertices, edges and triangles explicitly encoded



- Triangle-based data structures
  - Only vertices and triangles explicitly encoded
    - **Indexed with Adjacencies (IA)** [Gold, 1977]; Corner table [Rossignac et al., 2001]
    - **6.5** integer references per **triangle**
  - Compact representations (for fixed-connectivity meshes)
    - SOT, LR, SQuad [Gurung et al., 2010 and 2011]
    - from **3** to **1.08** (on average) references per **triangle**



- Edge-based data structures about 1.8 times IA

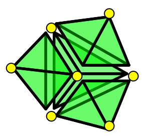


3D Simplicial complexes – manifold

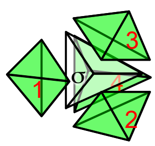
Tetrahedral meshes

20

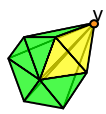
- Data structures representing all entities
  - Vertices, edges, faces and tetrahedra encoded
    - e.g., Facet-Edge [Dobkin and Lazlo, 1987], Triangle-Edge [Natarajan et al., 2004]



- Tetrahedron-based data structures
  - Only **vertices** and **tetrahedra** encoded
    - Indexed with Adjacencies (IA) [Nielson, 1997, Paoluzzi et al, 1993], CHF [Lage et al., 2005]
    - **8.5/8** references per **tetrahedron**



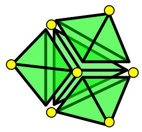
  - Compact representation
    - for fixed-connectivity meshes
    - **4** references per **tetrahedron -** SOT [Gurung et al, 2010]



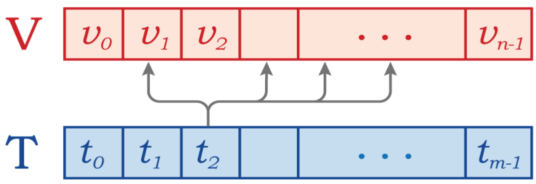
Indexed triangle/tetrahedral data structures

21

- Array of vertices **V**
  - Each vertex v_i encodes a position in Euclidean space and possibly other attributes
- Array of triangles/tetrahedra **T**
  - Each triangle/ tetrahedron t_i encodes the index in **V** of its vertices and possibly other attributes



For tetrahedral meshes



$R_{3,0}$

$v_i = \{x, y, z\}$

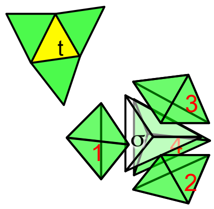
$t_i = \{i_{v0}, i_{v1}, i_{v2}, i_{v3}\}$

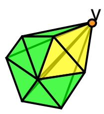
IA data structure:

Indexed data structure with Adjacencies

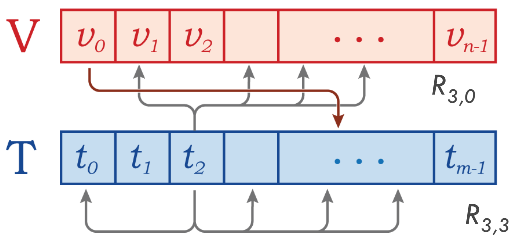
22

- Array of vertices **V**
  - Encodes position of each vertex
  - Encodes a single incident triangle/ tetrahedron in **T**
- Array of triangles/tetrahedra **T**
  - Encodes indices of three/four vertices in **V**
  - Encodes indices of three/ four adjacent triangles/ tetrahedra in **T**





For tetrahedral meshes



$R_{3,0}$

$R_{3,3}$

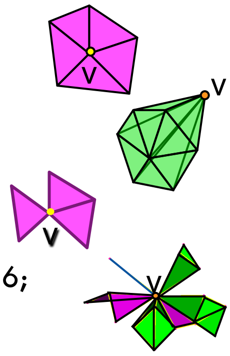
$v_i = \{x, y, z, i_t\}$

$t_i = \left\{ \begin{matrix} i_{v1}, i_{v2}, i_{v3}, i_{v4} \\ i_{t1}, i_{t2}, i_{t3}, i_{t4} \end{matrix} \right\}$

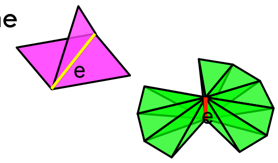
Representing non-manifolds

23

- **Combinatorial d-manifold** [without boundary]: pure complex in which the **link** of each k-cell is homemorphic to a triangulation of *(d-k-1)*-sphere
  - **star** of a cell: collection of all cells incident in it
  - **link:** boundary of the star



- The class of d-manifolds is **not decidable** for d>=6; it is an open problem for d=5
- A weaker definition of singularity related to the connected components of the link of a cell

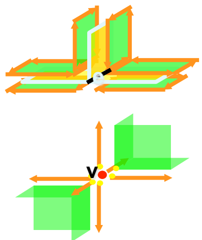


Representing non-manifolds

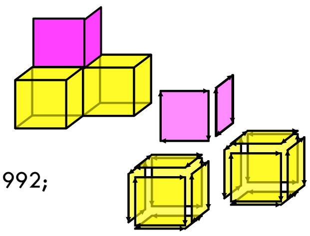
Cell complexes

24

- Extensions of edge-based data structures for manifolds
  - Radial Edge [Weiler, 1989]
  - Partial Entity [Lee and Lee, 2001]
  - Tri-cyclic Cusp data structure [Gursoz et. al, 1990]
  - Coupling Entities data structure [Yamaguchi and Kimura, 1995]
  - Extended maps [Cazier and Kraemer, 2010]



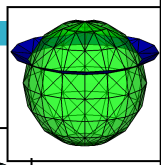
- Decomposition into manifold components
  - splitting at non-manifold vertices and edges
  - [Desaulniers and Stewart, 1992; Falcideno and Ratto, 1992; Rossignac and Cardoze, 1999; Pesco et al., 2004]

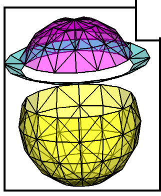


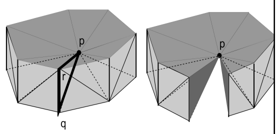
Decomposition approach to non-manifolds

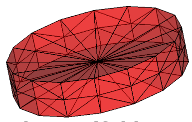
25

- **Theoretical issues**
  - Decomposition into manifolds is not feasible in higher dimensions
  - **Even in 3D:** we cannot decompose into manifolds parts without cutting at a manifold face
- **Decomposition into nearly manifold components** [DeFloriani, Morando, Puppo, 2003; Hui et al., 2006]
  - generated by splitting a simplicial complex only at simplexes corresponding to singularities
  - unique
  - valid in arbitrary dimensions







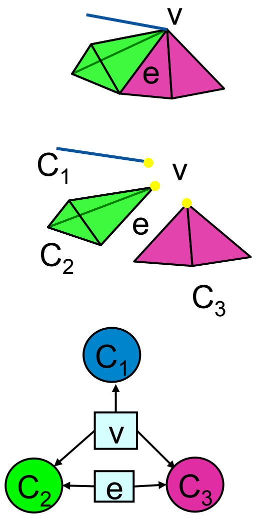


Valid nearly manifold component

Decomposition approach (2)

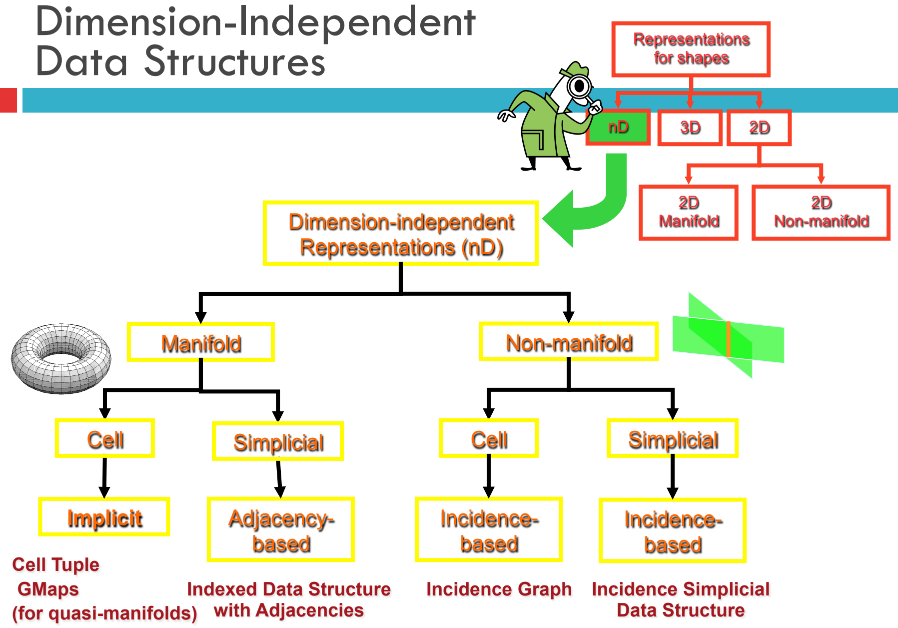
26

- Connectivity among the components
  - through a hypergraph
- Triangle- or tetrahedron-based data structures for the components
  - Double-Level Decomposition (DLD) [Hui et al., 2006] when using the IA data structure
  - Compact
  - Efficient topological queries
  - Difficult to update
  - Suitable for extending compact representations to "non-manifolds"



Dimension-Independent Data Structures

27

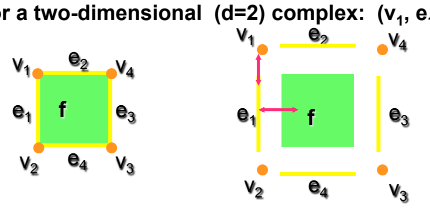


Cell tuple data structure [Brisson 1998]

28

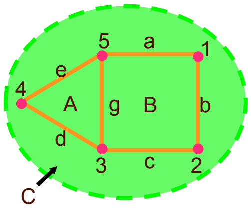
- Implicit representation for **manifold cell complexes**
- Basic elements: tuples of **cells** and **switch operators** between tuples
- **Cell-tuple:** (d+1)-tuple of cells $(\sigma_0, \ldots, \sigma_i, \ldots, \sigma_d)$ such that
  - σ_i is an i-cell,
  - σ_i belongs to the boundary of σ_{i+1} (I = 0, 1,…, d-1)

A cell-tuple for a two-dimensional (d=2) complex: (v_1, e_1, f)

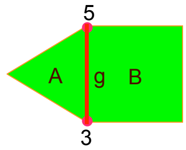


The Cell-tuple Data Structure: an Example for a 2-Complex

- The cell-tuple data structure for the 2-complex shown in the picture consists of 24 cell-tuples:
  - Each cell tuple consists of a vertex, edge and face
  - An edge appears in four tuples (since it is common to two faces and has two extreme vertices)



Cell tuples of edge g:
(3,g,A)
(3,g,B)
(5,g,A)
(5,g,B)

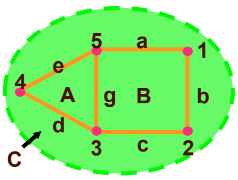


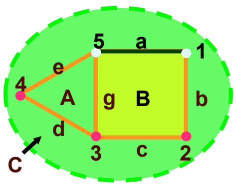
Cell-tuple Data Structure: Switch Operator

30

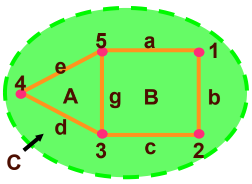
Given a cell-tuple $(\sigma_0, \sigma_1, \ldots, \sigma_d)$:

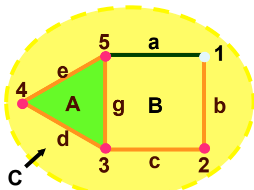
- ***switch***$_i$ $(\sigma_0, \ldots, \sigma_i, \ldots, \sigma_d) = (\sigma_0, \ldots, \gamma_i, \ldots, \sigma_d)$
  - γ_i is an i-cell different from σ_i
  - $(\sigma_0, \ldots, \gamma_i, \ldots, \sigma_d)$ is a cell-tuple





switch$_0$(1,a,B)= (5,a,B)

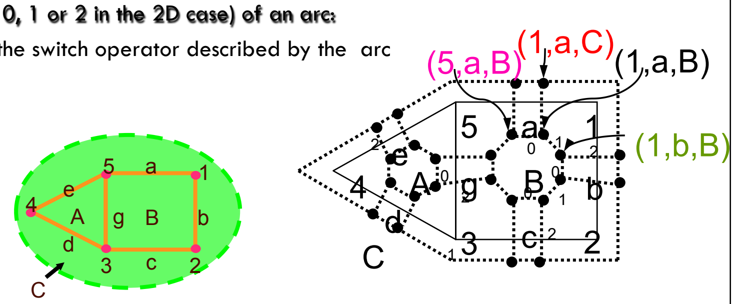




switch$_2$(1,a,B)= (1,a,C)

Cell-tuple data structure as a graph

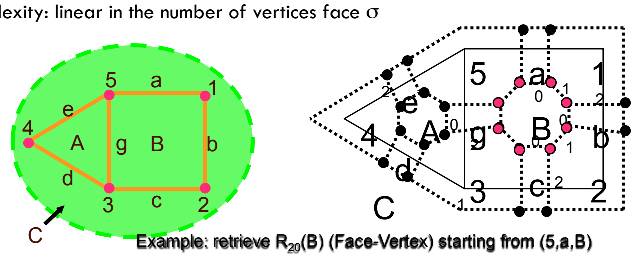
- **Cell tuples** are represented as **nodes of a labeled graph**
- Each **arc** of the graph represents a **switch operator**
- The **label (= 0, 1 or 2 in the 2D case) of an arc:**
  - index of the switch operator described by the arc



- For a 2-complex: **# tuples (nodes) = 4*# edges** and **# arcs = 6* # edges**
- Verbose representation
- All topological relations can be retrieved in optimal time

Retrieving Topological Relations: an Example

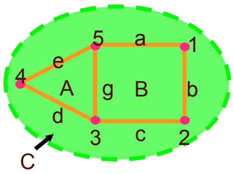
- **Retrieve boundary relation $R_{2,0}$ (Face-Vertex relation) for a face σ**
- Find a tuple containing σ
  - Let this be (v,**e**,σ), apply alternatively:
  - *switch*$_0$ (v,**e**,σ) = (v',**e**,σ), which gives the other vertex of e
  - *switch*$_1$ (**v**,e,σ) = (**v**,e',σ), which gives the other edge sharing vertex v and face σ
- Time complexity: linear in the number of vertices face σ



Example: retrieve R_{20}(B) (Face-Vertex) starting from (5,a,B)

Cell-tuple Data Structure as a Graph

- **Cell tuples** are represented as **nodes of a labeled graph**
- Each **arc** of the graph represents a **switch operator**
- In a 2-complex:
  - for each tuple we have three switch operators
  - **# tuples (nodes) = 4*# edges**
  - **# switch operators (arcs) = 6* # edges**
- It tends to be a verbose representation
- All topological relations can be retrieved in optimal time by applying sequences of switch operators, given a starting tuple



Generalized Maps (G-Maps)

[Lienhardt, 1988; 1994]

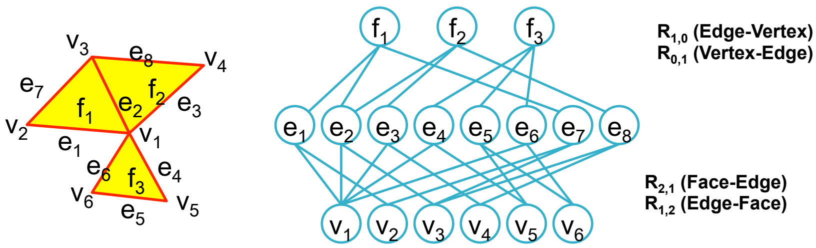
34

- Purely combinatorial objects
- Based on the concept of dart (corresponding to cell-tuple)
- Each dart is formed by N+1 involutions (very similar to switch operators)
- Domain: **quasi-manifold** complexes, a subclass of pseudo-manifolds
- Very elegant implementations [Levy and Mallet, 1999]

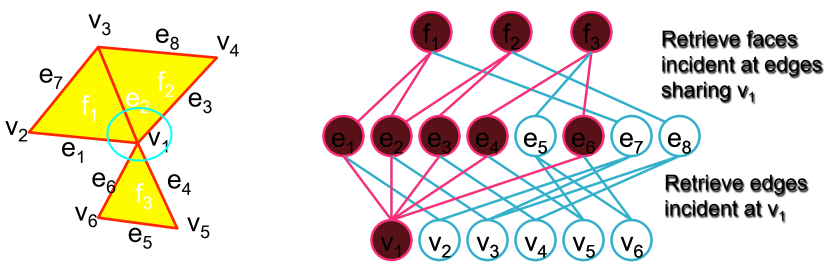
Incidence Graph [Edelsbrunner, 1987]

35

- General: for ***arbitrary cell complexes***
- Implementation of Hasse diagram
- All cells explicit encoded
- Immediate *boundary* (R_{pp-1}) and co-*boundary* (R_{pp+1}) relations encoded for each p-cell



Incidence Graph: Retrieving a Co-boundary Relation



Retrieving $R_{0,2}(v_1)$ (Vertex-Face relation): all faces incident at v_1

Incidence graph and implicit data structures

37

- Domain
  - IG: arbitrary cell complexes
  - Cell-tuple: manifold cell complexes
  - G-maps: quasi-manifold complexes
- Cell-tuples/ G-maps
  - paths in the incidence graph
  - ordered models
- Encoding
  - Cell tuples/G-maps as nodes of a labeled graph
    - Arcs of the graph represent switch operators (dart for G-maps)
  - G-maps more verbose than IG: IG/G-Maps ~50% in 2D; IG/G-Maps ~18% in 3D

Data structures for simplicial complexes

38

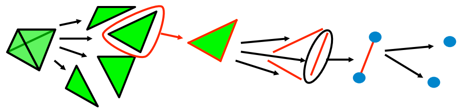
- ***Common requirements***
  - Domain: abstract simplicial complexes of arbitrary dimension
  - Dimension-independent design and implementation
  - Scalable to manifolds
  - Efficient support to both connectivity queries and updates
- ***Conflicting requirements***
  - Explicit encoding of all simplices
  - Compactness

Incidence Simplicial (IS) data structure

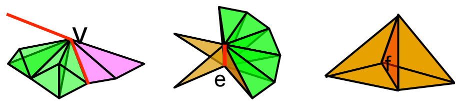
[DeFloriani et al., 2011]

39

- All simplices are explicitly encoded
- Simplified version of the Incidence Graph (IG)
  Topological connectivity
  - relation between an i-simplex and the (i-1)-simplices on its **boundary** (as IG)



  - **minimal** encoding of the local neighborhood (**co-boundary**) of a simplex

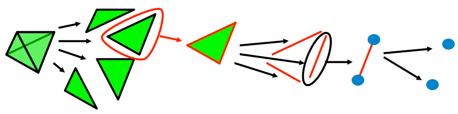


IS data structure

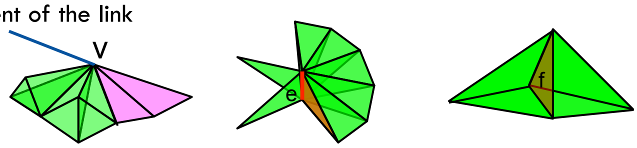
an example for simplicial 3-complexes

40

- Boundary relations:
  - **Tetrahedron-Triangle**
  - **Triangle-Edge**
  - **Edge-Vertex**



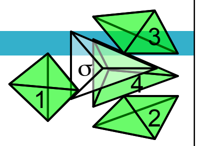
- Co-boundary relations as partial relations:
  - **Vertex-Edge:** one edge for each connected component of the link
  - **Edge-Triangle:** one triangle for each connected component of the link
  - **Triangle-Tetrahedron:** one tetrahedron for each connected component of the link



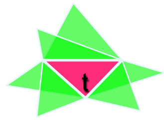
Generalized IA (IA*) data structure

[Canino et al., 2011]

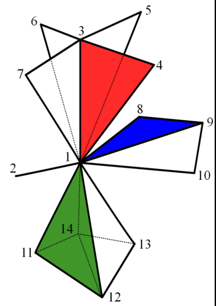
41



- Only vertices and **top simplexes** encoded
  - Top simplexes: not on the boundary of any other simplex (e.g., in 3D: tetrahedra, dangling triangles and edges)

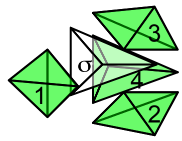


- Adjacency-based representation
  - Extension of IA data structure
- Topological connectivity
  - **boundary relations** from a top simplex and its bounding vertices
  - **adjacency relations** among top simplexes
  - minimal encoding of the **star** of each **vertex**

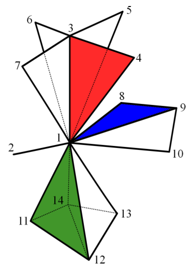


IA* data structure: an example for 3-complexes

42



- Only **vertices** and **top simplexes** (tetrahedra, dangling triangles and edges)
- **Vertex connectivity** for top simplexes (R_{k0} relations)
- **Tetrahedron-Tetrahedron** relation
- **Triangle-Triangle** and **Edge-Edge** relations for dangling triangles and edges
- **One simplex** for each cluster of k-dimensional simplexes incident at a **vertex**



IS vs IA* data structure

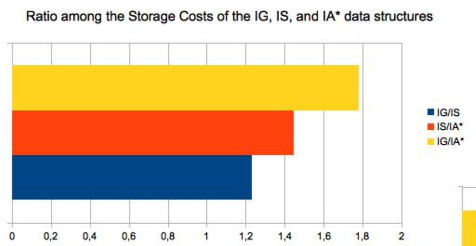
43

- Extended to quad and hexahedral meshes and more
- **Storage cost**
  - **IS:** more compact than **manifold** edge-based (50-70%) and facet-based (40-60%) data structures
  - **IA*** more compact than **IS** (see next slide)
  - **IA*: 5%** more compact than dimension-specific IAs
- T**opological connectivity queries**
  - IA* more efficient than IS on boundary (30% less) and vertex-based co-boundary queries (35% less)

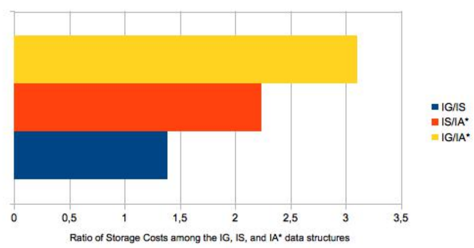
Comparison on storage costs

44

Simplicial 2-complexes

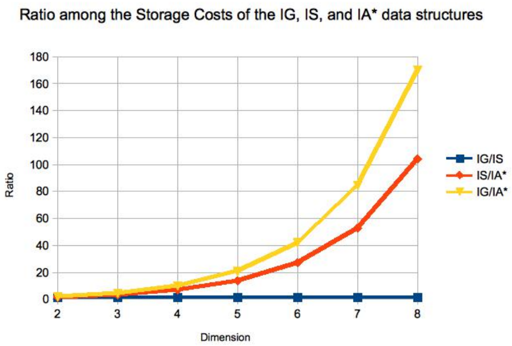


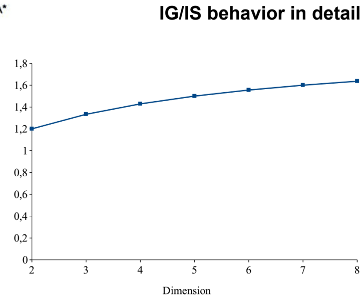
Simplicial 3-complexes



Comparison on storage costs (2)

45





The Mangrove library

46

- Rapid prototyping of data structures for simplicial and cell complexes
  - **flexible:** graph-based representation (***mangrove***) for any data structure
  - **easy to use:** simple and concise set of primitives supported
- Multi-platform, written in C++
- Implementation of five data structures
  - IS and IA* data structures
  - Incidence Graph (IG)
  - Two data structures specific for 2D and 3D simplicial complexes in 3D space
- Released as *GPLv3* software at http://mangrovetds.sourceforge.net

The Mangrove library (2)

47

- Current version:
  - arbitrary cell complexes (IG and IS data structures)
  - IS and IA* for quad and hexahedral meshes
    - Based on fixed cardinality of boundary relations
- **Topological editing operators:** under development
  - homology-preserving and homology-modifying Euler operators in arbitrary dimensions for cell complexes
  - Operators for simplicial complexes:
    - Stellar operators
    - Face collapse
    - Edge contraction

High-dimensional simplicial complexes
Data structures

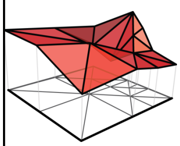
48

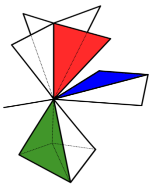
- Simplex tree [Boissonnat and Maria, 2012]
  - For abstract simplicial complexes of any dimension
  - All simplexes explicitly stored in a trie
  - Applications: construction of flag complexes and homology computation
- Tidy set [Zomorodian, 2010]
  - For simplicial sets (obtained from abstract simplicial complexes)
  - Dual graph representation of the complex:
    - nodes = top simplexes ; arcs = their intersection
  - Application: computing homology of flag complexes
- Blocker data structure [Attali et al., 2011]
  - Simplicial complexes close to clique complexes
  - Representation: 1-skeleton plus inclusion minimal simplexes

Towards localized data structures

49

- Bottleneck on future exascale computing shifts for processing costs to memory access costs
- Moving to multi-core architectures
  - Limiting factors: per-core memory size and bandwith
- Requirements are still:
  - random-access traversal operators
  - efficient updates



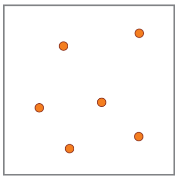


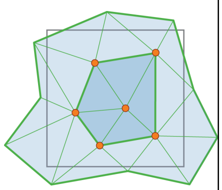
A spatio-topological approach

The PR-star octree [Weiss et al., 2011]

50

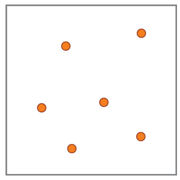
- "Topology through space"
  - topological connectivity queries through a spatial index on embedding space
- Efficient reconstruction of topological relations
  - optimal application dependent local data structures to be generated at runtime
- Our approach (PR-star Octree)
  - tetrahedral meshes
  - generalizes to complexes in arbitrary dimensions



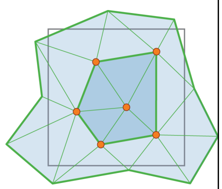


The PR-star octree

- **Strategy**
  - augment PR(Point-Region) octree (index on the vertices of the mesh) with the tetrahedra incident at its vertices

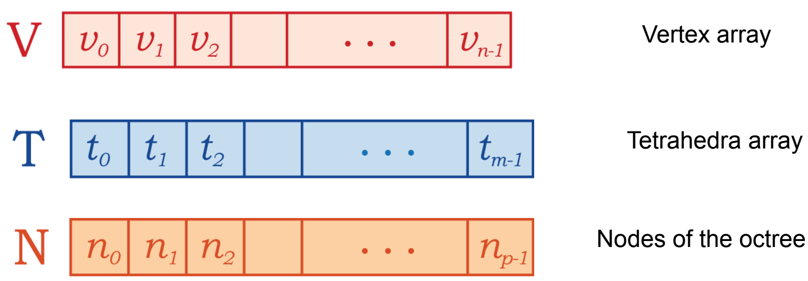


- **Data structure**
  - global indexed representation of the mesh (without adjacencies)
  - an octree node indexes a contiguous range of vertices and tetrahedra



The PR-star octree
Storage costs

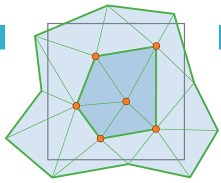
- **Topological overhead:** 19% wrt compact indexed representation with adjacencies (IA)
- **Total cost:** PR-star is 62% of IA



Applications of PR-star octree

General Strategy

53

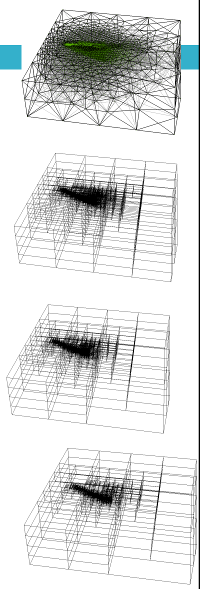


- Iterate through octree nodes
- For each leaf octree node
  - Step 1: Build application-dependent local data structure
  - Step 2: Process mesh locally
  - Step 3: Discard local data structure
- Cost of building data structures is amortized over multiple local operations

Applications of PR-star octree

Static and Dynamic

54



- **Computing the star of a vertex**:
  - ~ *70%* faster with PR-star than with IA
  - cost amortized over a large portion of the mesh
- **Mesh simplification** (through edge contraction)
  - similar simplification results in around the same amount of time as IA
  - 1% of the memory
- Successfully applied to **compute discrete Morse complexes** on tetrahedral meshes [Weiss et al., Eurovis2013] – 50% times faster than IA for Morse gradient computation

Experiments performed on irregular and semi-regular data sets containing up to 14 Millions of tetrahedra