# Non-manifold modeling system

Human-centered CAD Lab.

# Manifold model

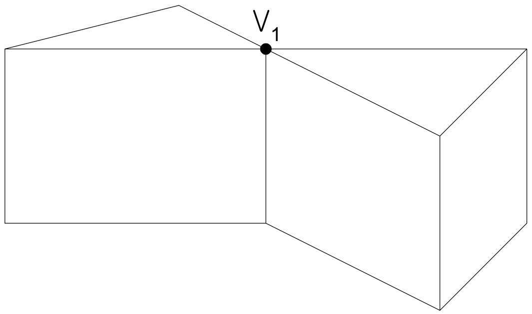
- Every point on a surface
  - Two dimensional
    ⇨every point has a neighborhood which is homeomorphic to a two dimensional disk
  - Design is carried out in abstract shape in conceptual design stage ⇨ non-manifold model
- Dimension reduction(mixed dimension) is often used in analysis
  ⇨ non-manifold model

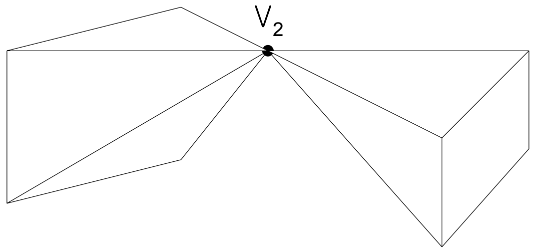
# Manifold model – cont’

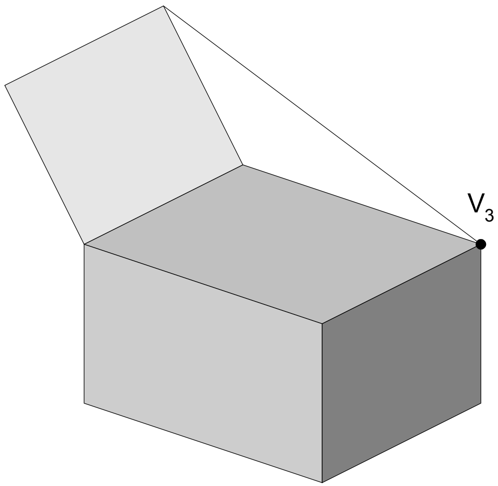
- Design process
  - Incomplete lower level description -> true solid

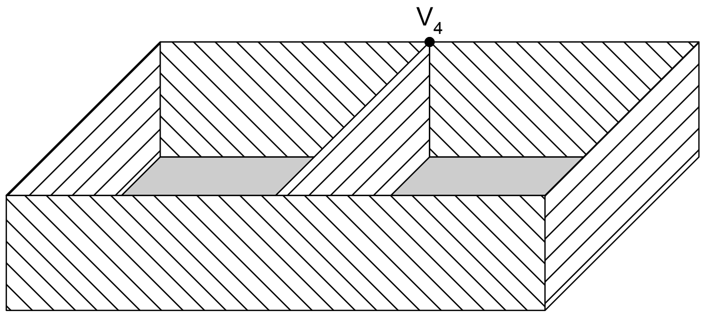
    ⇨ whole design process can be supported by non-manifold modeler
  - Non-manifold modeler
  - Wire-frame, surface, solid, cellular model, and their mixture can be handled

# Manifold model – cont'



# Manifold model – cont'

- Issues to develop a non-manifold modeler
  - Data structure
  - Euler operator
  - Modeling commands

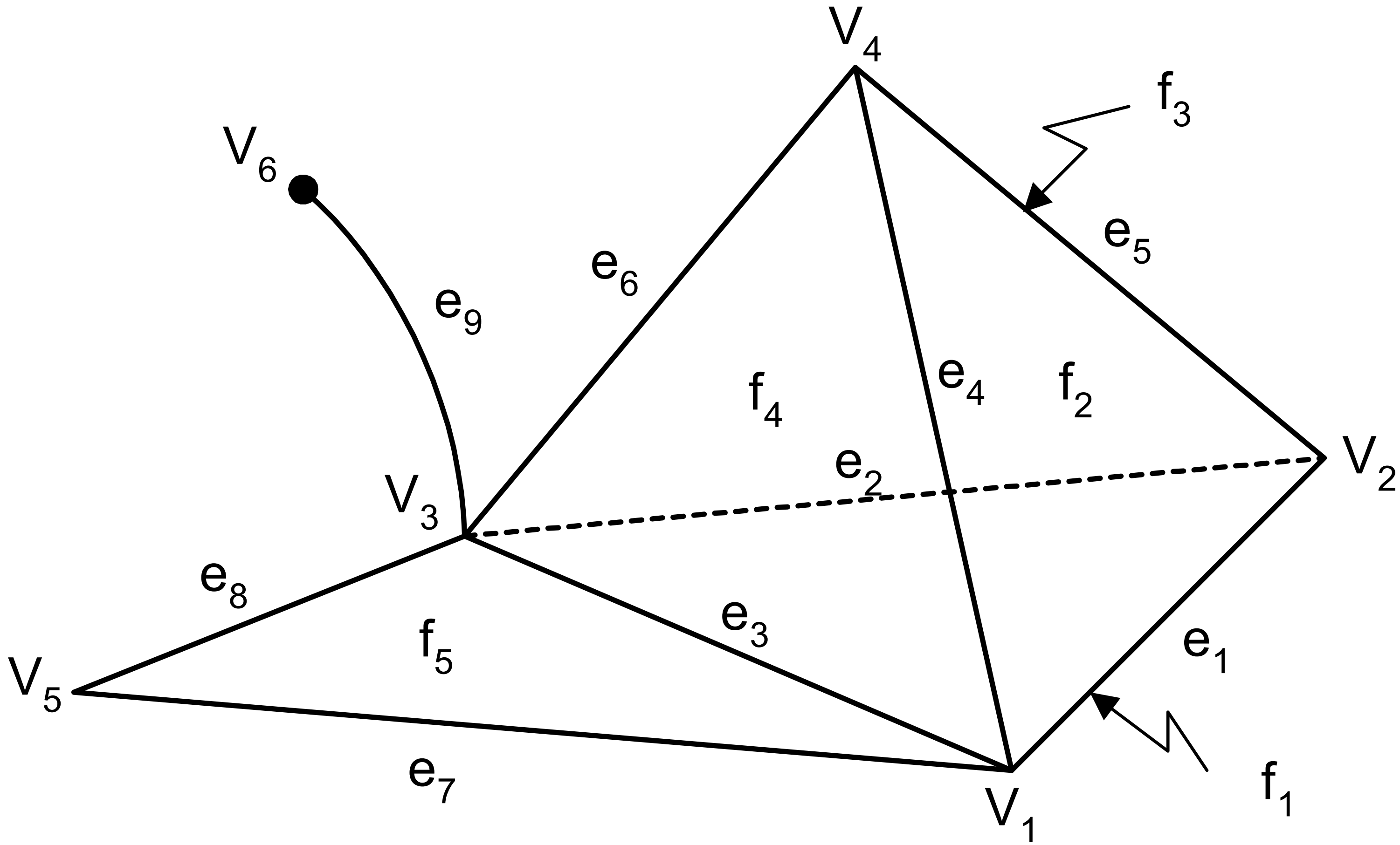
# Data structure

- Masuda's structure
  - Model is a collection of solid with volume+ laminar face + dangling edge etc.

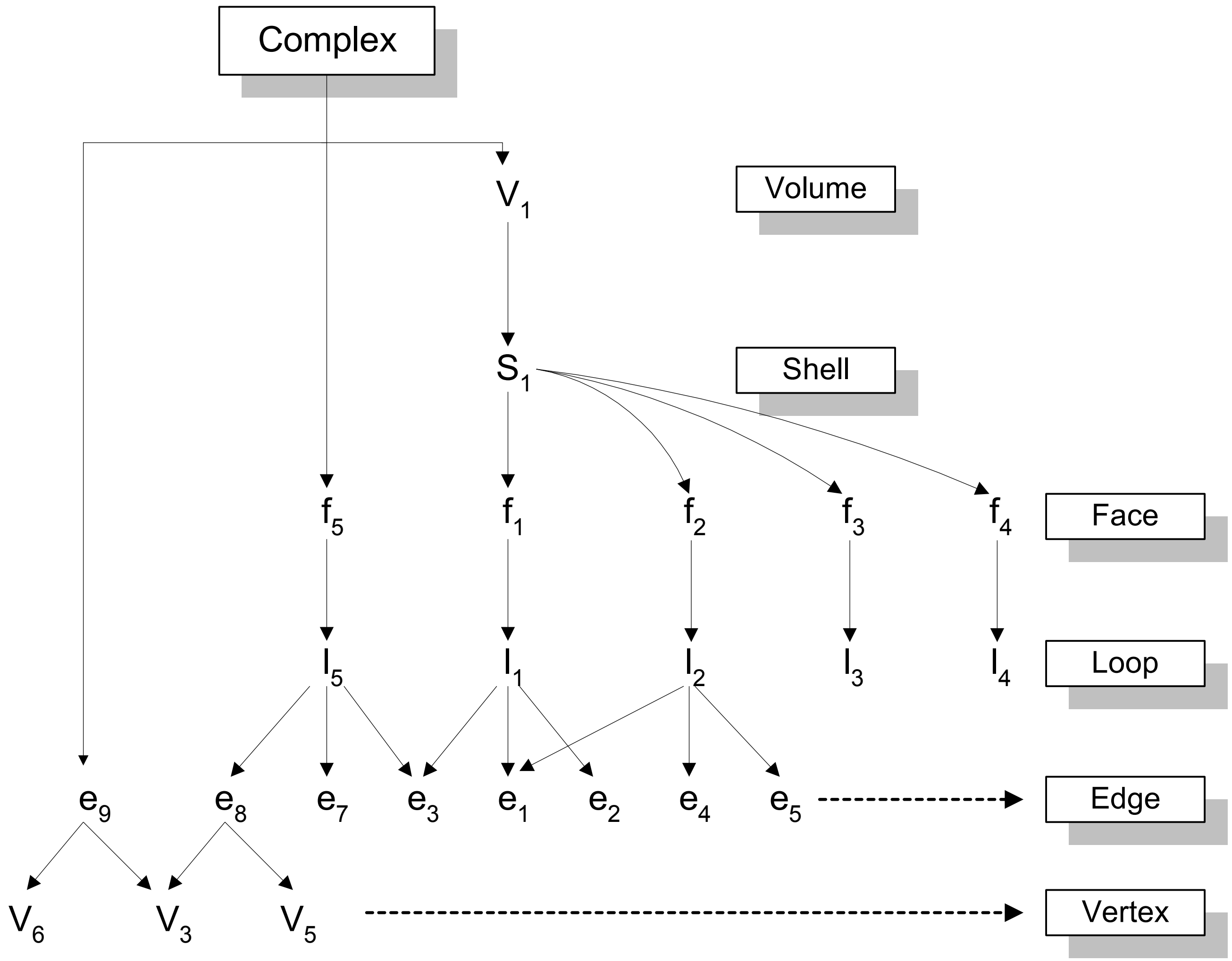
    ⇨ entity called complex stores these as a list

    ⇨ compact, easy to understand

# Data structure – cont'



# Data structure – cont’



# Data structure – cont'

- Does not provide adjacency information between topological entities
- Insufficient information on adjacency between faces
  - Difficult to determine whether a shell  S1 should be split into two or a laminar face should be added when a new face is added
  - Can be determined by geometric calculations but inefficient

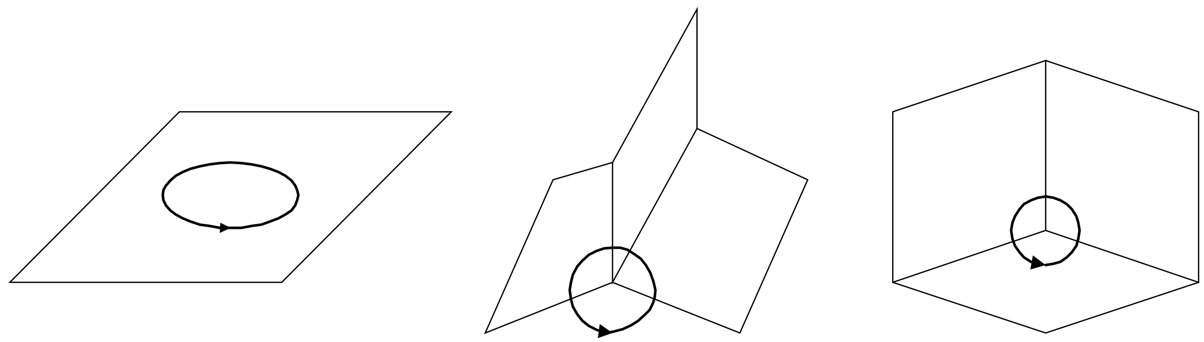
# Data structure – cont'

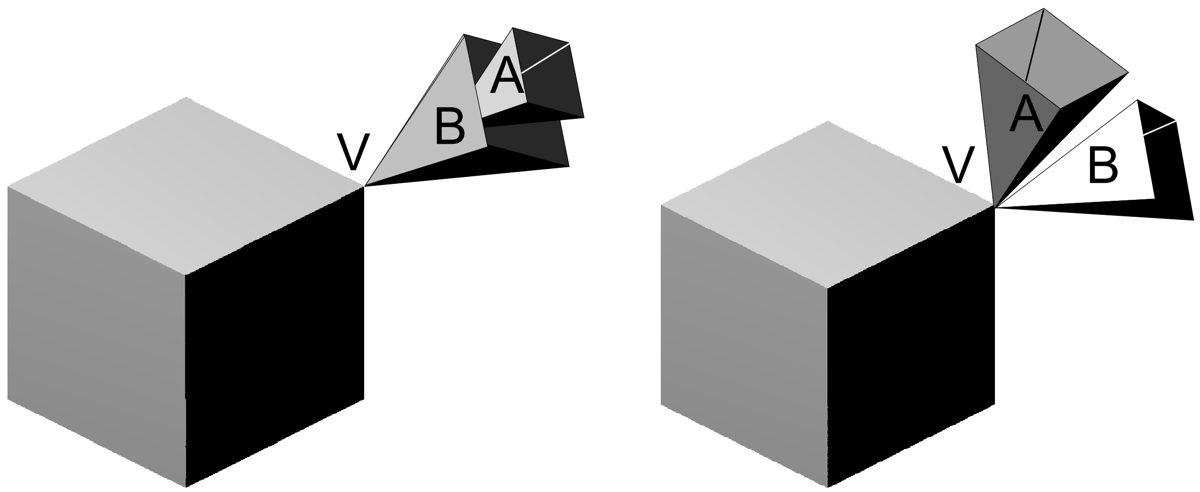
- Weiler's radial edge data structure for adjacency, store cyclic ordering
- In manifold model, 2 cycles are enough
  - face-edge cycle : list of edges for each loop (face)
  - vertex-edge cycle : list of connected edges at a vertex

# Data structure – cont'

- In non-manifold model, 3 cycles are necessary
  - loop cycle : face-edge cycle
  - radial cycle : cycle of faces connected to a certain edge (in manifold models, there are two faces all the time )
  - Disk cycle :
  - -> similar to vertex-edge cycle
  - -> one vertex may have many disk cycles

# Data structure – cont'





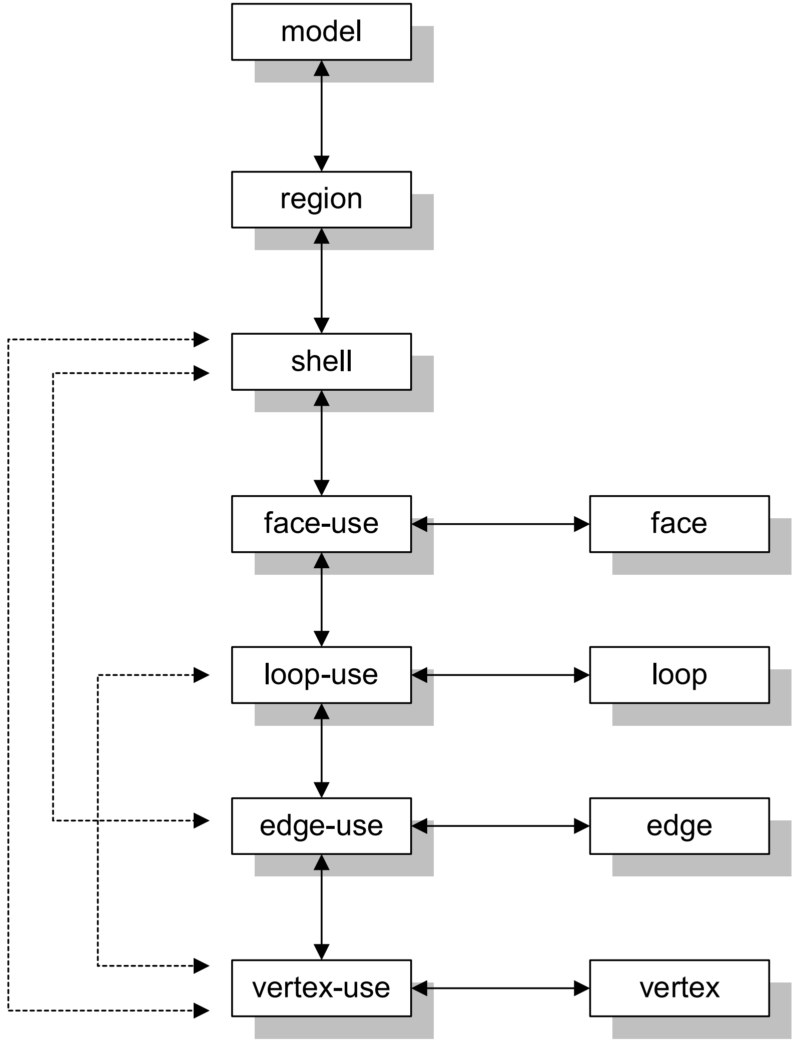
# Data structure – cont'

- Weiler's radial edge representation
  - Stores radial cycle and loop cycle, not disk cycle
  - ⇨ models in the previous slide are represented by the same data
- Can be differentiated by geometric calculations, but inefficient
- In Gursoz and choi's Vertex Based Representation, disk cycle is stored explicitly

# Radial edge data structure

- Model, region, shell, face, loop, edge, vertex
  - Model = complex (group of regions) ,
  - Region = volume (bounded by one or several shells)
  - Single solid has two regions : Inside, outside
  - Shell = boundary of region

# Radial edge data structure – cont'



# Radial edge data structure – cont'

- Model :
  - Equivalent to complex, simply means the group of all the regions involved
- Region :
  - Equivalent to the volume and bounded by one or several shells
  - Single solid has two regions, one for the inside and the other for outside
- Shell :
  - Oriented boundary surface of a region

# Radial edge data structure – cont’

- Additional topological entities
  - face-use, loop-use, edge-use, vertex-use
  - Similar to representing the connectivity between loops, vertices, and edges through half-edges indirectly
- face-use: inside and outside of a face, use of a face by a shell
- loop-use: boundary of face-use, same direction as face-use
- edge-use: boundary of loop-use
- vertex-use: boundary of edge-use

# Radial edge data structure – cont’

- shell <-> edge-use
  - wire-frame model
- shell <-> vertex-use
  - shell consists of a single vertex
- loop-use <-> vertex-use
  - isolated point on a face as its hole loop

# Radial edge data structure – cont’

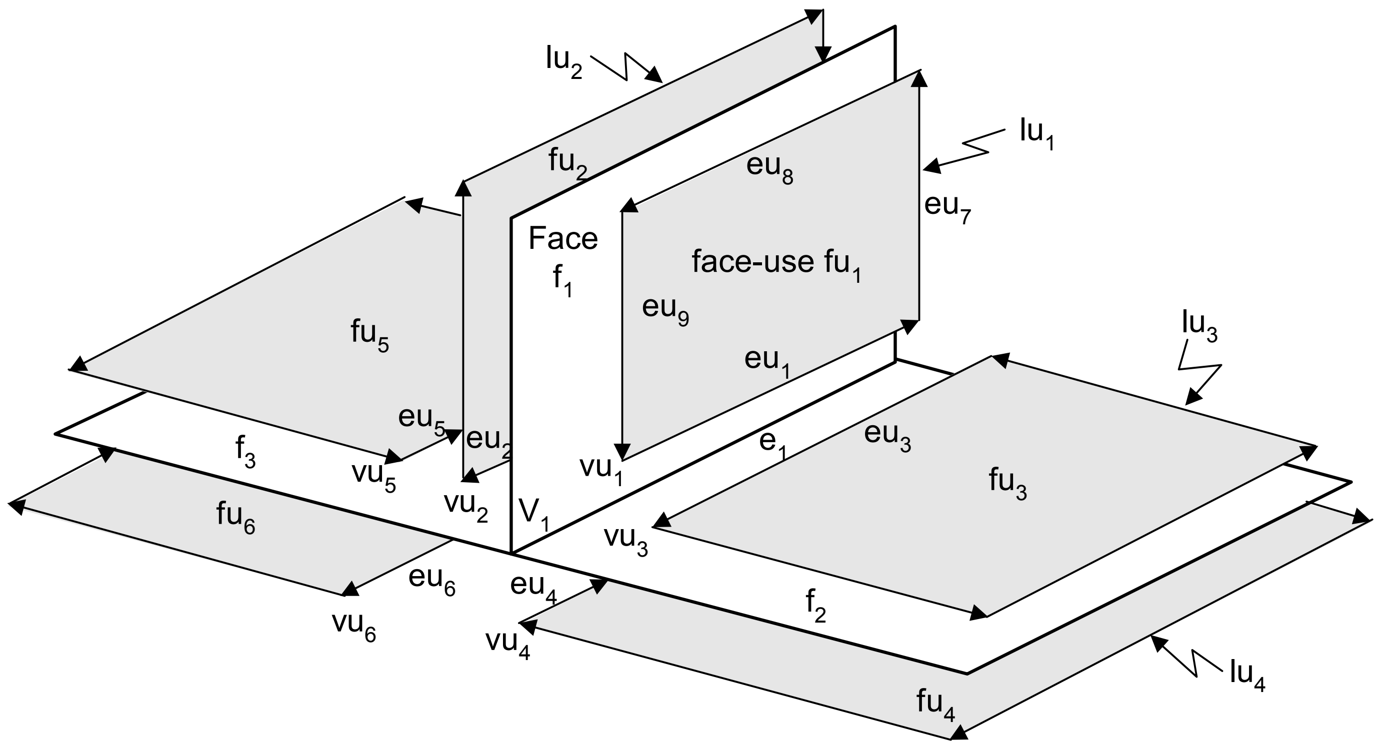
- Edge-use
  - Radial pointer points to the edge-use on the face-use radially adjacent to the face-use of the given edge-use

    ⇨ radial ordering of faces about an edge can be derived by tracing the pointers

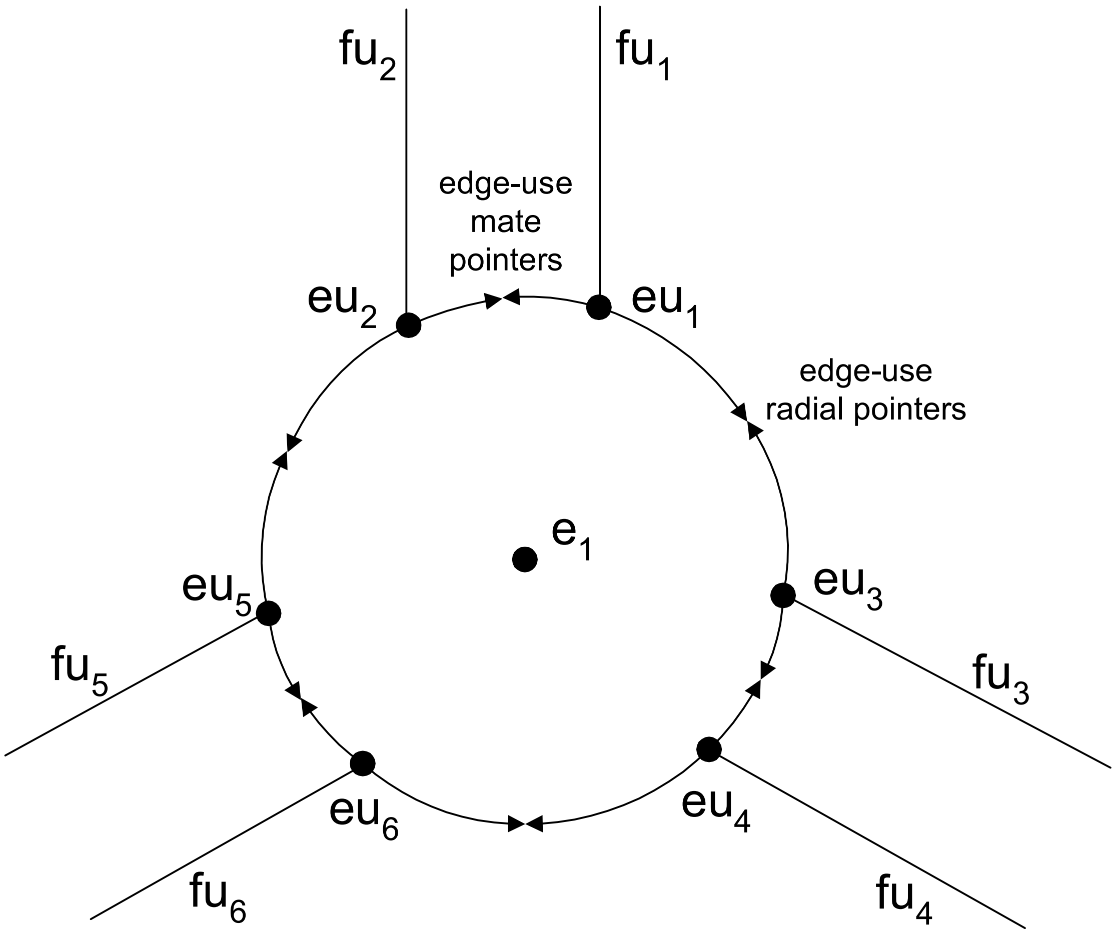
# Radial edge data structure – cont'

- vertex-uses
  - Stores vertex-edge cycle
  - By storing all the vertex-use for a vertex, all the edge-use connected to the vertex can be derived
  - disk cycle is not stored explicitly

# Radial edge data structure – cont’



# Radial edge data structure – cont'



# Operators to manipulate topology

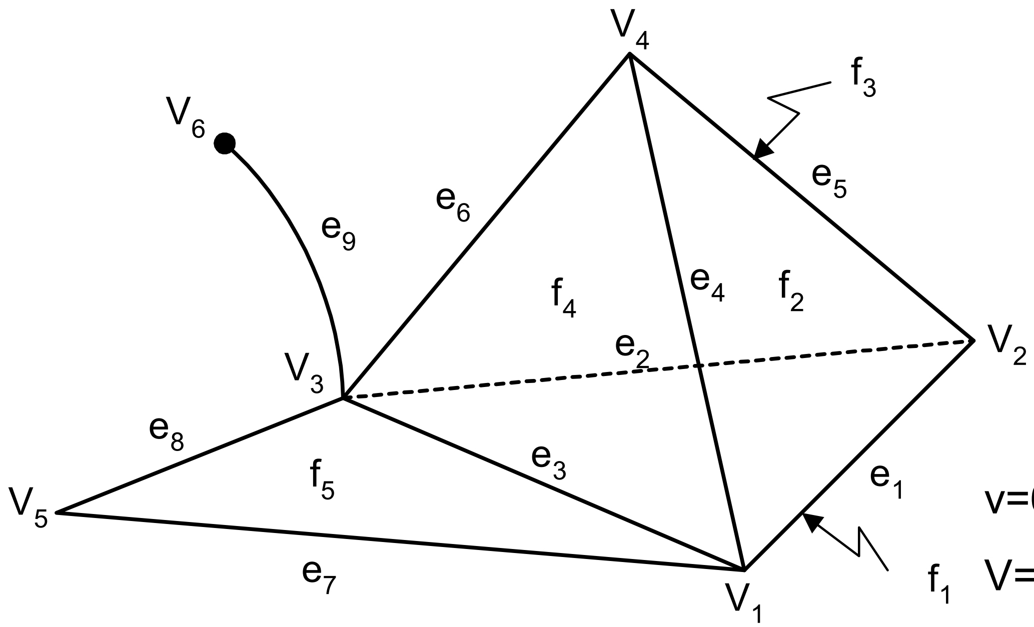
- Generalized Euler-Poincare formula for non-manifold models

- $$v - e + (f-r) - (V-Vh+Vc) = C- Ch +Cc$$

    - r = rings or holes in faces
    - V = number of closed volumes
    - Vh = passages through the volumes
    - Vc = voids in the volumes
    - C = number of complexes, or disjoint objects
    - Ch = passages through objects
    - Cc = cavities and voids in the objects

# Operators to manipulate topology – cont’

- Each complex can be composed of several volume and dangling edge




$v=6,\ e=9,\ f=5,\ r=0$

$V=1,\ V_h=0,\ V_c=0,\ C=1,\ C_h=0,\ C_c=0$

$6 – 9 + (5-0) – (1-0+0) = 1 – 0 + 0$

# Operators to manipulate topology – cont'

- For manifold case,

$$V = C,\ Vh = Ch,\ Vc = Cc$$

$$V - e + (f\text{-}r) - (V\text{-}Vh+Vc) = V - Vh + Vc$$

$$\therefore\ v - e + (f\text{-}r) = 2(V\text{-}Vh+Vc)$$

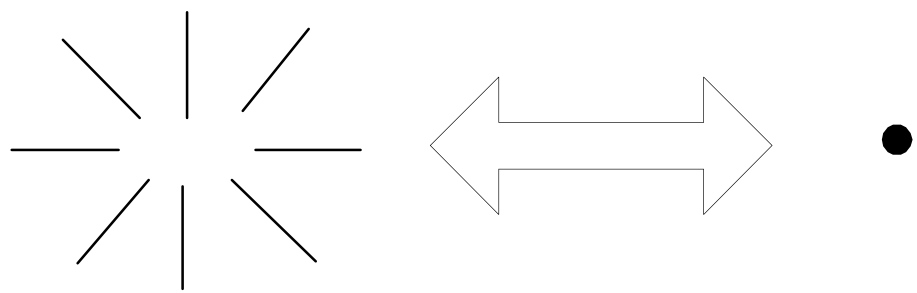
- number of shells equals V + Vc

$$\therefore\ v - e + (f\text{-}r) = 2(s\text{-}Vh) \quad (r = h,\ Vh = p)$$

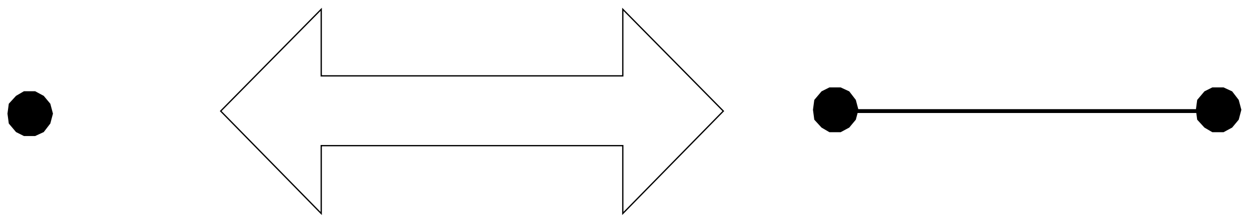
- Nine independent base vectors
  ⇨ nine operators

# Operators to manipulate topology – cont'

- mvC (kvC): make(kill) vertex, complex

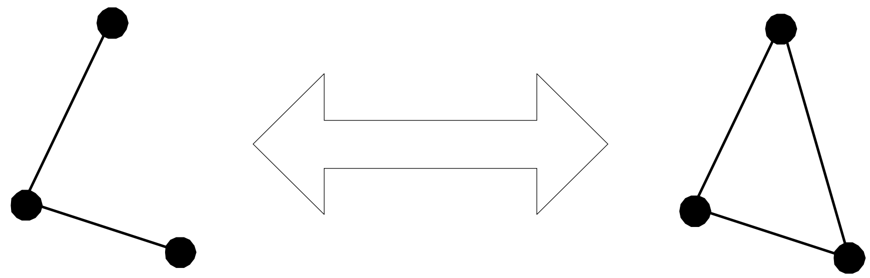


- mev (kev): make(kill) edge, vertex

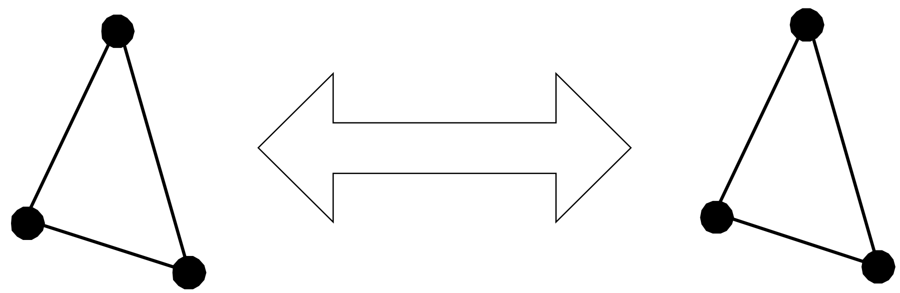


# Operators to manipulate topology – cont'

- meCh (keCh): make(kill) edge, complex_hole

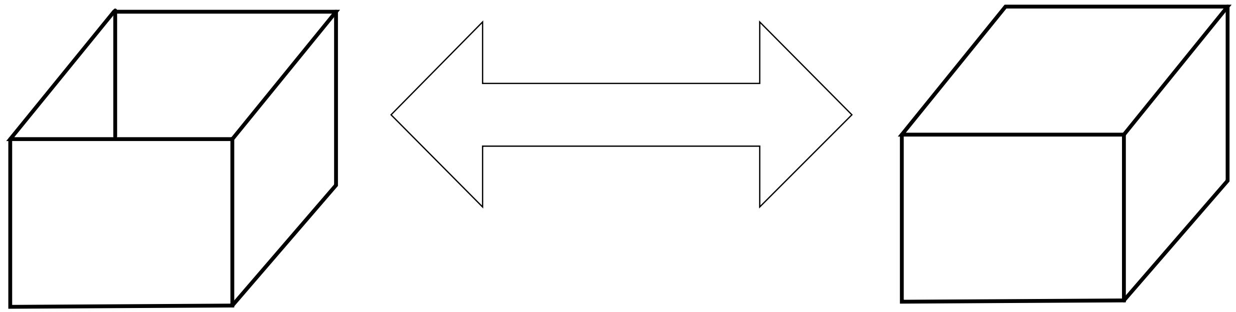


- mfkCh (kfmCh): make(kill) face, kill(make) complex_hole



# Operators to manipulate topology – cont'

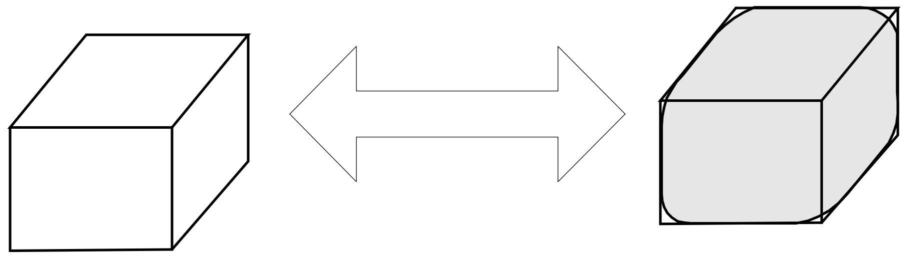
- mfCc (kfCc): make (kill) face, complex_cavity



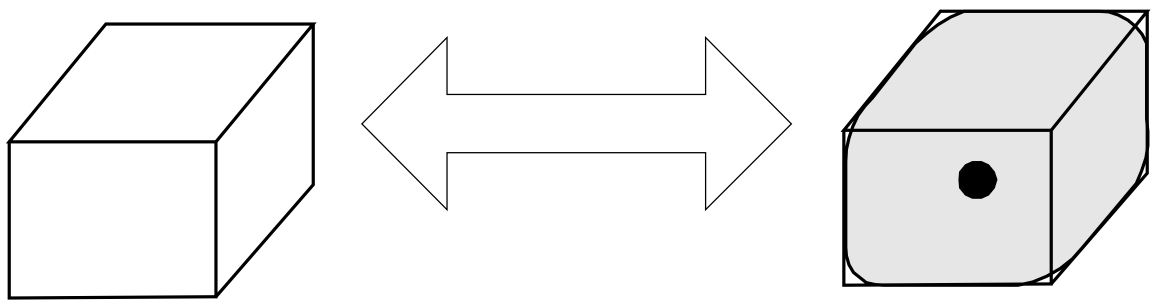
- mvr (kvr): make (kill) vertex, ring

# Operators to manipulate topology – cont'

- mVkCc (kVmCc): make (kill) Volume, kill (make) complex_cavity

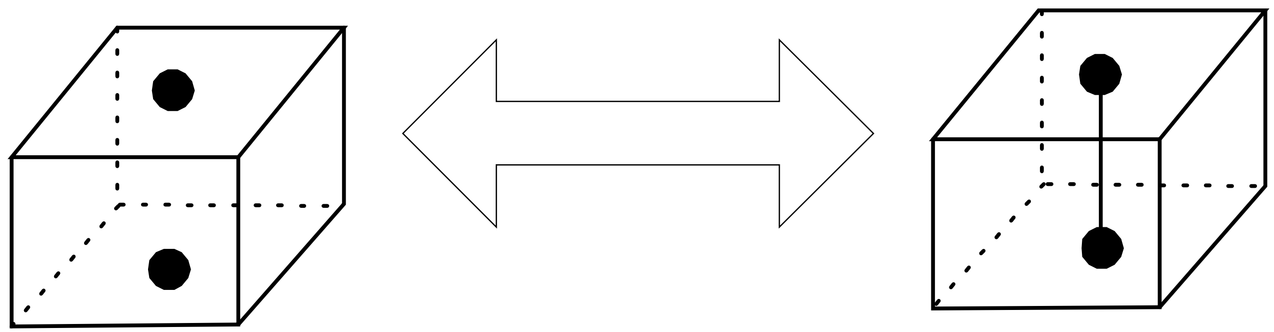


- mvVc (kvVc): make (kill) vertex, Volume_cavity



# Operators to manipulate topology – cont'

- meVh (keVh): make (kill) edge, Volume_hole



# Operators to manipulate topology – cont’