# The gem data structure for $d$-dimensional colored triangulations

Jorge Stolfi
Instituto de Computação
Universidade Estadual de Campinas (UNICAMP)
13084-971 Campinas, SP, Brasil
stolfi@ic.unicamp.br

Talk at Meshing Workshop, 29th ACM SOcG, 2013-06-18.

Joint work with Arnaldo Jovanini Montagner and Lucas Moutinho Bueno

June 16, 2013

**Abstract**

We describe in detail a novel data structure for $d$-dimensional triangulations. In an arbitrary $d$-dimension triangulation, there are $d!$ ways in which a specific facet of an simplex can be glued to a specific facet of another simplex. Therefore, in data structures for general $d$-dimensional triangulations, this information must be encoded using $\lceil \log_2(d!) \rceil$ bits for each adjacent pair of simplices. We study a special class of triangulations, called the *colored triangulations*, in which there is a only one way two simplices can share a specific facet. The *gem data structure*, described here, makes use of this fact to greatly simplify the repertoire of elementary topological operators.

# The gem data structure for $d$-dimensional colored triangulations

Arnaldo Jovanini Montagner, Lucas Moutinho Bueno and Jorge Stolfi

Computer Science Institute (IC)

State University of Campinas (UNICAMP)

Campinas, SP, Brasil

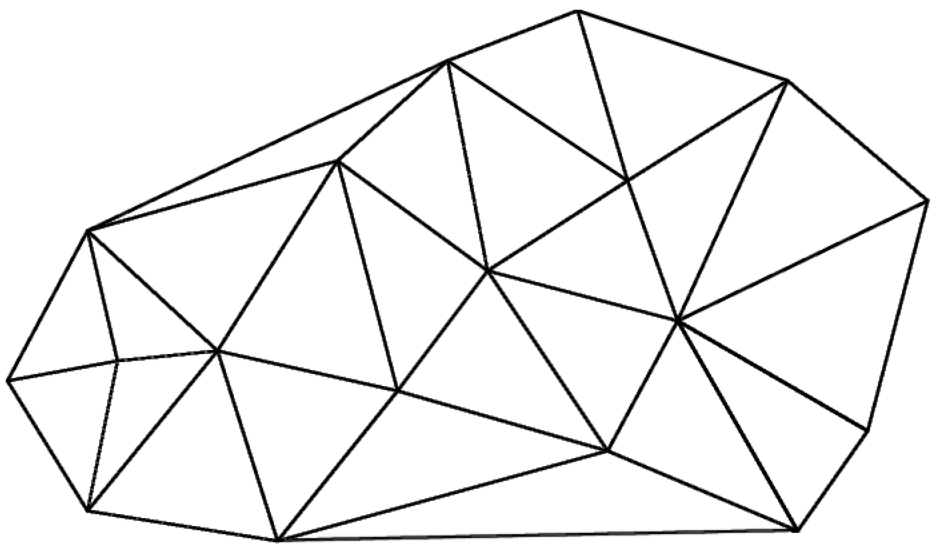
stolfi@ic.unicamp.br

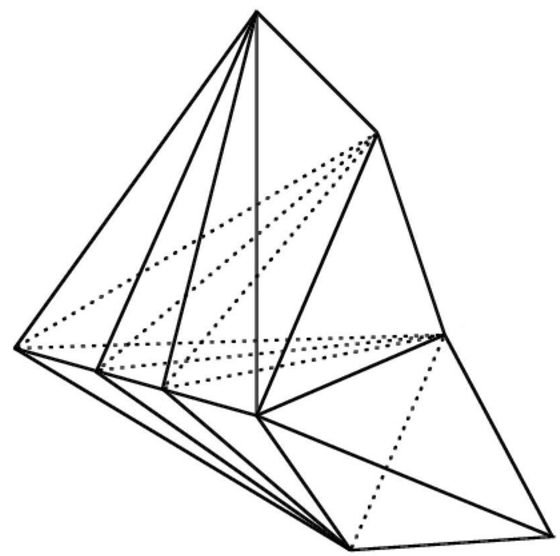
# SUMMARY

- Triangulations and their data structures
- The gem data structure.
- Relation to other structures.
- Subdivision schemes for gems.
- Turning a triangulation into a gem.
- Conclusions and future work.

# Triangulations

*Triangulation*: set of $d$-simplices, glued by facets.

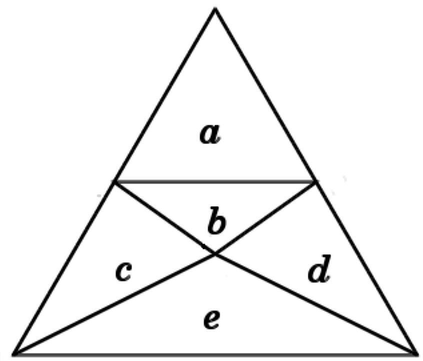


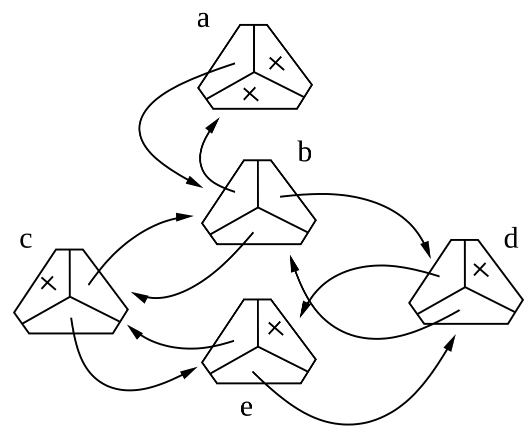


# Triangulation data structures

## Pointer data structures:

- One record per cell.
- One pointer per facet, to adjacent cell.



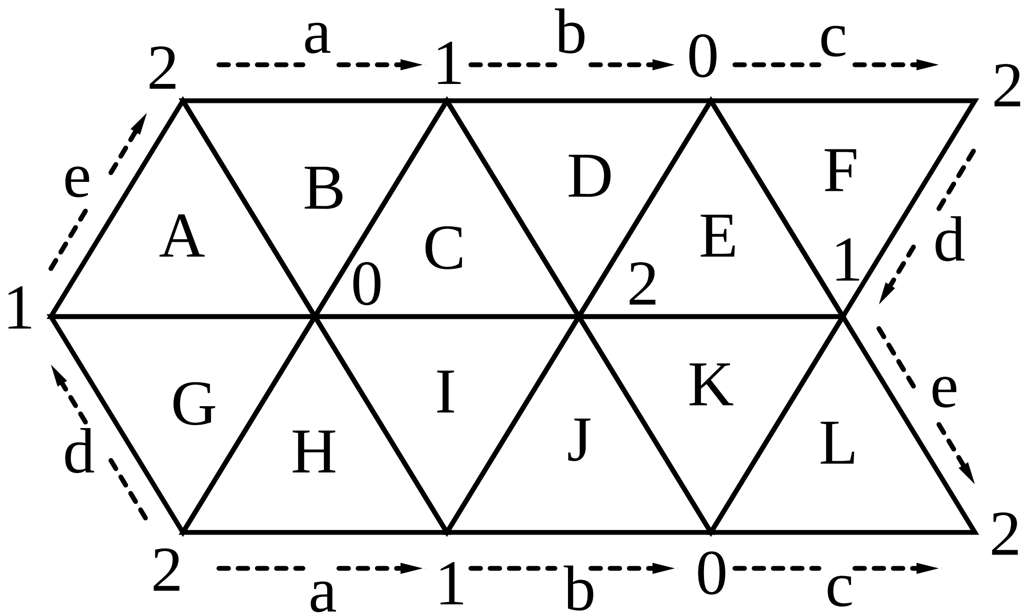

## Problem: which pointer is the right one?

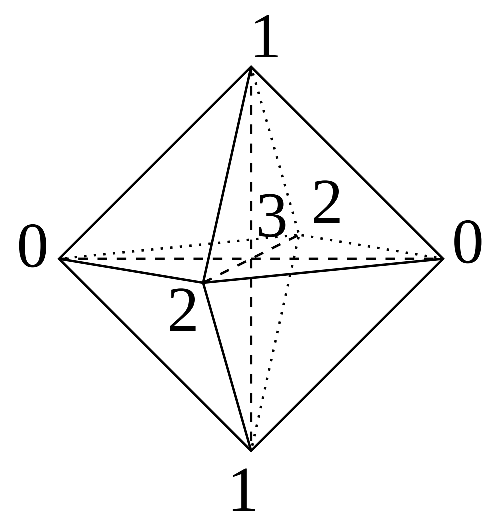
- Check all links (D. T. Lee & B. J. Schachter 1980 [7]).
- Add $\lceil \log_2((d+1)!) \rceil$ permutation bits per link
  (J. R. Shewchuck 1996 [10], J.-D. Boisonnat & al. 2002 [1], . . . )

# Colored triangulations

Colored $d$-dimensional triangulation:

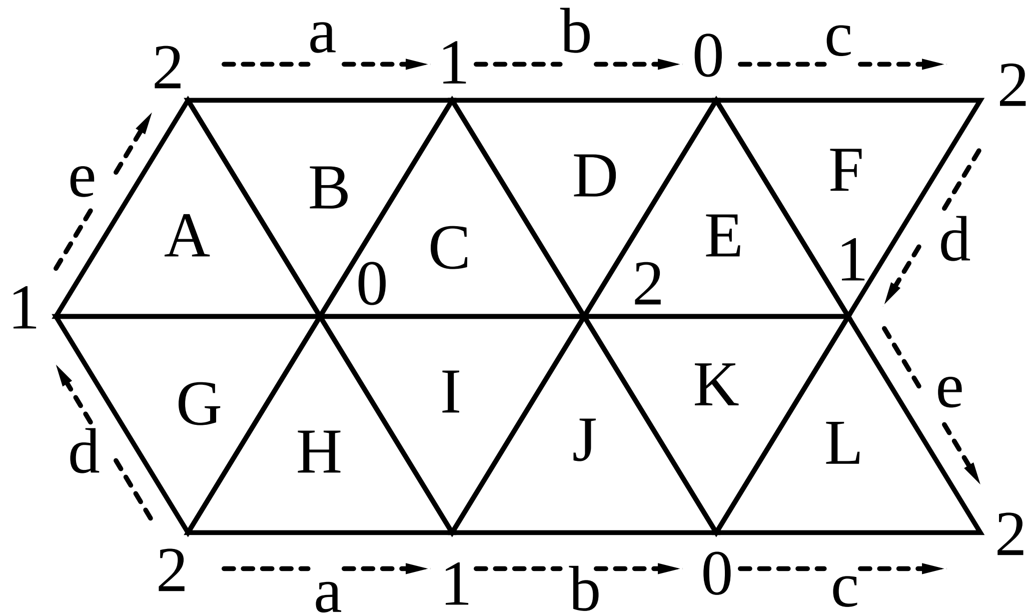
- Vertices are labeled with $d$ "colors" $0, 1, \ldots, d$.
- Each element (simplex) has at most one vertex of each color.

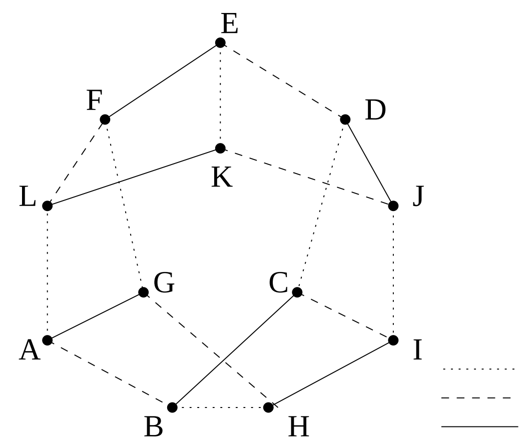


# Gems (1)

*Gem* $=$ the dual graph of a colored triangulation:

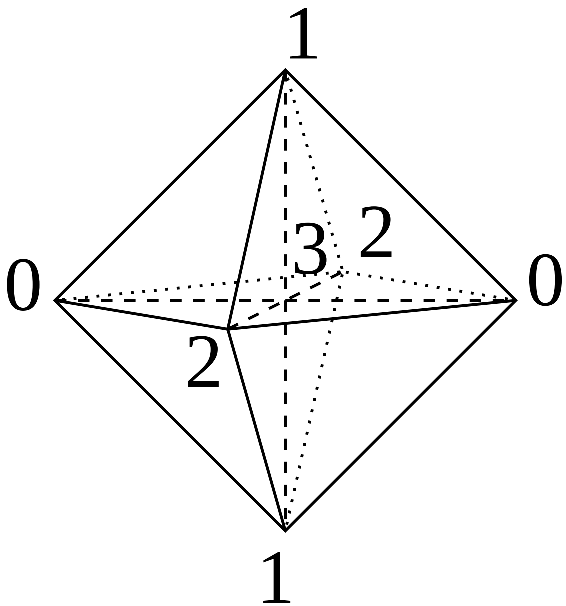


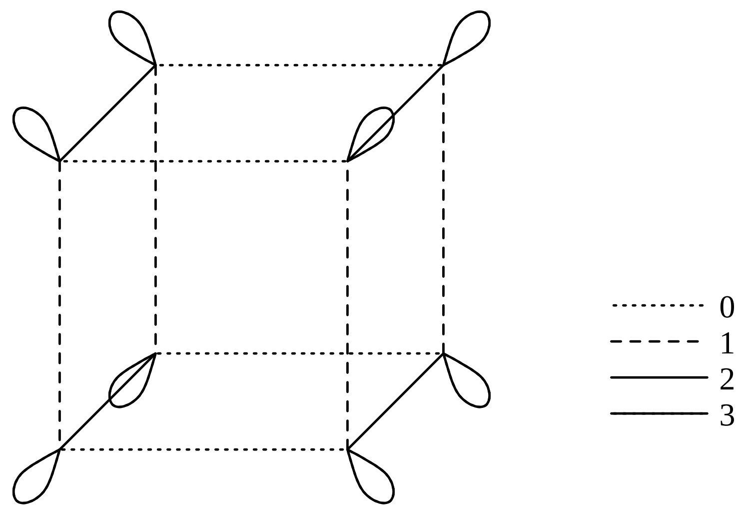




A regular graph, edge-colored with colors $0, 1, \ldots, d$.
(M. Ferri 1976 [5], S. Lins 1982 [9].

# Gems (2)

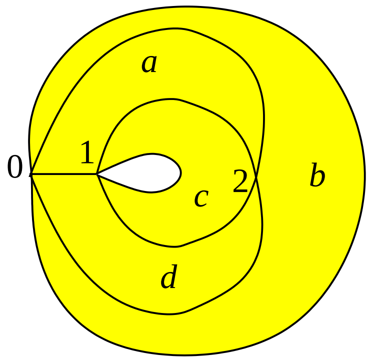
Self-loops denote unglued cell facets (free border).

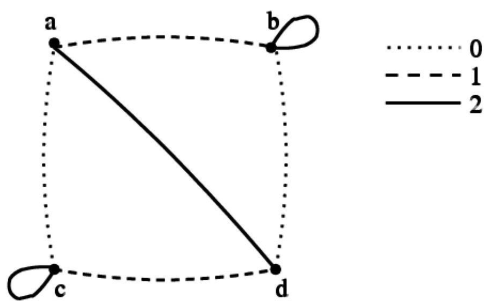


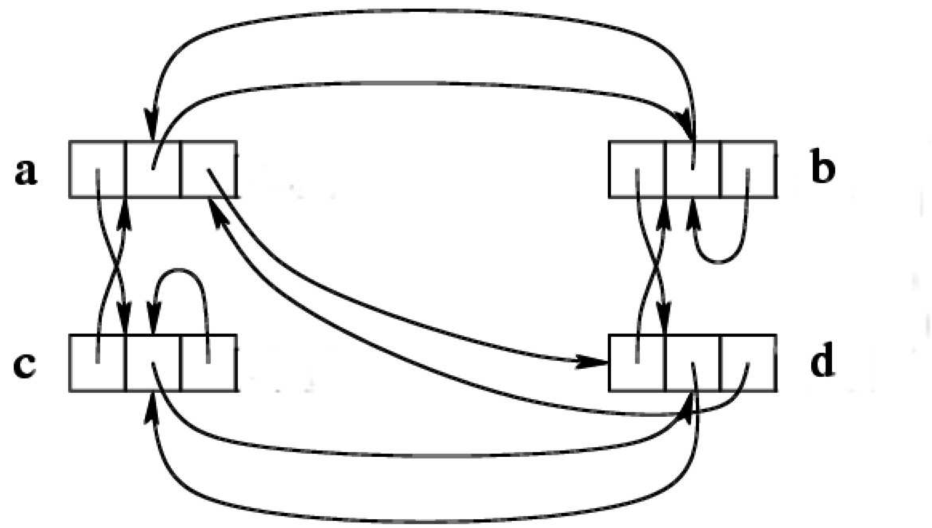



## Data structure

The *gem* data structure:



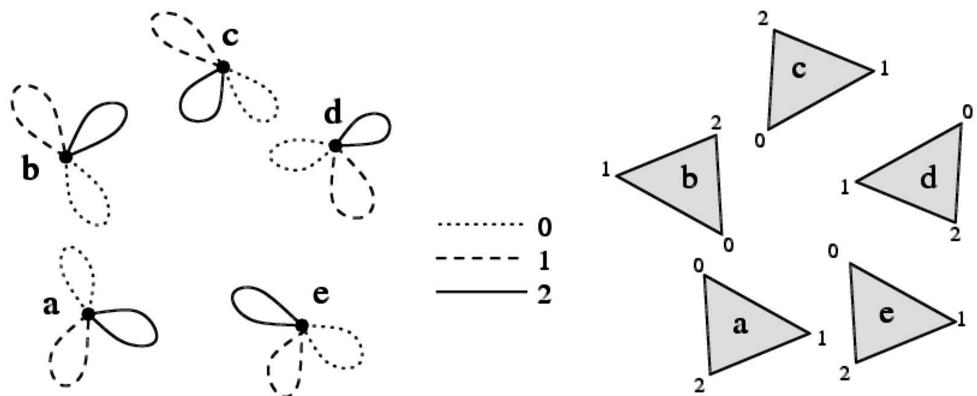




$\texttt{Step}(a,i) \;=\; \phi_i(a) \;=\;$ follow pointer $i$ of node $a$.

# Gem structure operations: Makenode

`Makenode()` creates an unattached simplex:

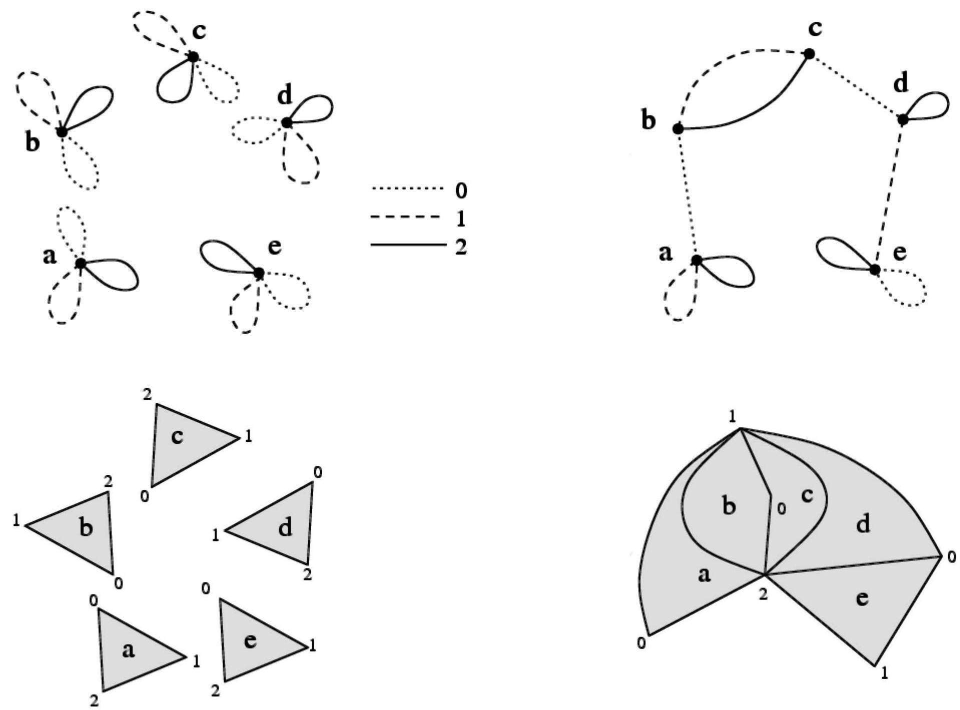
```
a = MakeNode();
b = MakeNode();
c = MakeNode();
d = MakeNode();
e = MakeNode();
```



# Gem structure operations: Swap

`Swap(`$a$`,`$b$`,`$i$`)` exchanges the $i$-pointers of $a$ and $b$:

```
Swap(a,b,0);
Swap(d,e,1);
Swap(d,c,0);
Swap(b,c,1);
Swap(c,b,2);
```

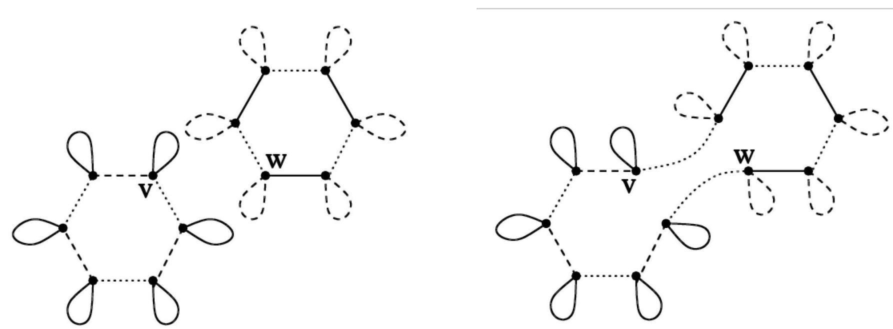



Unsafe - to be used by authorized personnel only!

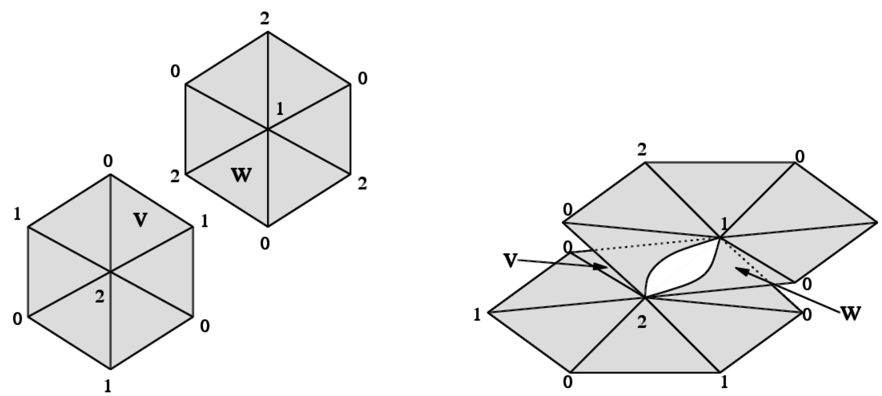
# Gem structure operations: Splice

`Splice`$(a,b,i)$ exchanges four pointers of color $i$:

**Splice**$(v,w,i)$**:**
    $v' \leftarrow \phi_i(v)$**;**
    $w' \leftarrow \phi_i(w)$**;**
    **Swap**$(v',w',$**i);**
    **Swap**$(v,w,$**i).**




`Splice`$(v,w,$`0`$)$:

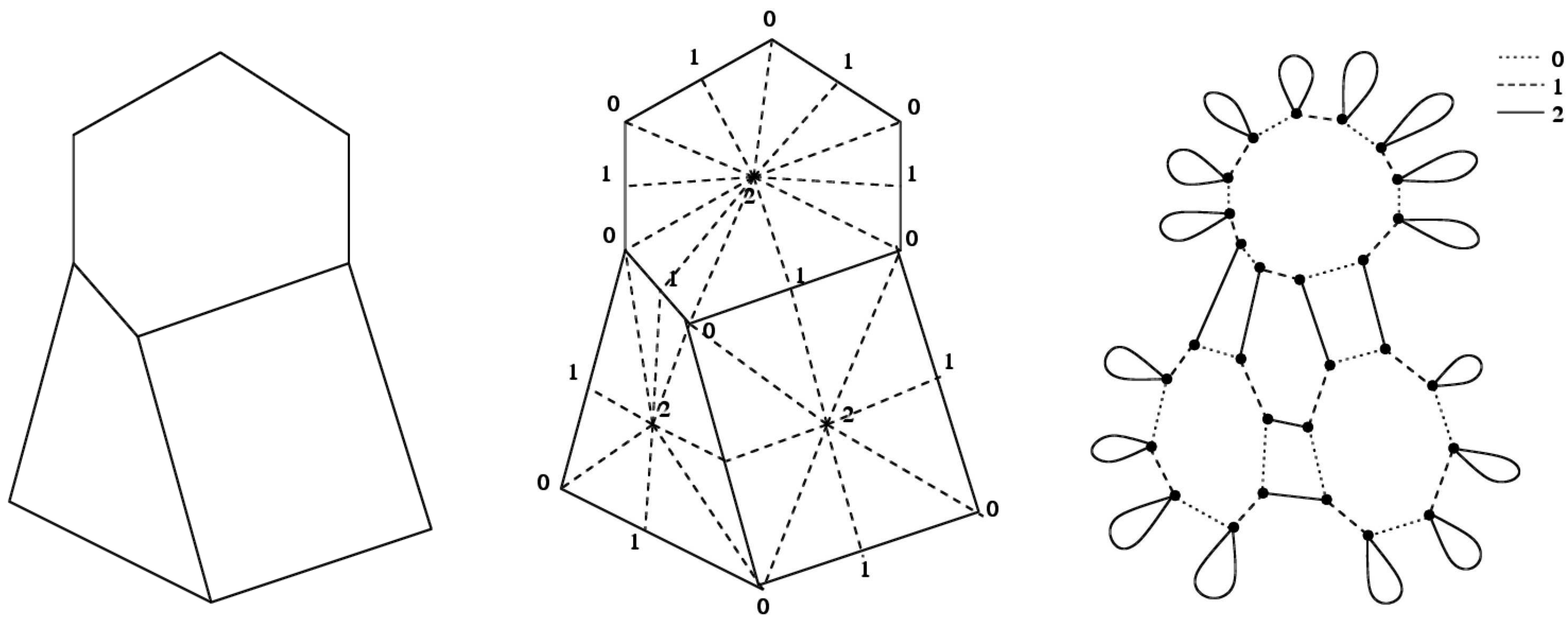



Safe for any parameters!

## Barycentric subdivision

Barycentric gems and representation of general maps:

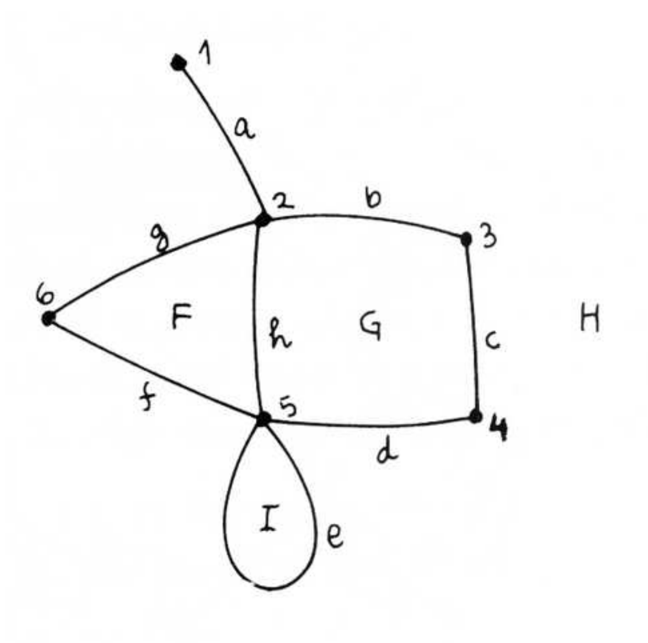
- $n$-G-maps (P. Lienhardt 1989 [8]).
- Cell-tuple structure (E. Brisson 1989 [2]).

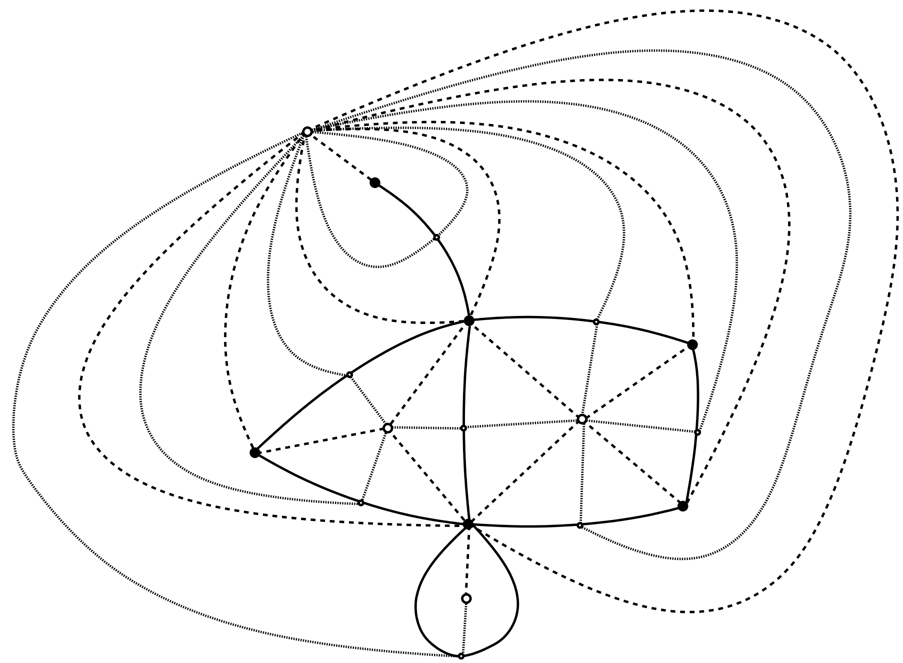


# Relationship to quad-edge structure (1)

Quad-edge data structure for 2D maps

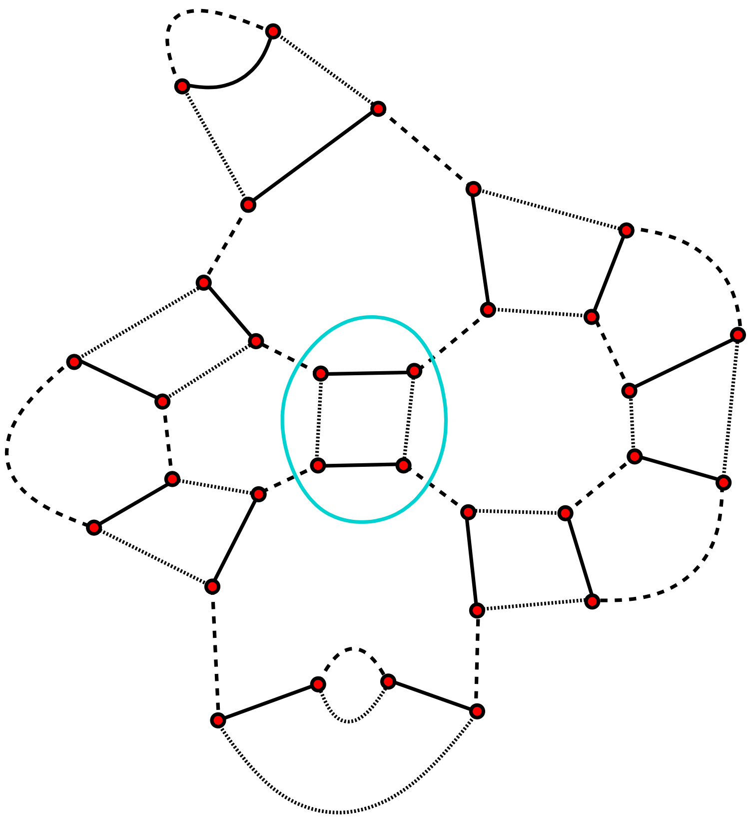
- L. J. Guibas and J. Stolfi 1985 [6].

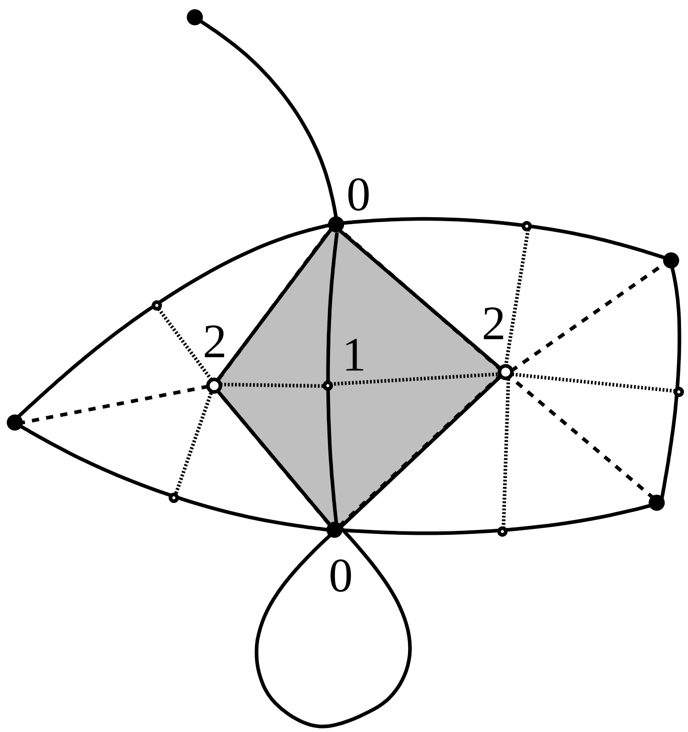


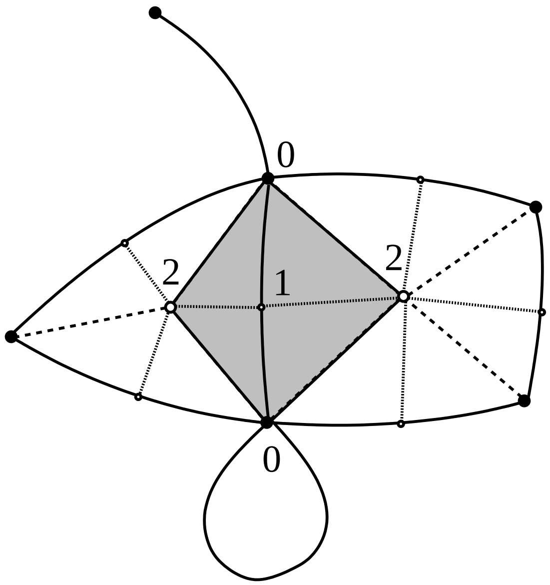
# Relationship to quad-edge structure (2)

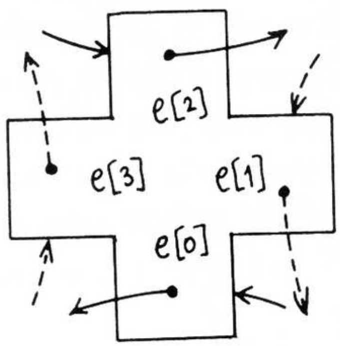
The barycentric gem partitions into $(0, 2)$-colored squares:

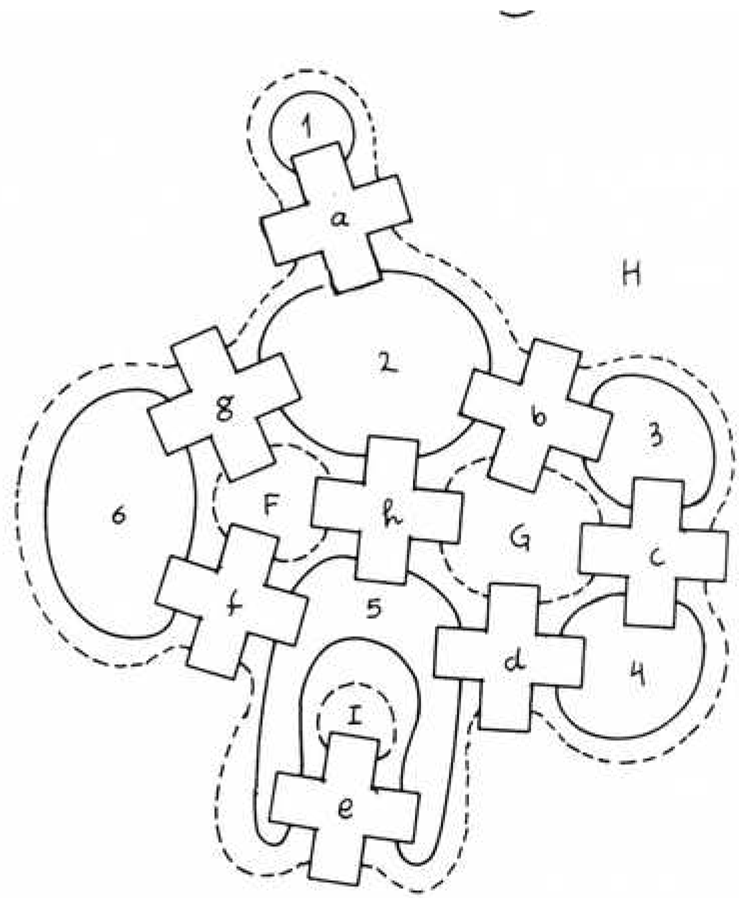




# Relationship to quad-edge structure (3)



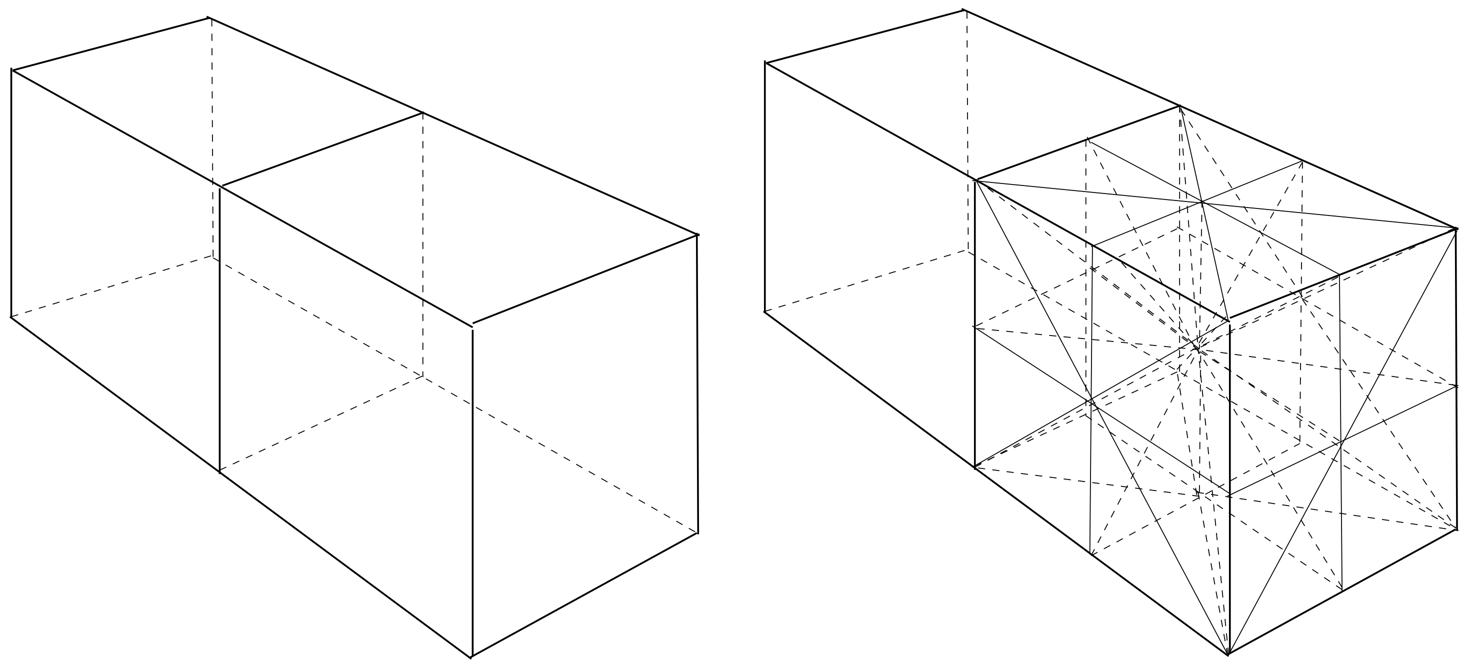



# Relationship to facet-edge structure (1)

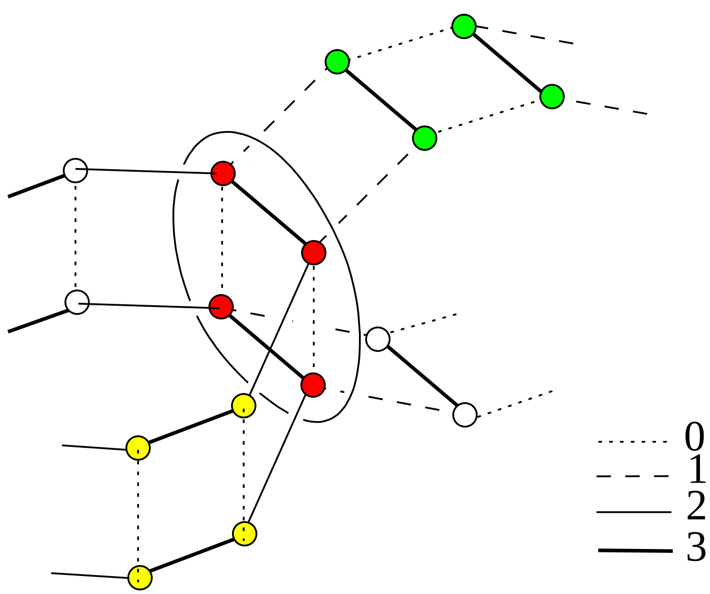
Facet-edge data structure for 3D maps

- D. P. Dobkin and M. J. Laszlo 1987 [4].

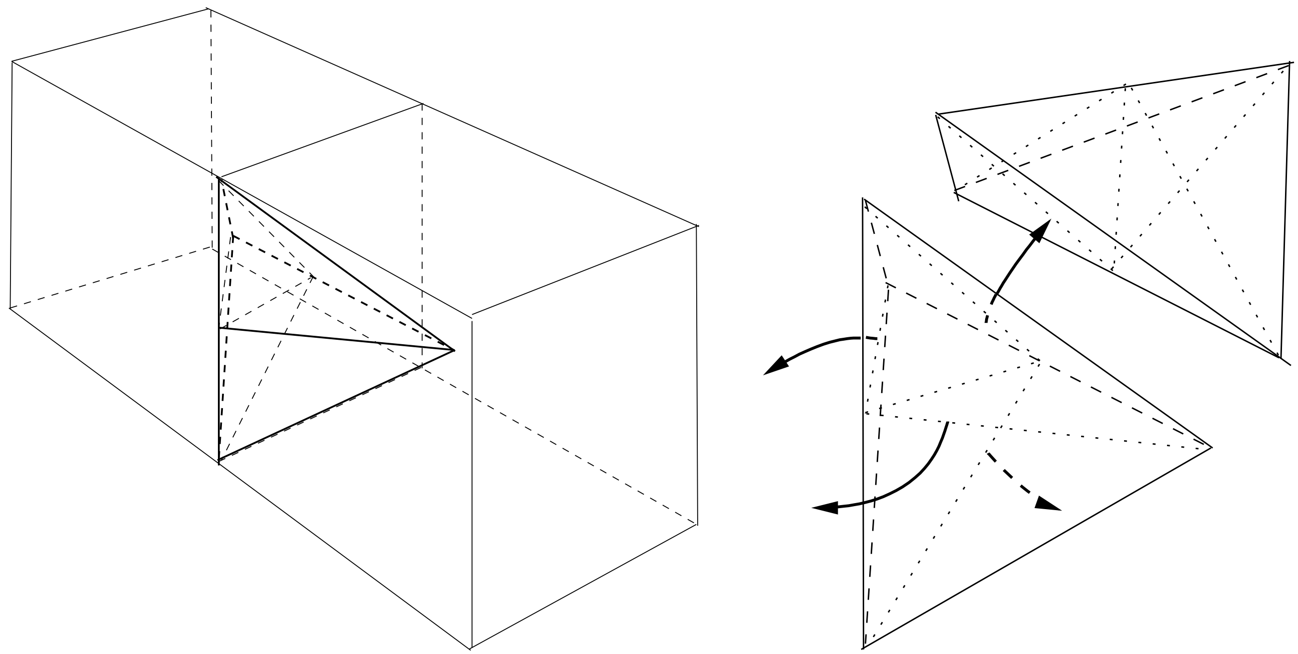


# Relationship to facet-edge structure (2)

The barycentric gem partitions into $(0, 3)$-colored squares:



# Relationship to facet-edge structure (3)



# Generalizing quad-edge/facet-edge

Barycentric gem property
(Lienhardt's $n$-G-map axiom 2 [8]):

$$\phi_i \phi_j \; = \; \phi_j \phi_i \qquad \text{if } |i-j| \geq 2$$

## Generalizes quad-edge/facet-edge for $d$ dimensions!

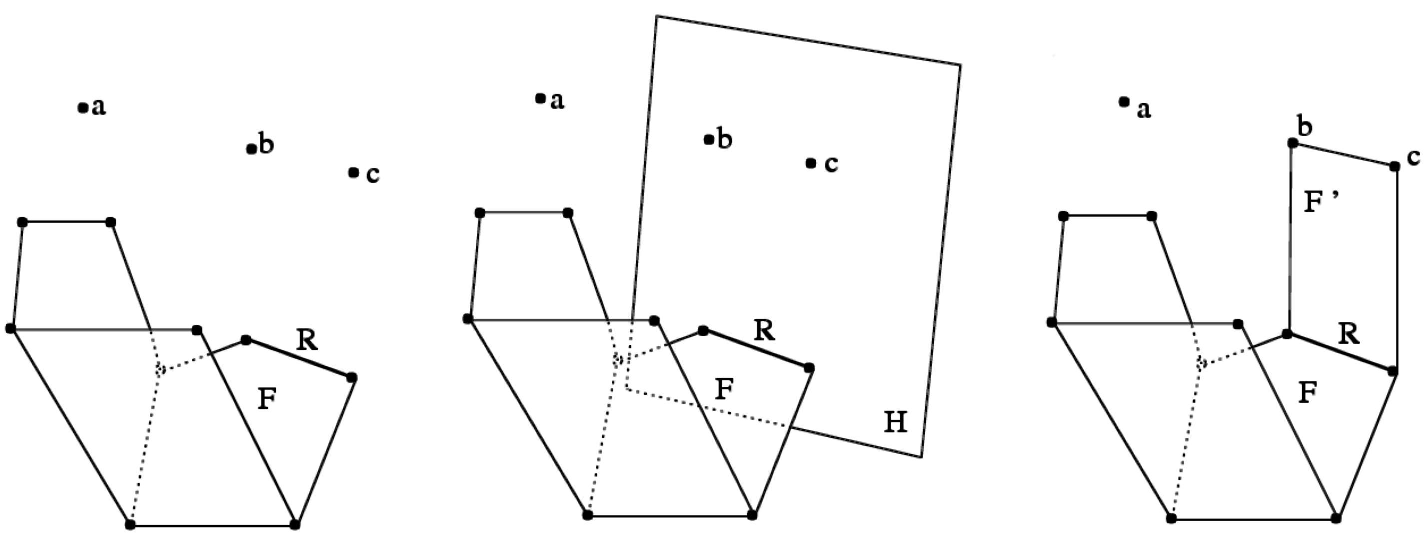
Example for $d = 7$:

- edges colored $0, 2, 5, 7$ comprise disjoint $4$-cubes.
- store 16 nodes as 16 parts of same record.
- add 4 bits per pointer to identify which part.
- $\phi_0, \phi_2, \phi_5, \phi_7$ need no pointers.
- save a few more pointers using $\phi_6 = \phi_2 \phi_6 \phi_2$.
- structure supports duality.

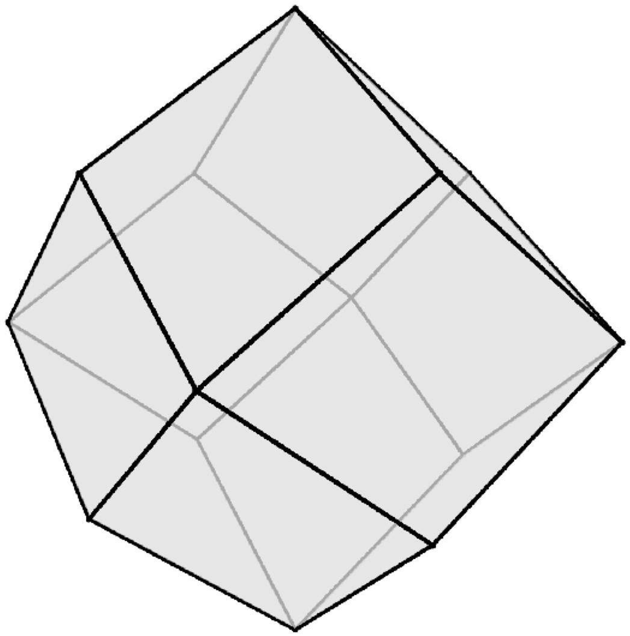
# Applications: True convex hull (1)

Application of barycentric gems ($n$-G-maps, cell-tuple): True exact convex hull, with non-simplicial facets.

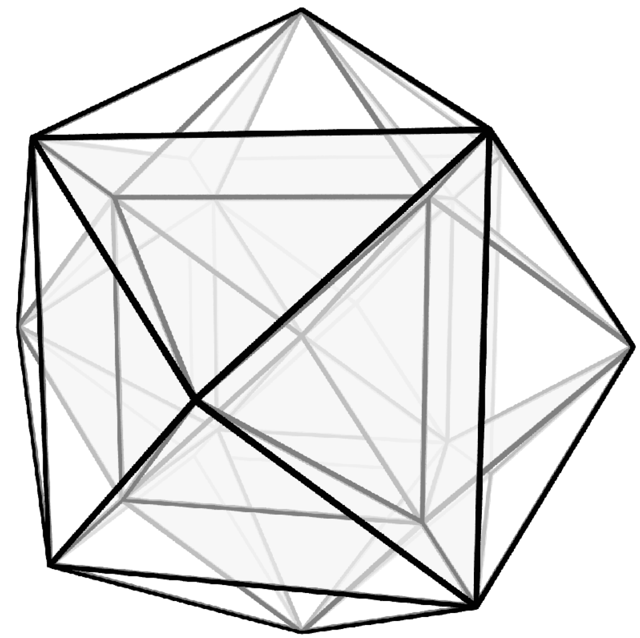
Gift-wrapping algorithm (D. R. Chand & S. S. Kapur 1970 [3]).



# Application: True convex hull (2)



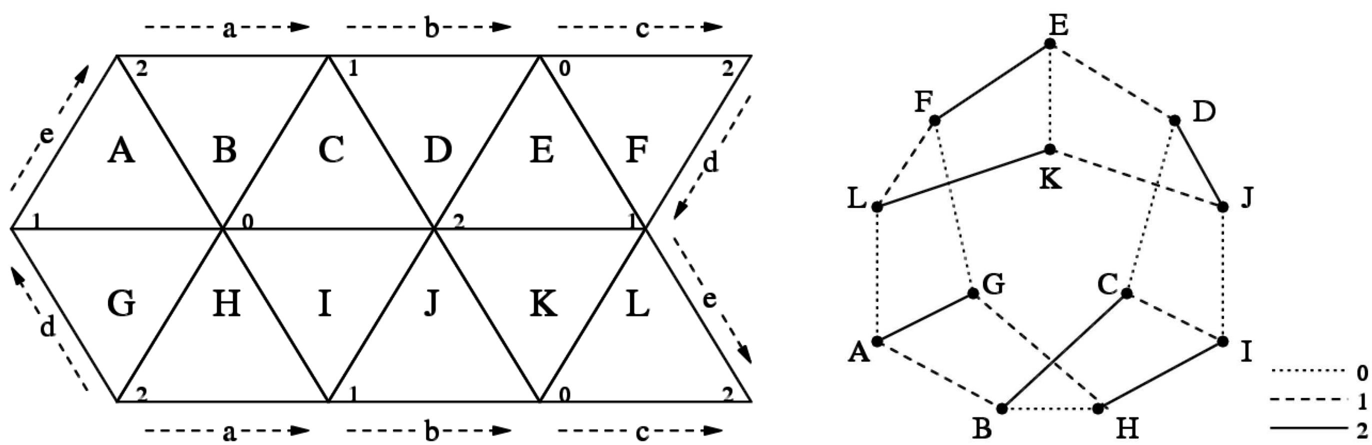
Rhombic dodecahedron (3D)



Regular 24-cell (4D)

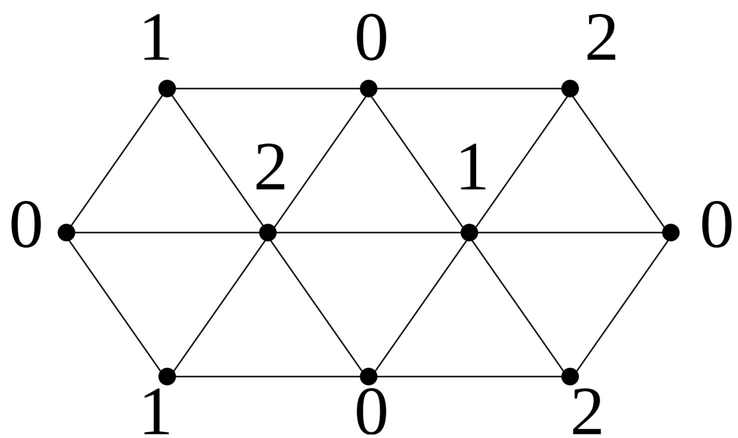
# Non-barycentric gems (1)

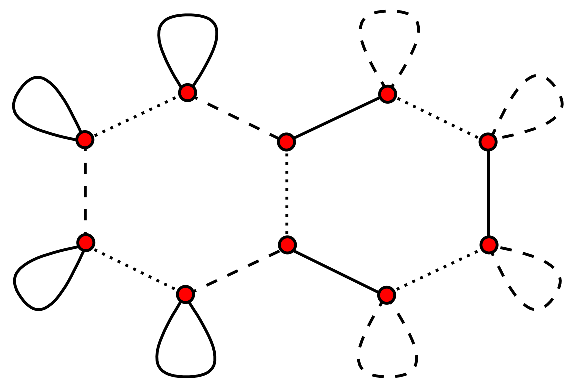
Gems need not be barycentric subdivisions:



# Non-barycentric gems (2)

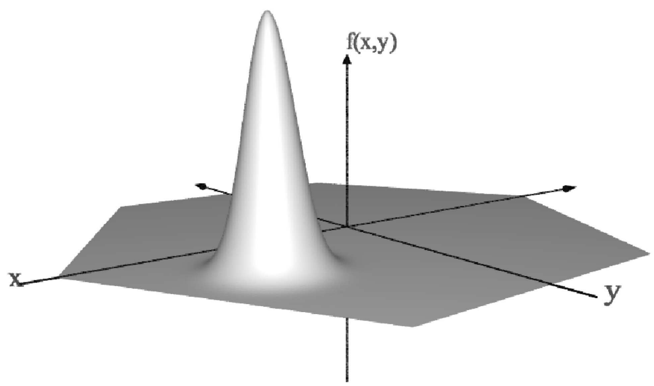
The free border of a gem need not be of color $d$:

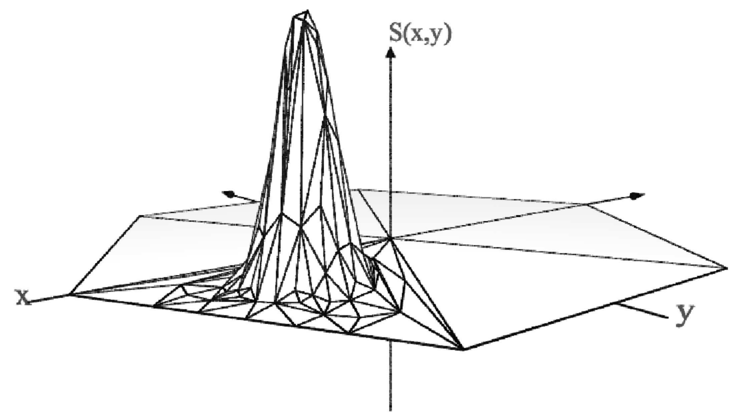


# Applications: Adaptive subdivision (1)

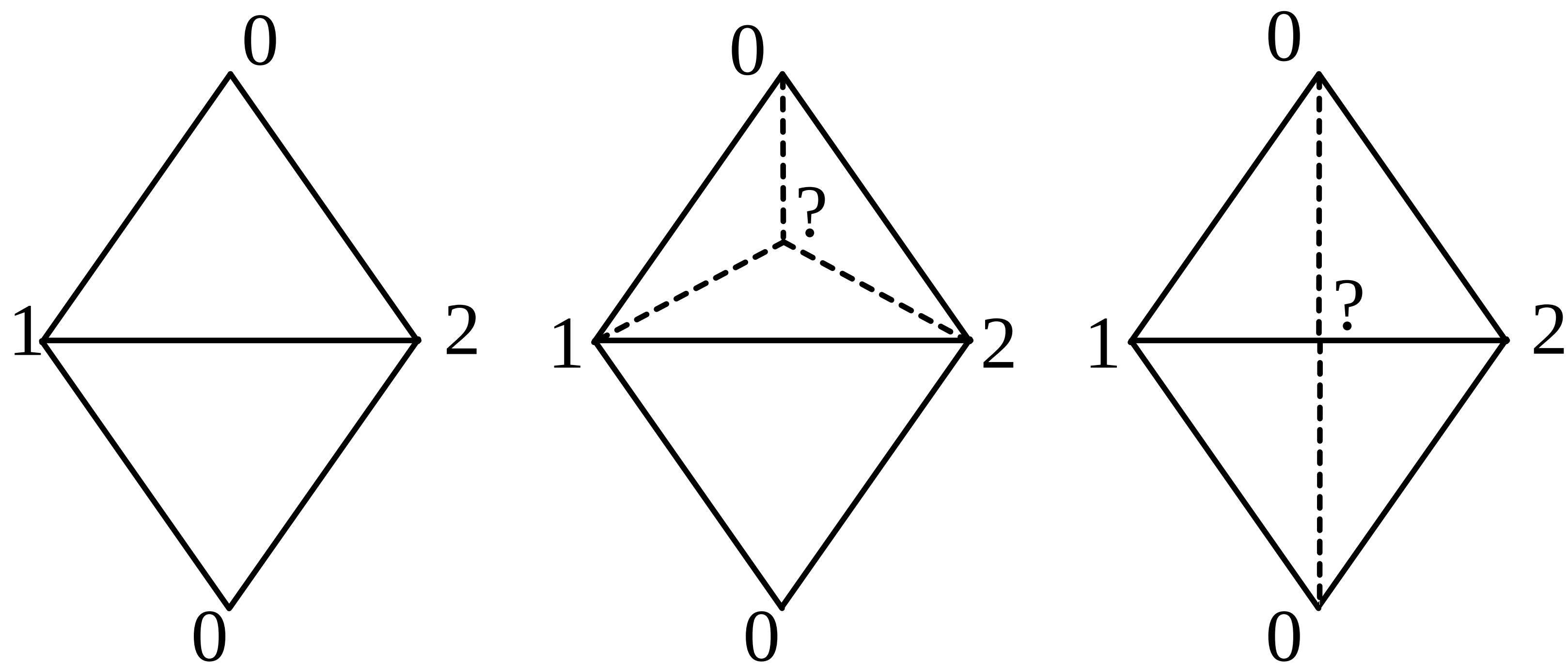
Application of non-barycentric gems:
Approximation by adaptive triangular mesh.



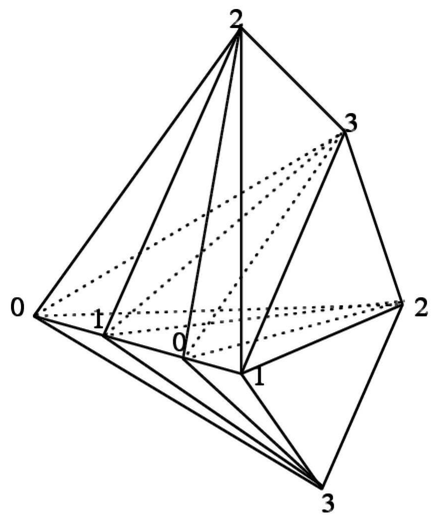
## Applications: Adaptive subdivision (2)

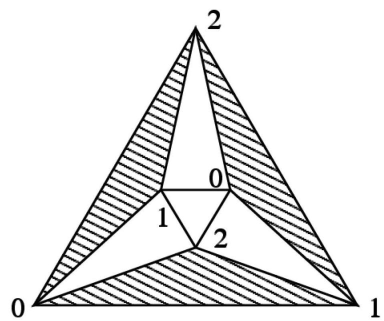
Most popular subdivision schemes don't work:

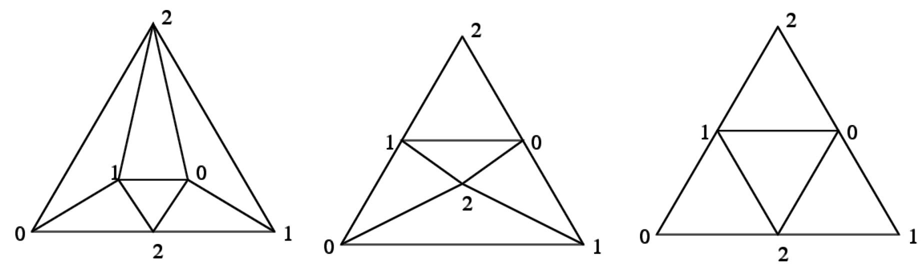


# Applications: Adaptive subdivision (3)

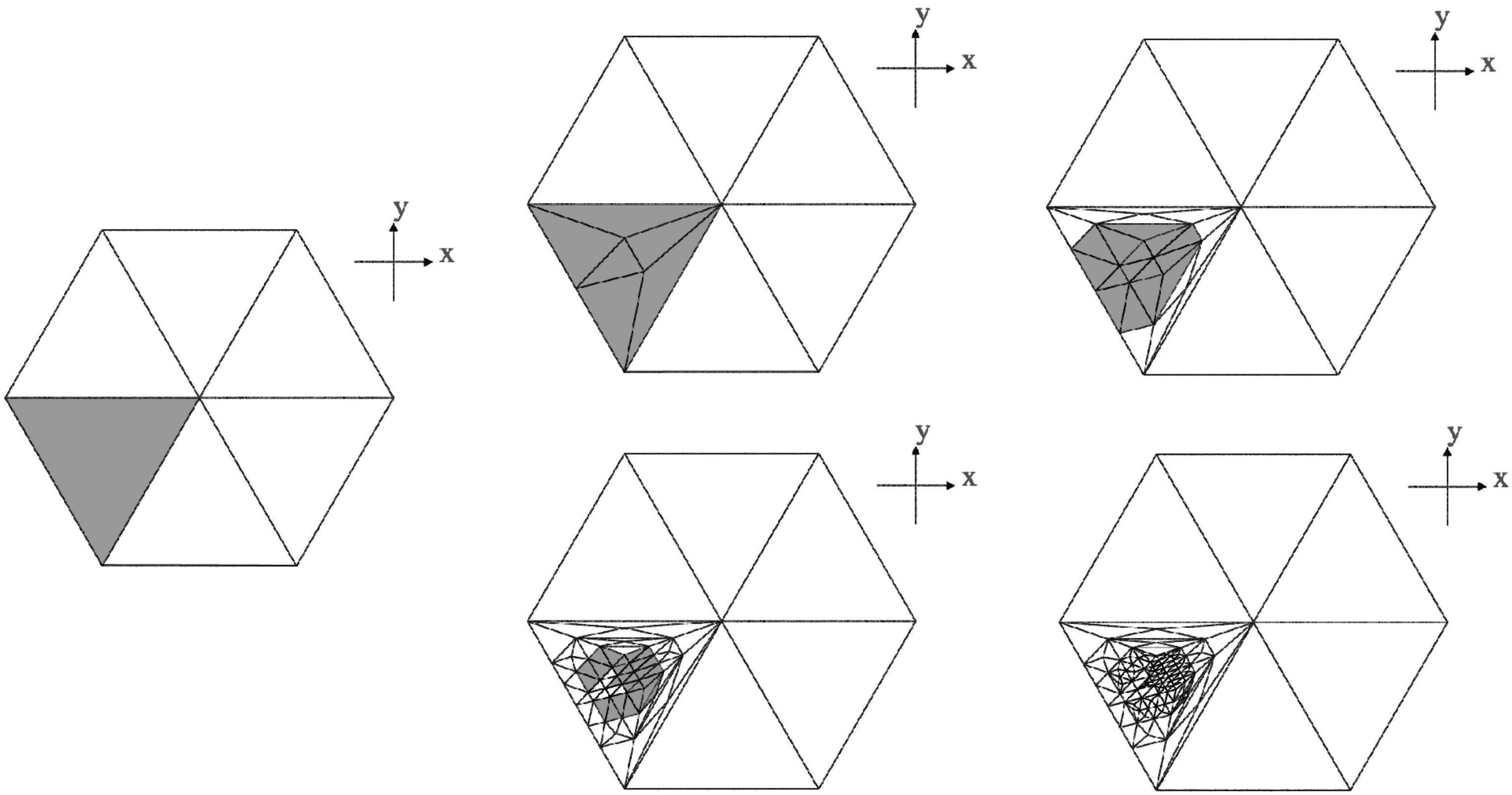
Local colored refinement schemes do exist:



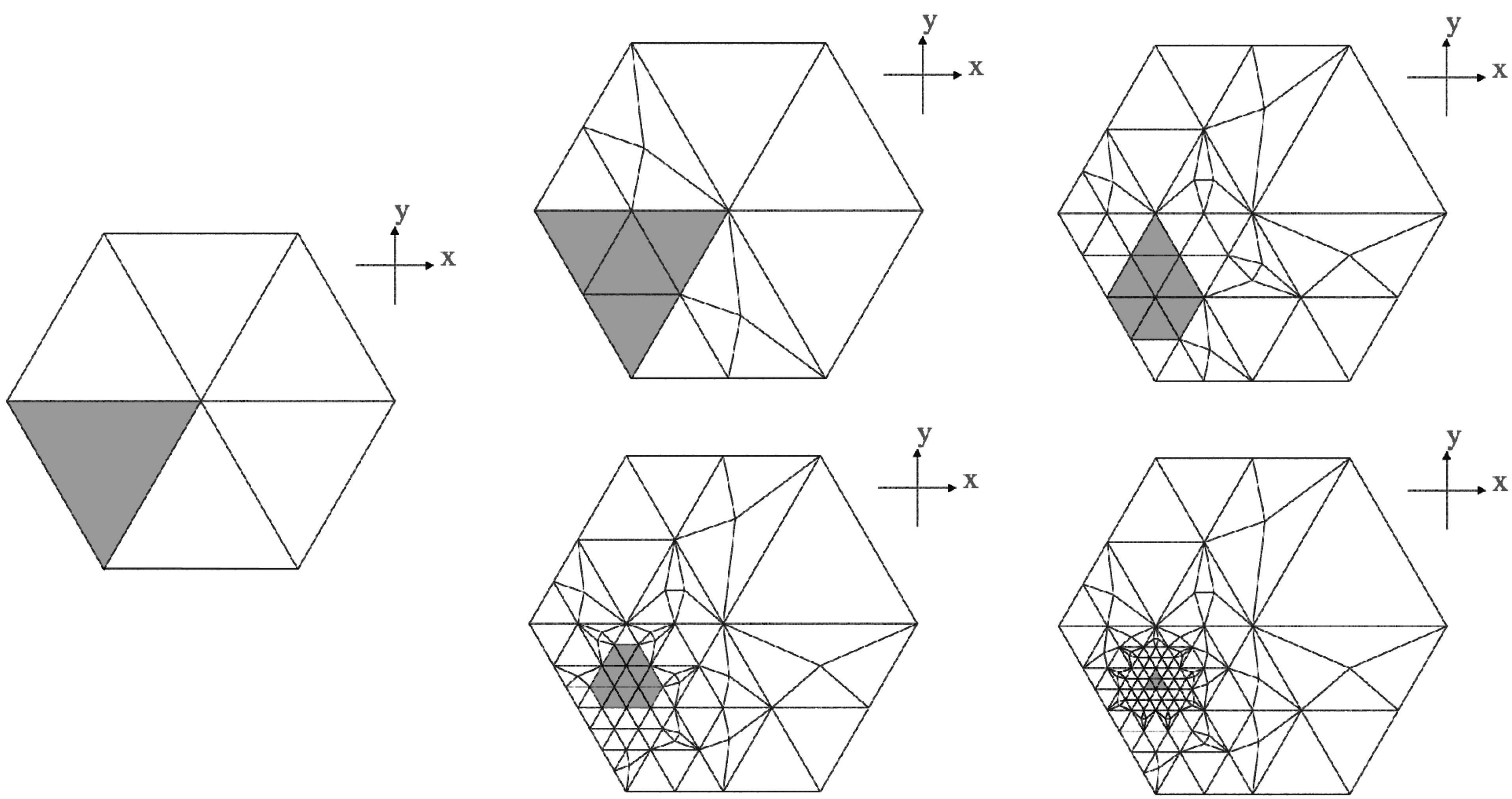



# Applications: Adaptive subdivision (4)

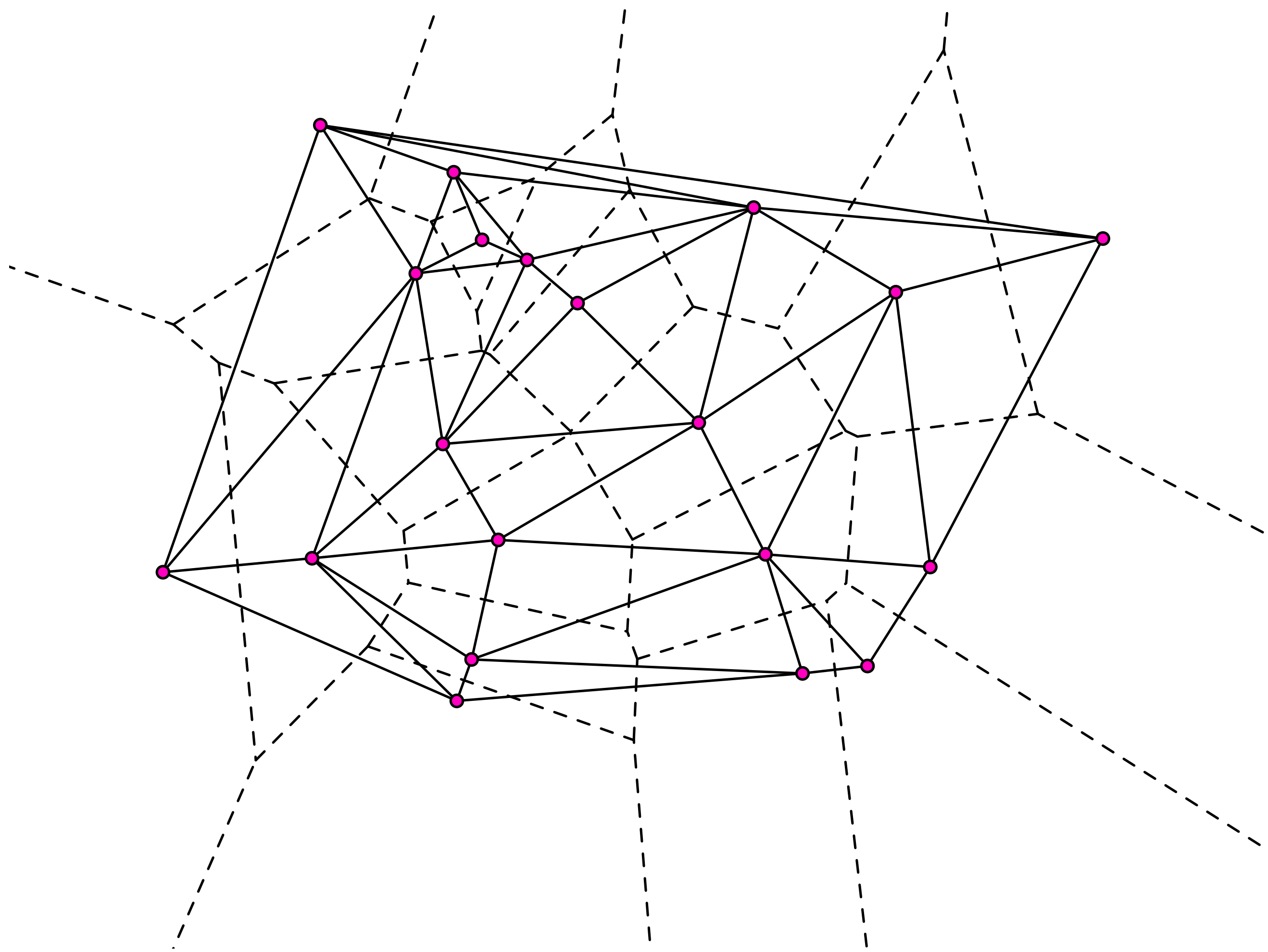


# Applications: Adaptive subdivision (5)

Can be done with minimum-angle guarantee:

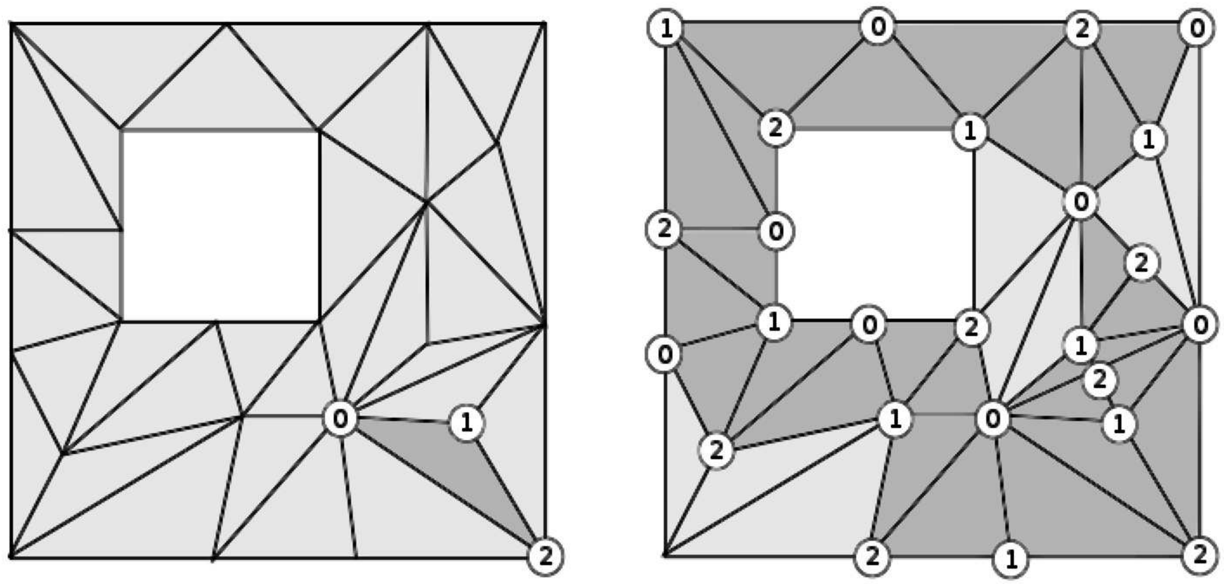


# Applications: Colorizing by splitting (1)

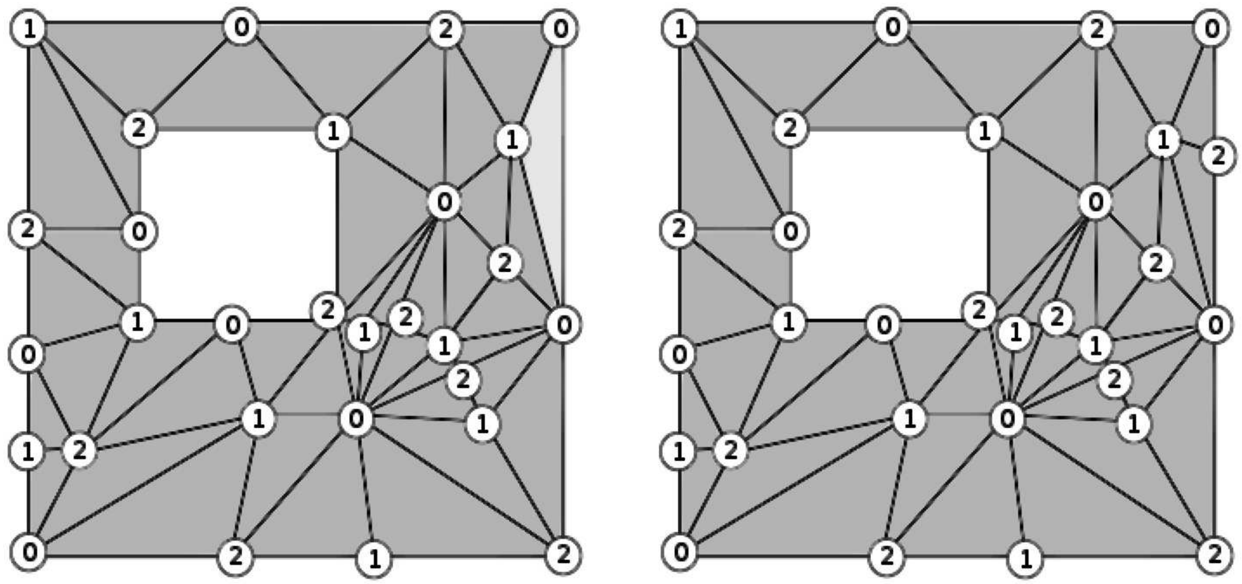


# Applications: Colorizing by splitting (2)

Turning an arbitrary triangulation into colored one.
Barycentric: easy but expensive, $n_{\mathrm{F}} \to 6n_{\mathrm{F}}$.
Moutinho's algorithm: $n_{\mathrm{F}} \to \leq 2n_{\mathrm{F}}$:



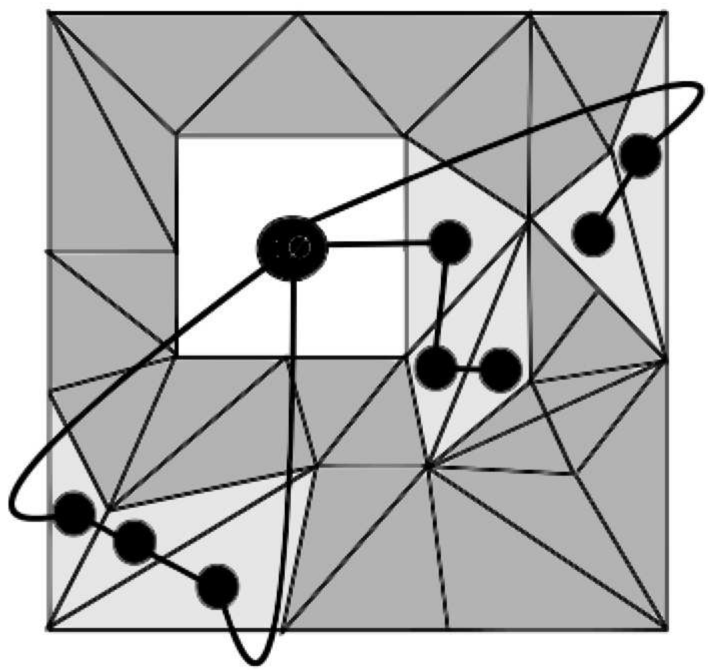
# Applications: Colorizing by splitting (2)



Sometimes splits a triangle in 2 to 6 pieces.
On average each triangle becomes at most 2, usually less.
If the triangulations is 3-colorable, does not split.

# Applications: Colorizing by splitting (3)

Border-sensitive shelling:



## Conclusions

Triangulations:
barycentric $\subset$ colored $\subset$ general.

Disadvantages of the gem data structure:

- Restricted triangulations (e.g. no Delaunay).
- Needs care in creation, or a splitting step.
- Restricted operations (gluing, subdivision).
- More wasteful than quad-edge or facet-edge for maps.

## Conclusions

Advantages of the gem data structure:

- Extends $n$-G-maps and cell-tuple:
  - Non-barycentric triangulations.
  - Arbitrary free borders.
- Very simple data structure and topological operators.
- Simplified connection to geometry.
- Generalized quad-edge/facet-edge structures.
- Residues are gems too.
- Poly-ality ($d!$ views) vs. duality (2 views).

# Further work

Future work and open problems:

- Efficient adaptive subdivision in $d \geq 3$ dimensions.
- Colorizing by frugal splitting in $d \geq 3$ dimensions.

## References


[1] J.-D. Boisonnat, O. Devillers, S. Pion, M. Teillaud, and M. Yvinec. Triangulations in CGAL. *Computational Geometry*, 22(1–3):5–19, 2002.

[2] Erik Brisson. Representing geometric structures in $d$ dimensions: Topology and order. *Proc. 5th Annual ACM Symp. on Computational Geometry*, pages 218–227, June 1989.

[3] Donald R. Chand and Sham S. Kapur. An algorithm for convex polytopes. *Journal of the ACM*, 17(1):78–86, 1970.

[4] David P. Dobkin and Michel J. Laszlo. Primitives for the manipulations of three-dimensional subdivisions. In *Proc. 3rd ACM Symp. on Comp. Geometry*, pages 86–99. ACM Press, June 1987.

[5] M. Ferri. Una rappresentazione delle varietà topologiche triangolabili medianti grafi $(n+1)$-colorati. *Bolletino dell'Unione Matematica Italiana*, 13-B:250–260, 1976.

[6] Leonidas J. Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Transactions on Graphics*, 4(2):74–123, April 1985.

[7] D.-T. Lee and B. J. Schachter. Two algorithms for constructing a delaunay triangulation. *Int. J. Computer Information Science*, 9:219–242, 1980.

[8] Pascal Lienhardt. Subdivisions of n-dimensional spaces and n-dimensional generalized maps. *Proc. 5th Annual ACM Symp. on Computational Geometry*, pages 228–236, June 1989.

[9] Sóstenes Lins. Graph-encoded maps. *Journal of Combinatory Theory (B)*, 32:171–181, 1982.

[10] Jonathan R. Shewchuck. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In *Proceedings of the 1st Workshop on Applied Computational Geometry*, pages 124–133, May 1996.