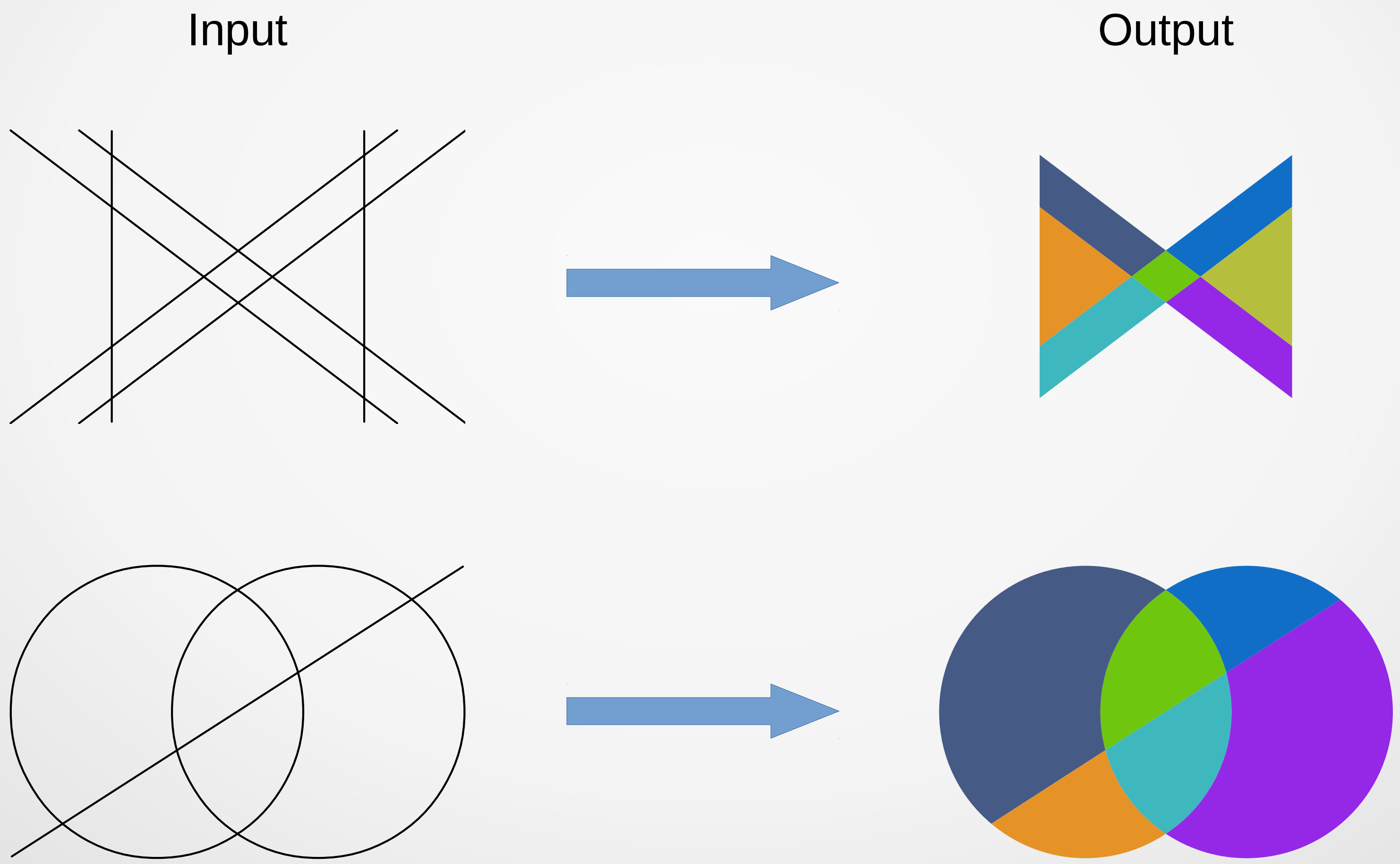
A Planar Subdivision Method, Beyond Straight Lines

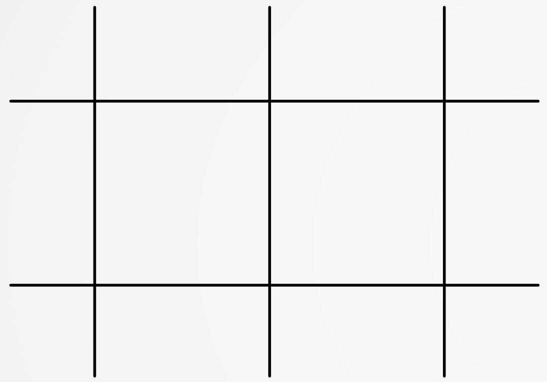
IS-LAB seminar
Saeed, September 2016
[with special thanks to Adam Duracz]

Objective

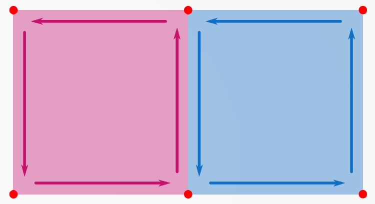


Problem Statement - Assumptions

Input: a set of curves

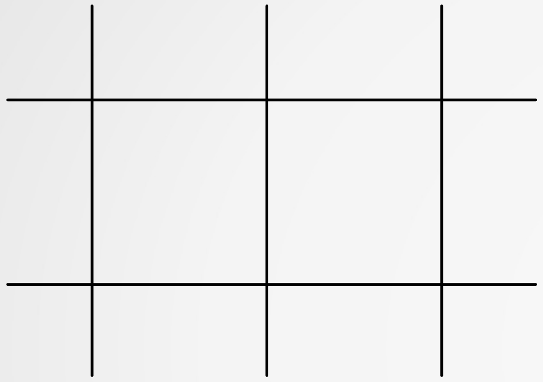


Output: partitioning of space

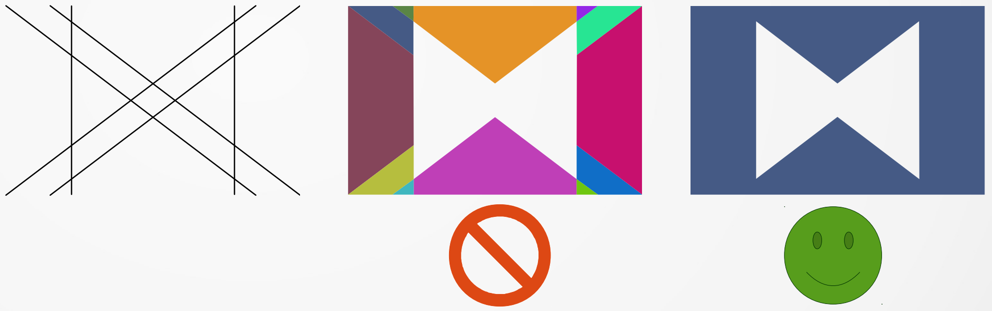


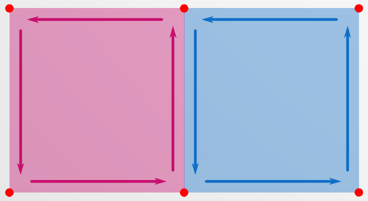
- **Space**: a 2D plane.
- **Classes of Curves**:
  - straight lines; as an example of an *unbounded* class,
  - circles; as an example of a *bounded* class.
- **No Redundancy**: if two curves were identical, their intersection would be the same curve which is beyond a finite set of points.

Problem Statement - Terminology



- **Curves:** A curve set contains the level-curves of some multivariable functions.
- **Faces:** A face is the “interior” region of a “Jodan Curve”, i.e. a closed and simple compilation of curves.
- **Unbounded Faces**





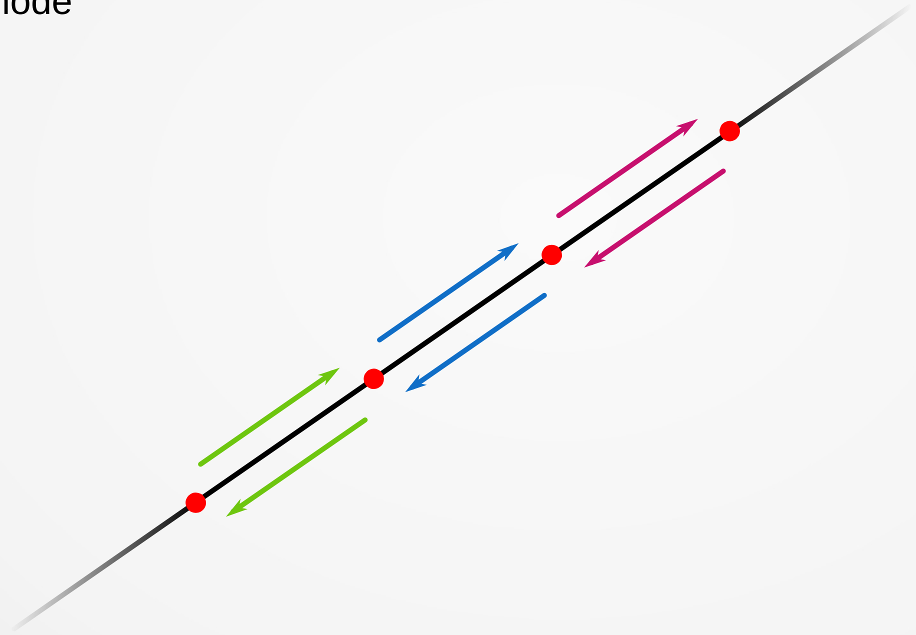
Functions

- **member:** a function of points which returns the face that encompass a given point.
- **neighbor:** is a function of faces and returns all other faces that are neighboring the input face by the mean of [at least] one shared edge in their boundaries.

State of The Art - DCEL

Doubly Connected Edge List

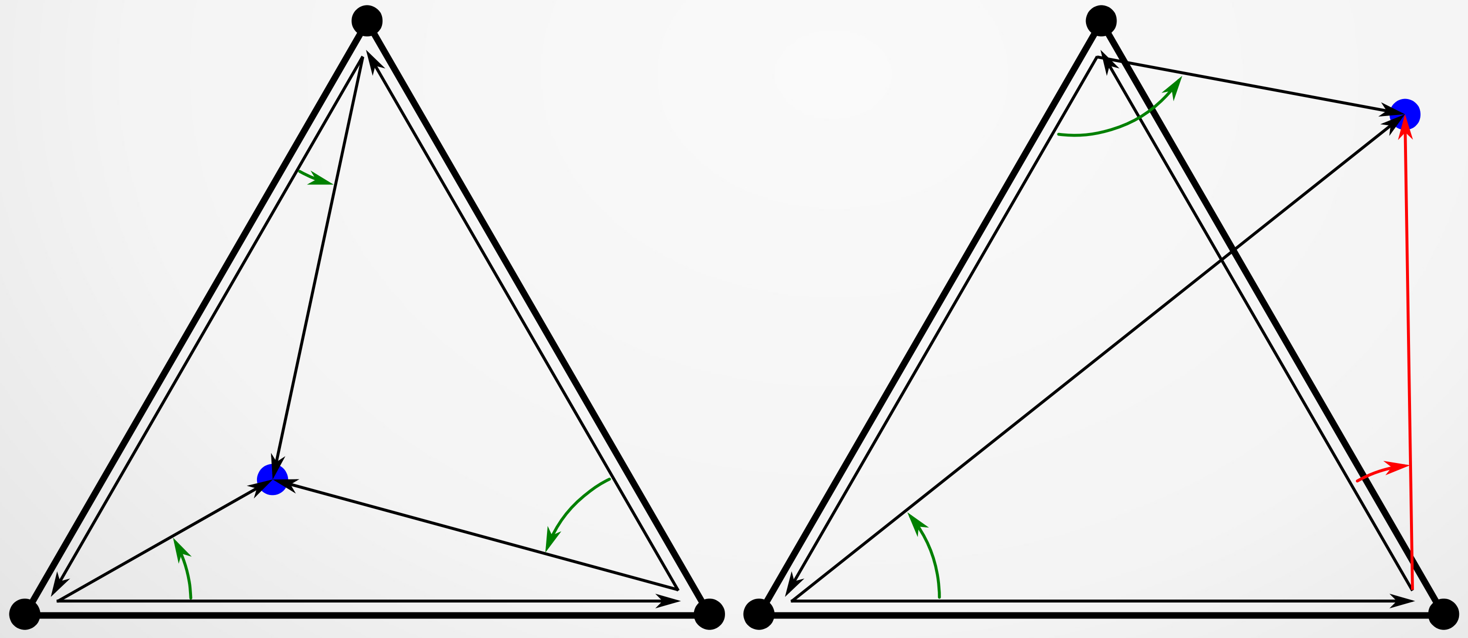
- **Half-Edge**
  - twin
  - directed
  - start-node, end-node
- Membership



State of The Art - DCEL

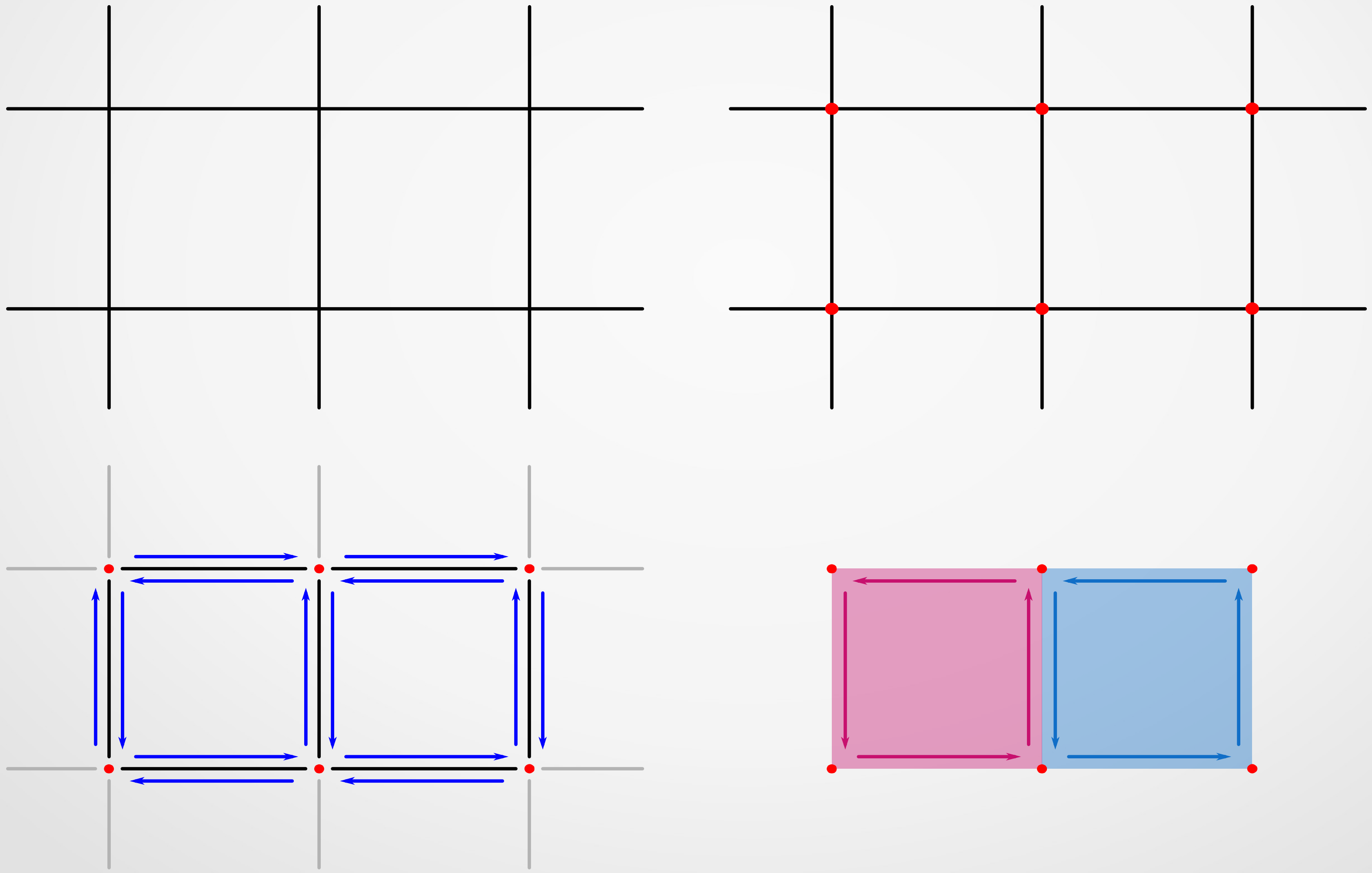
Doubly Connected Edge List

- Half-Edge
- **Membership**
  - cross-product
  - convexity assumed



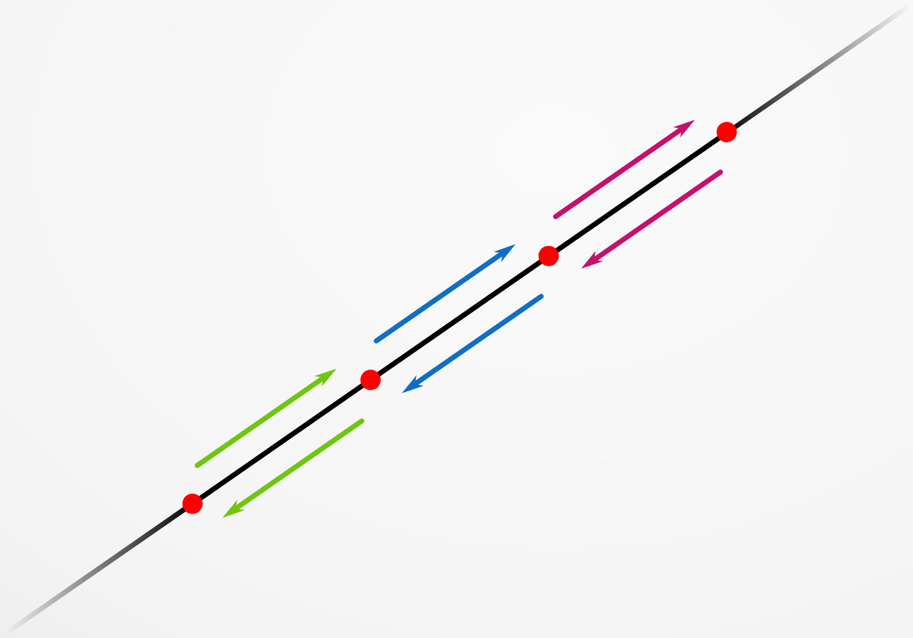
State of The Art - DCEL

Partitioning Procedure



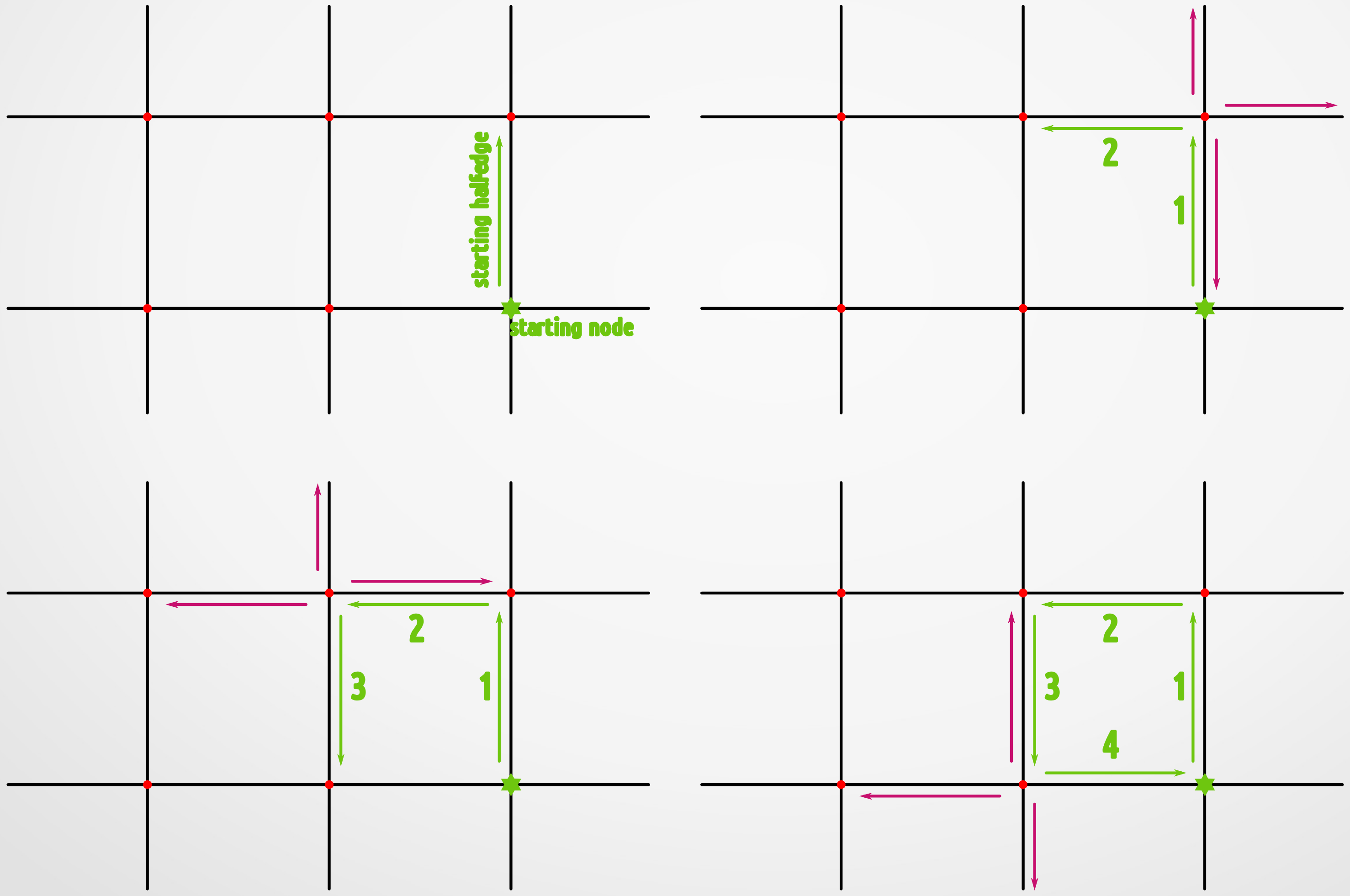
State of The Art - DCEL

Partitioning Procedure – intersection & half-edge identification

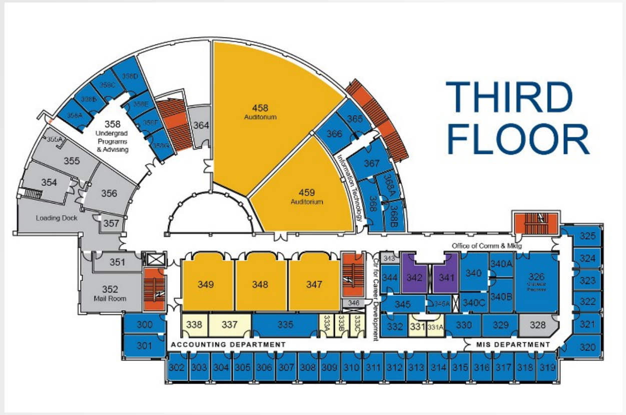


State of The Art - DCEL

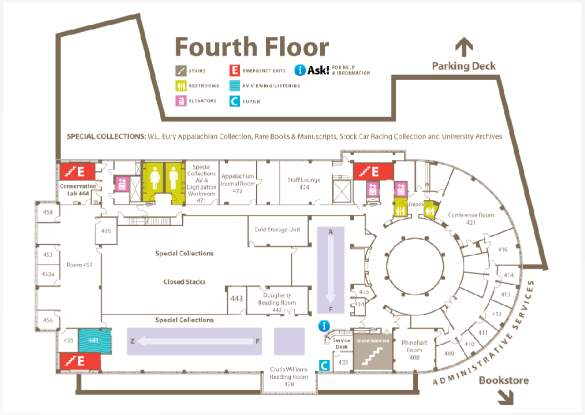
Partitioning Procedure - Face Identification via path following



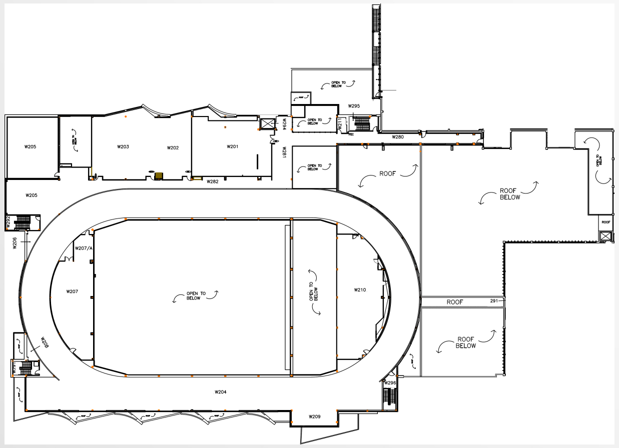
Why circles?

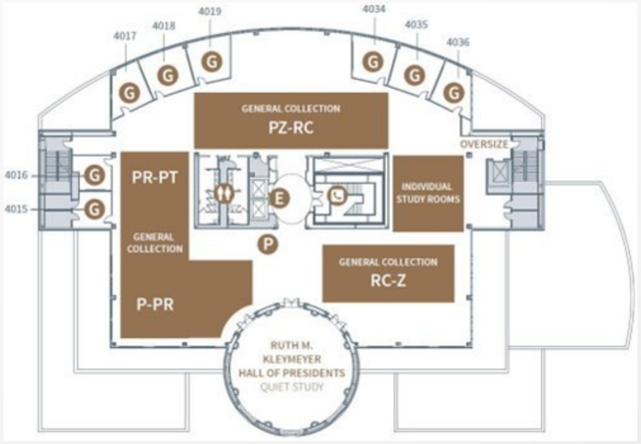


West Virginia University



Belk Library


McMaster University



Southern Indiana University

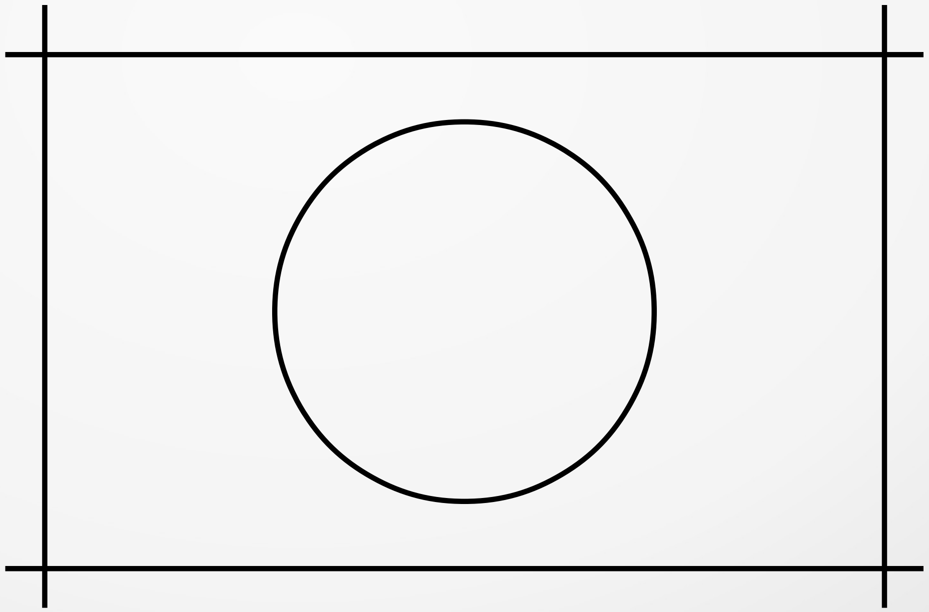
* all maps are from the web, links under titles.

So, what's the deal with circles?

Circles - Challenges

Challenges:

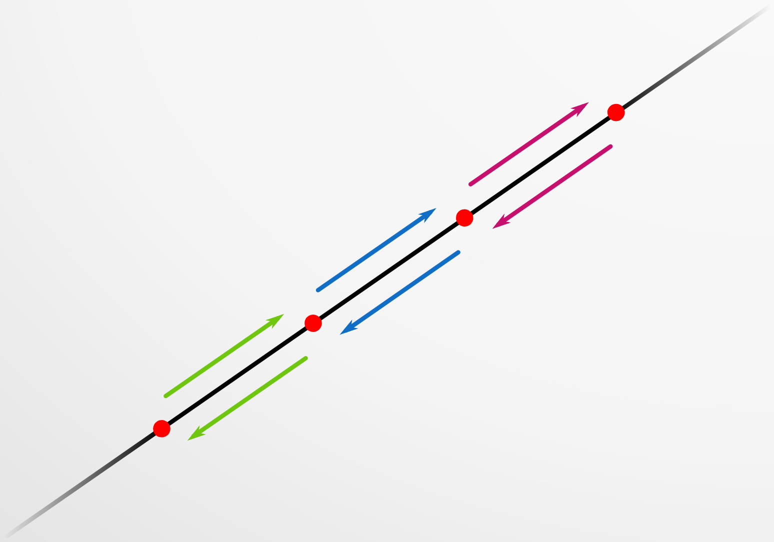
- **None intersecting circles**
- Sorting nodes over curves
- Edges are no longer vectors
- Holes
- Membership function

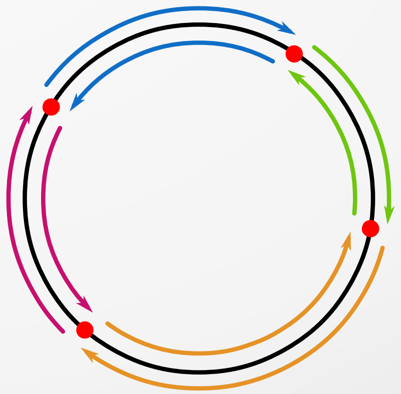


Circles - Challenges

Challenges:

- None intersecting circles
- **Sorting nodes over curves**
- Edges are no longer vectors
- Holes
- Membership function

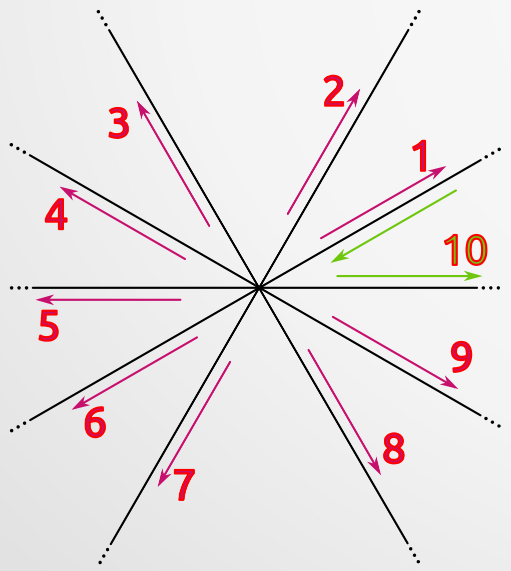




Circles - Challenges

Challenges:

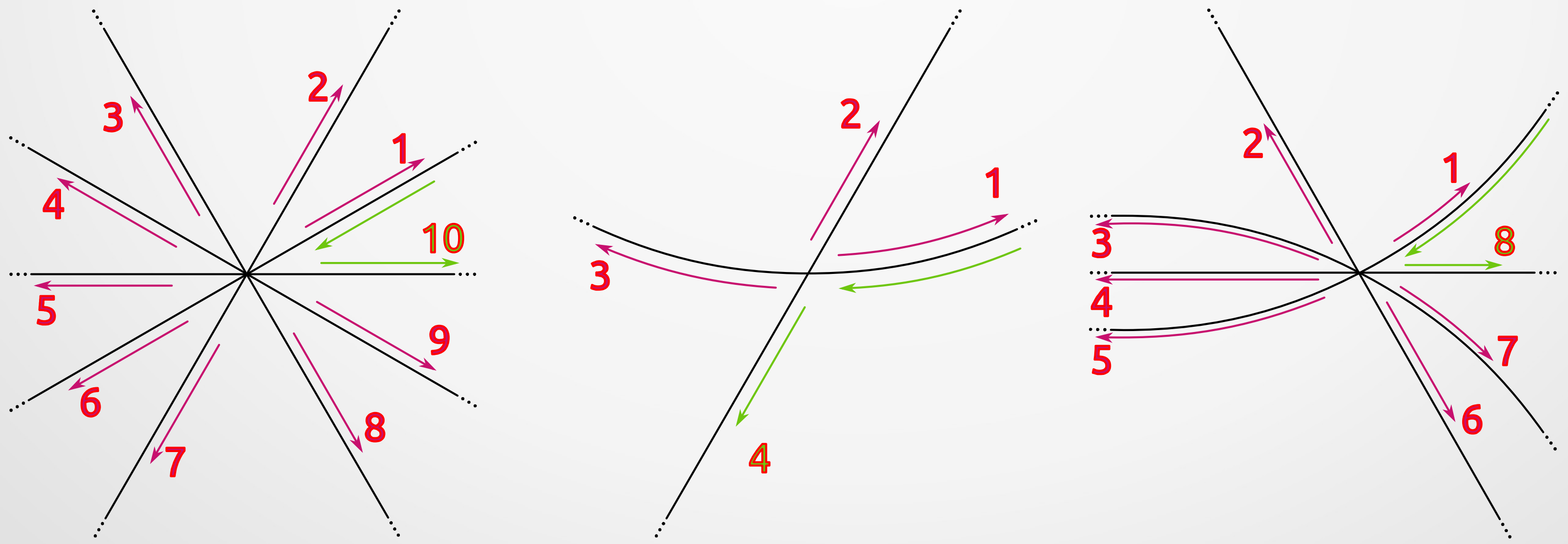
- None intersecting circles
- Sorting nodes over curves
- **Edges are no longer vectors**
  - ✓ finding the correct successor via departure angle of the edge
- Holes
- Membership function



Circles - Challenges

Challenges:

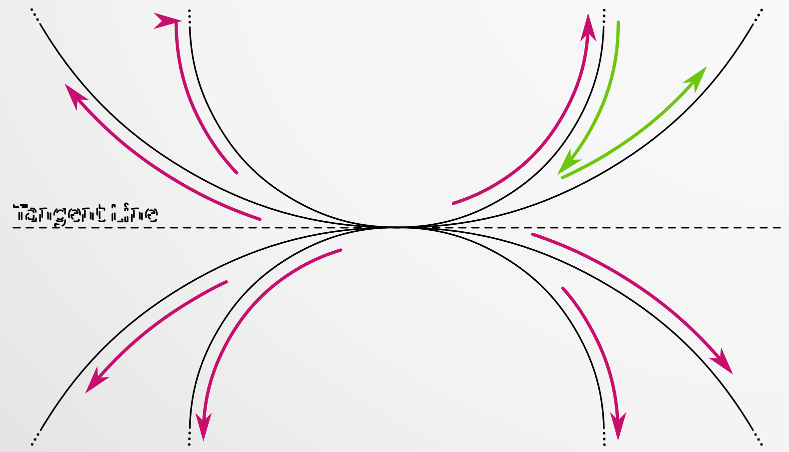
- None intersecting circles
- Sorting nodes over curves
- **Edges are no longer vectors**
  - ✓ finding the correct successor via departure angle of the edge – first derivative
- Holes
- Membership function



Circles - Challenges

Challenges:

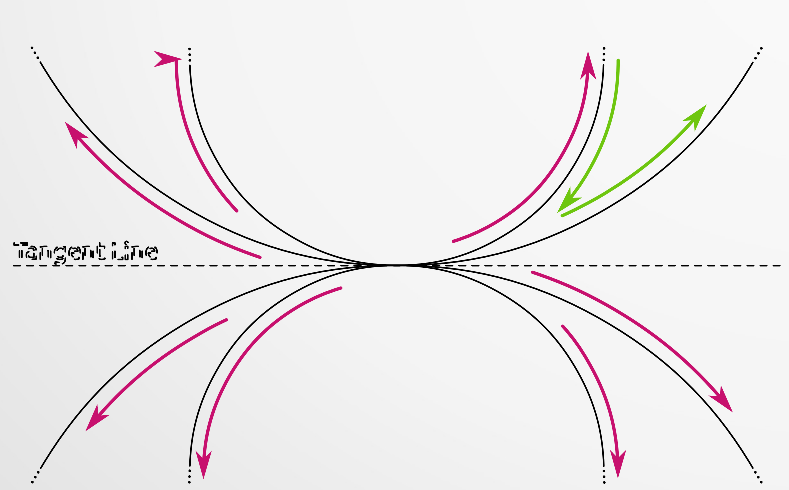
- None intersecting circles
- Sorting nodes over curves
- **Edges are no longer vectors**
  - ✓ Tangency
- Holes
- Membership function

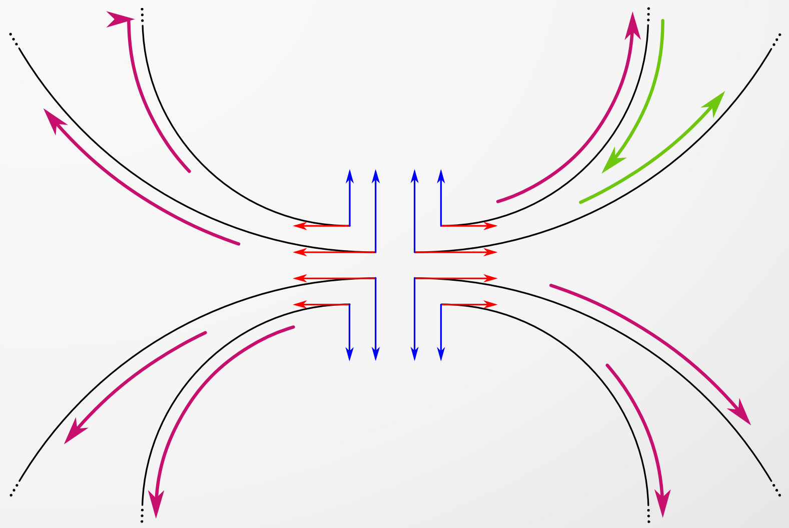


Circles - Challenges

Challenges:

- None intersecting circles
- Sorting nodes over curves
- **Edges are no longer vectors**
  - ✓ Tangency – second derivative
- Holes
- Membership function

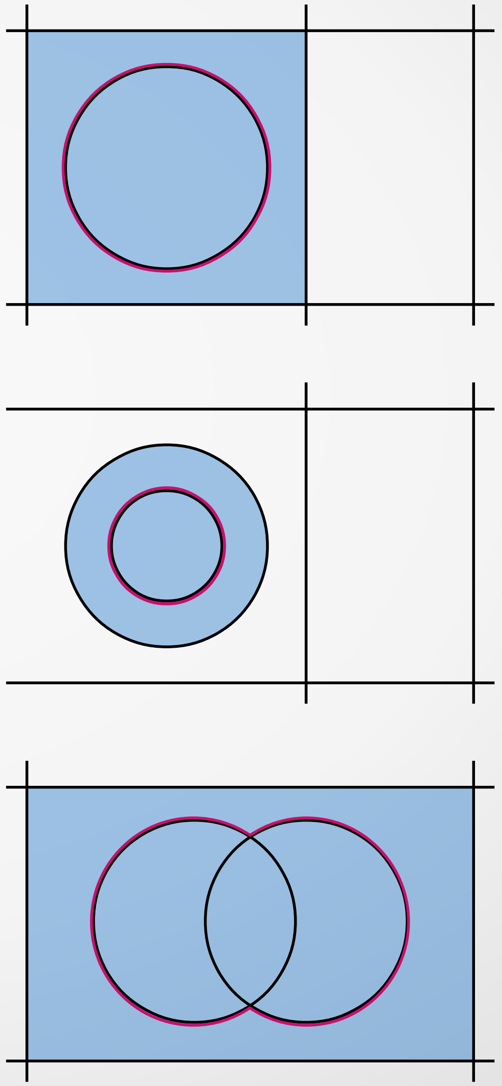




Circles - Challenges

Challenges:

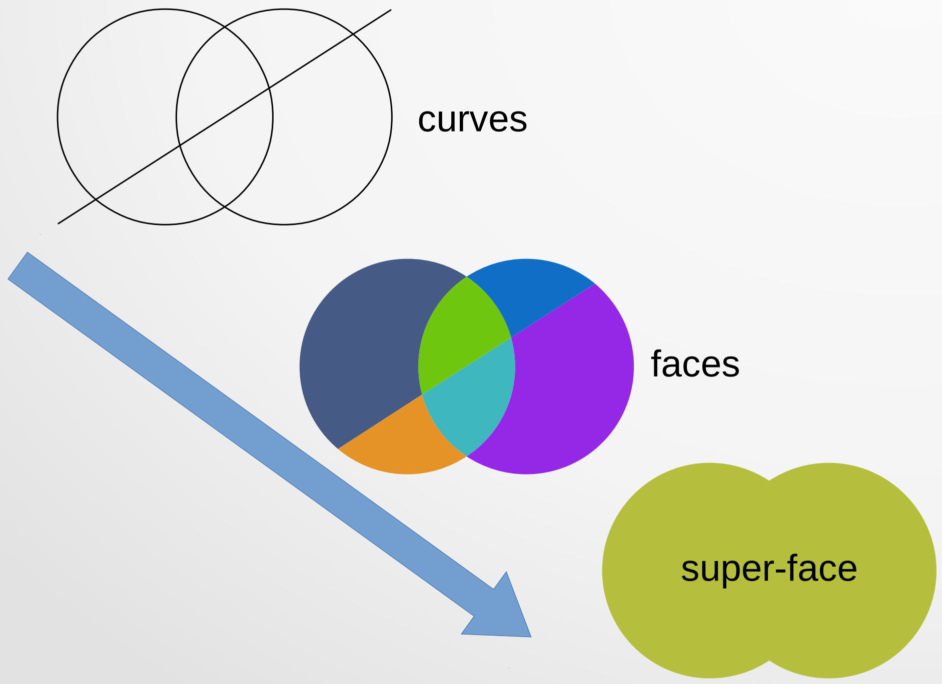
- None intersecting circles
- Sorting nodes over curves
- Edges are no longer vectors
- **Holes**
- Membership function

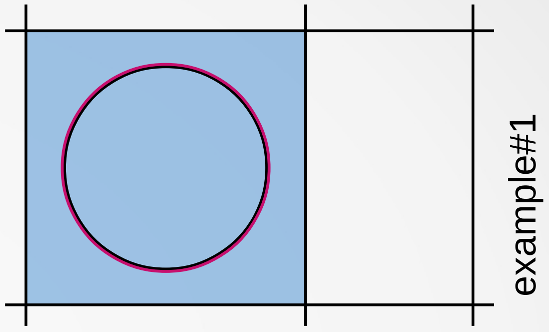


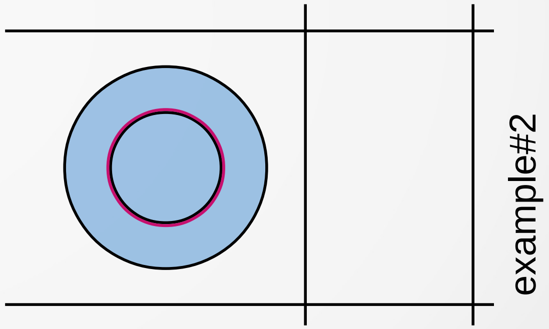
Circles - Challenges

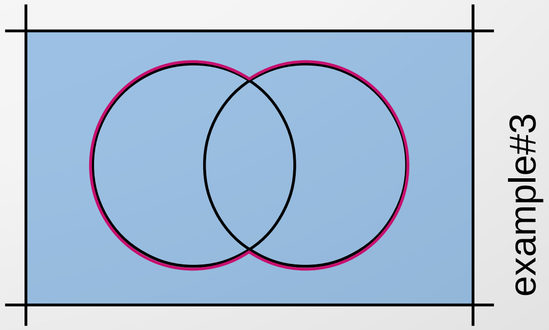
Challenges:

- None intersecting circles
- Sorting nodes over curves
- Edges are no longer vectors
- **Holes**
  - ✓ Subgraphs and super-face
- Membership function





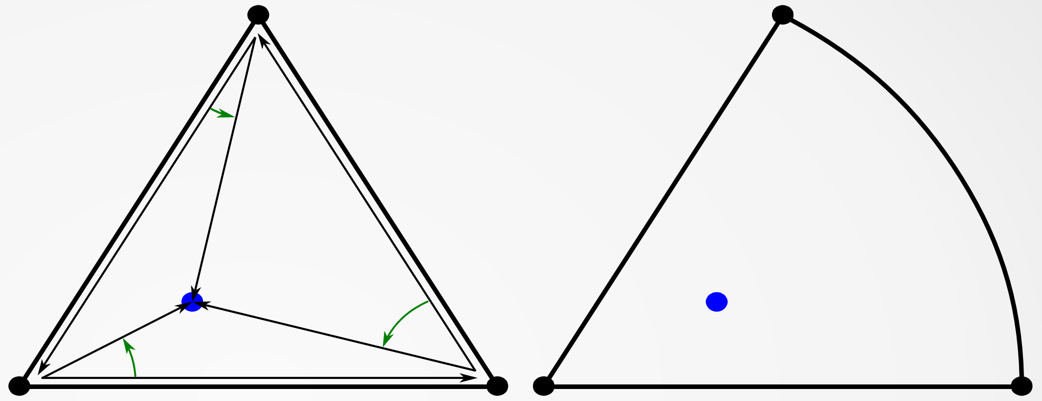




Circles - Challenges

Challenges:

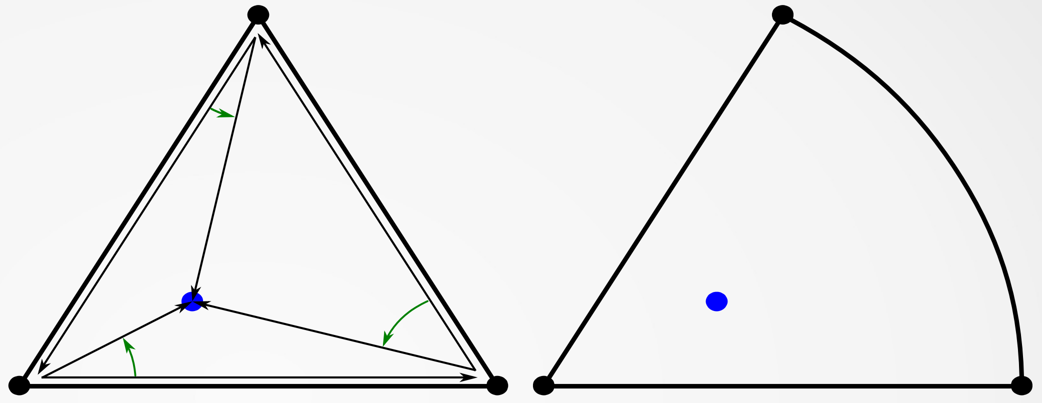
- None intersecting circles
- Sorting nodes over curves
- Edges are no longer vectors
- Holes
- ✓ **Membership function**



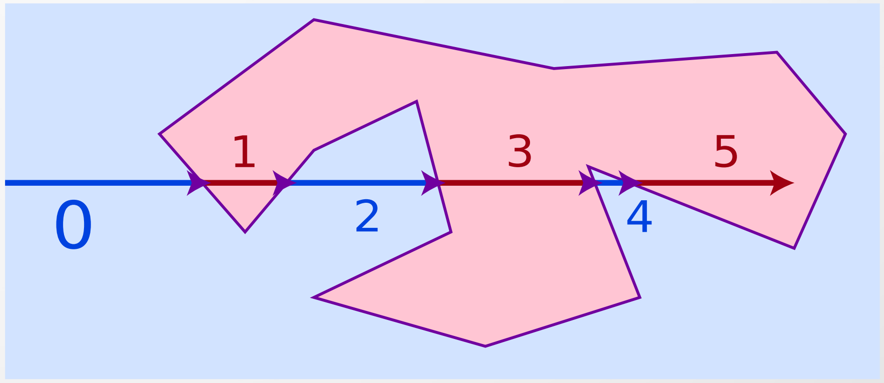
Circles - Challenges

Challenges:

- None intersecting circles
- Sorting nodes over curves
- Edges are no longer vectors
- Holes
- **Membership function**
  - ✓ Point-In-Polygone, ...



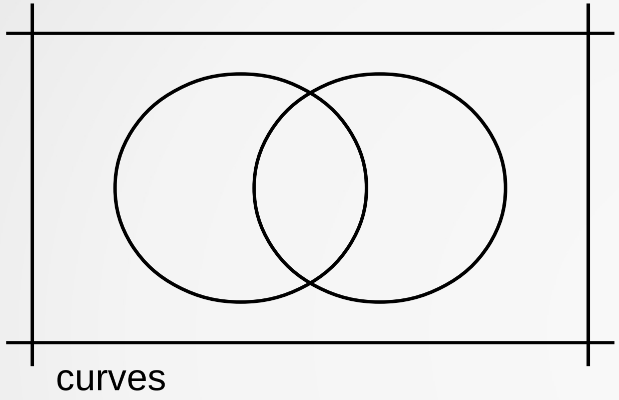
Point-In-Polygone: “The number of intersections for a ray passing from the exterior of the polygon to any point; if odd, it shows that the point lies inside the polygon. If it is even, the point lies outside the polygon; this test also works in three dimensions”*.

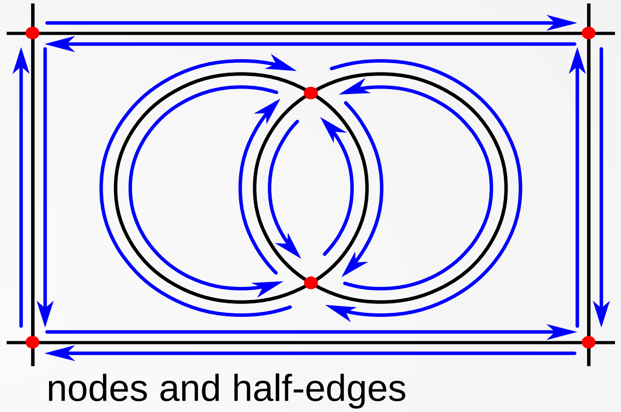


* “Point-In-Polygone”, the description and image from wikipedia.

Subdivision

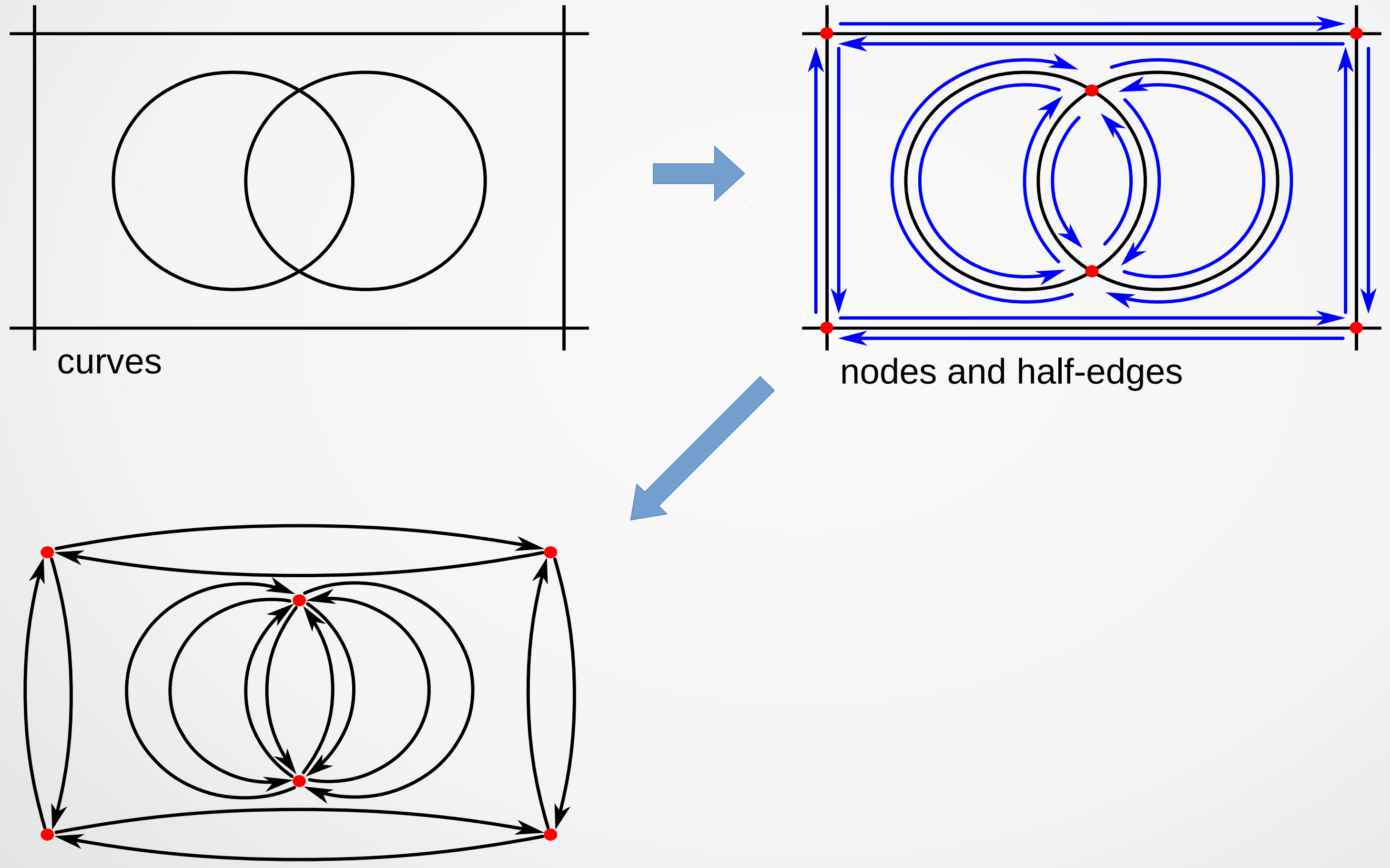
Intersection: finding nodes and half-edges





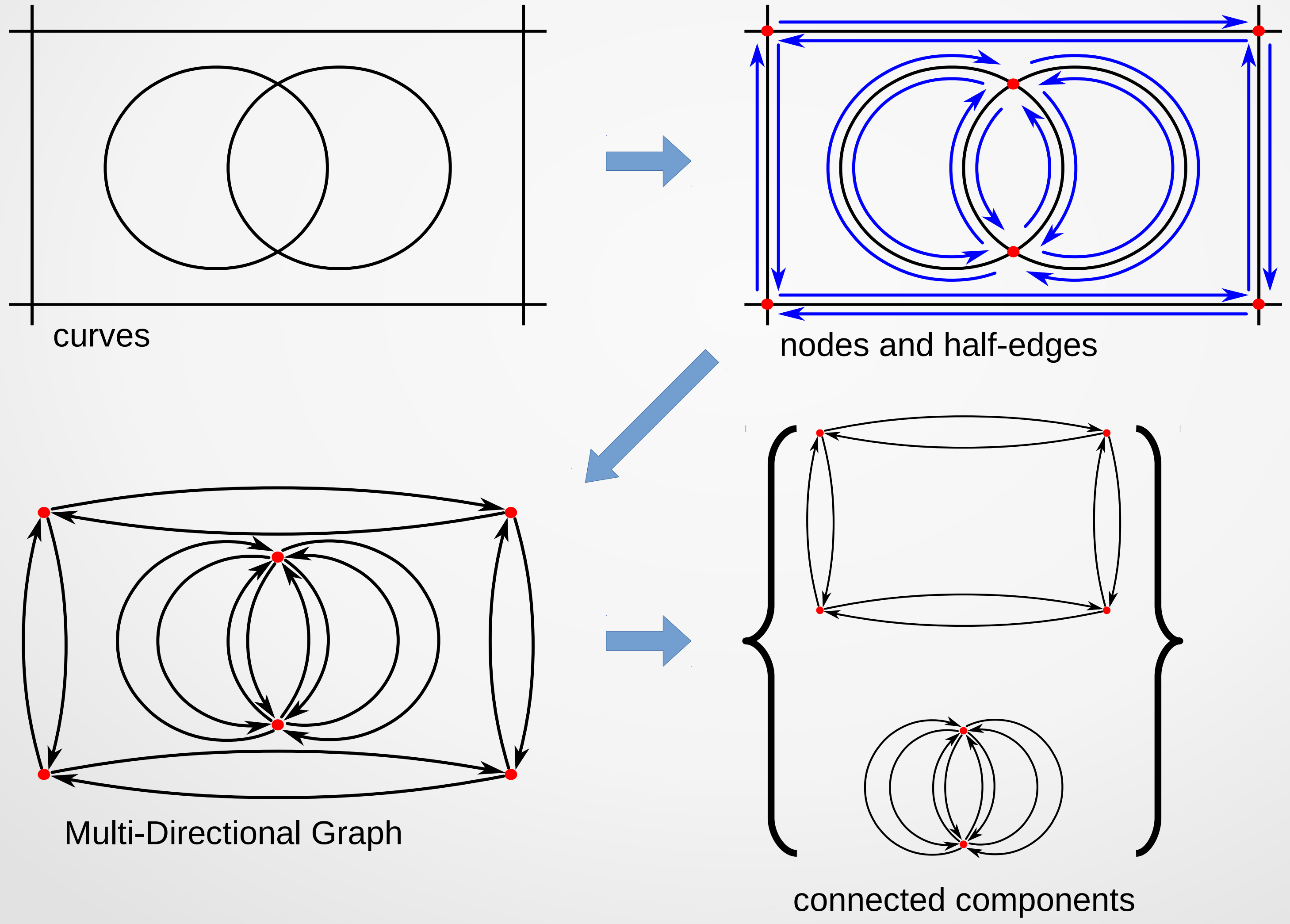
Subdivision

Constructing a Multi-Directional Graph (MDG)



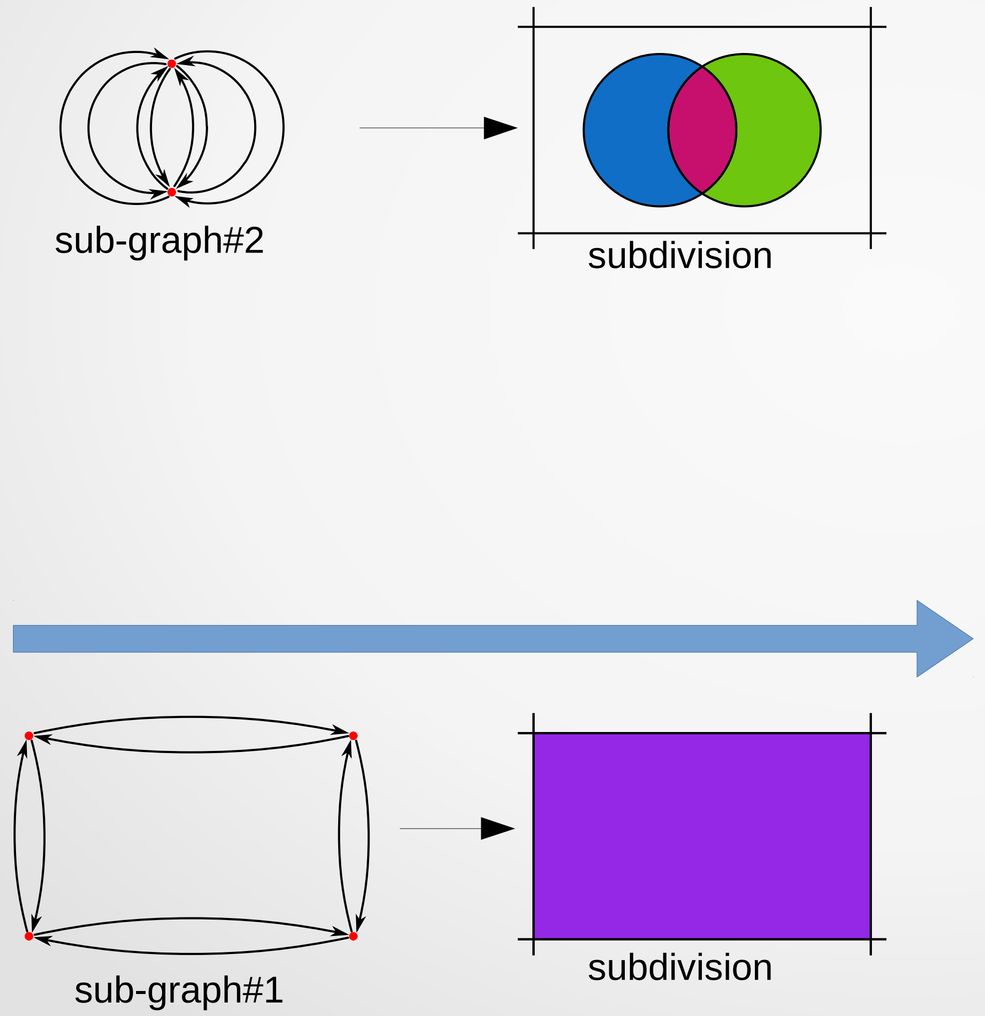
Subdivision

Connected components



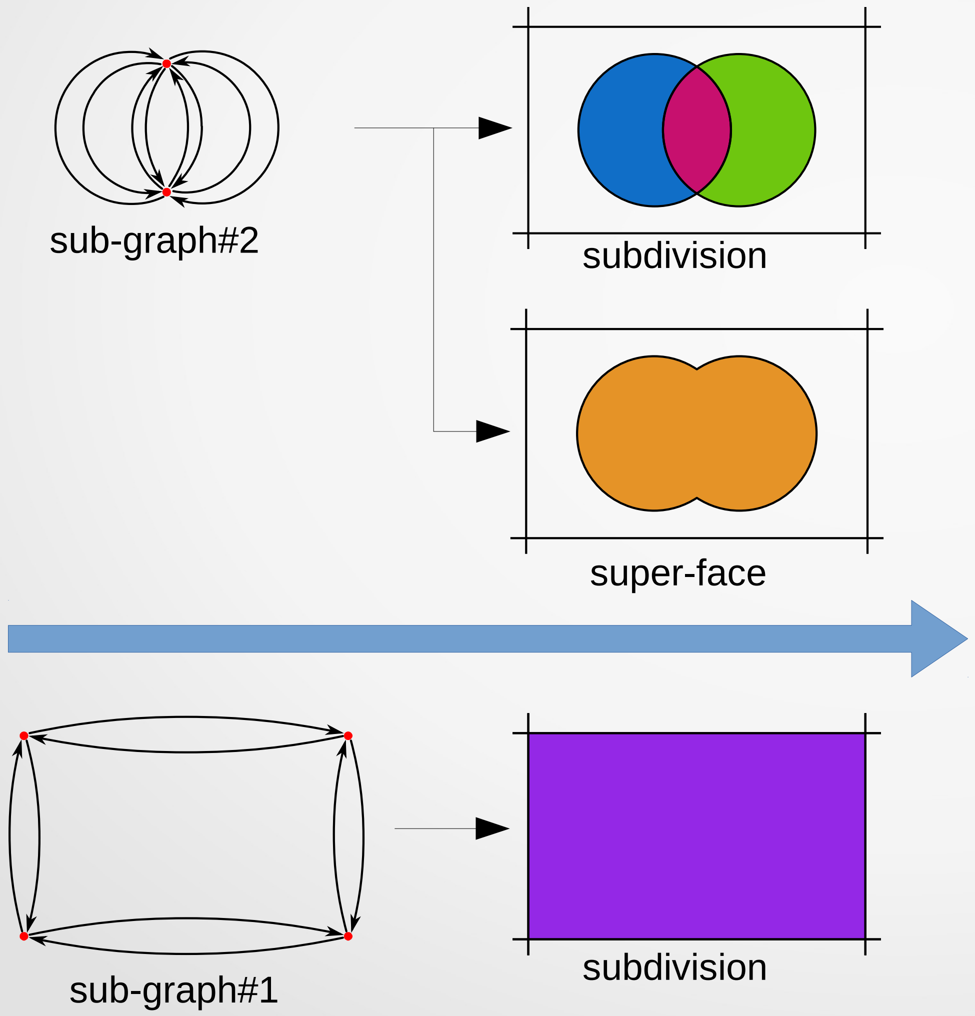
Subdivision

Decomposing sub-graphs (i.e. connected components)



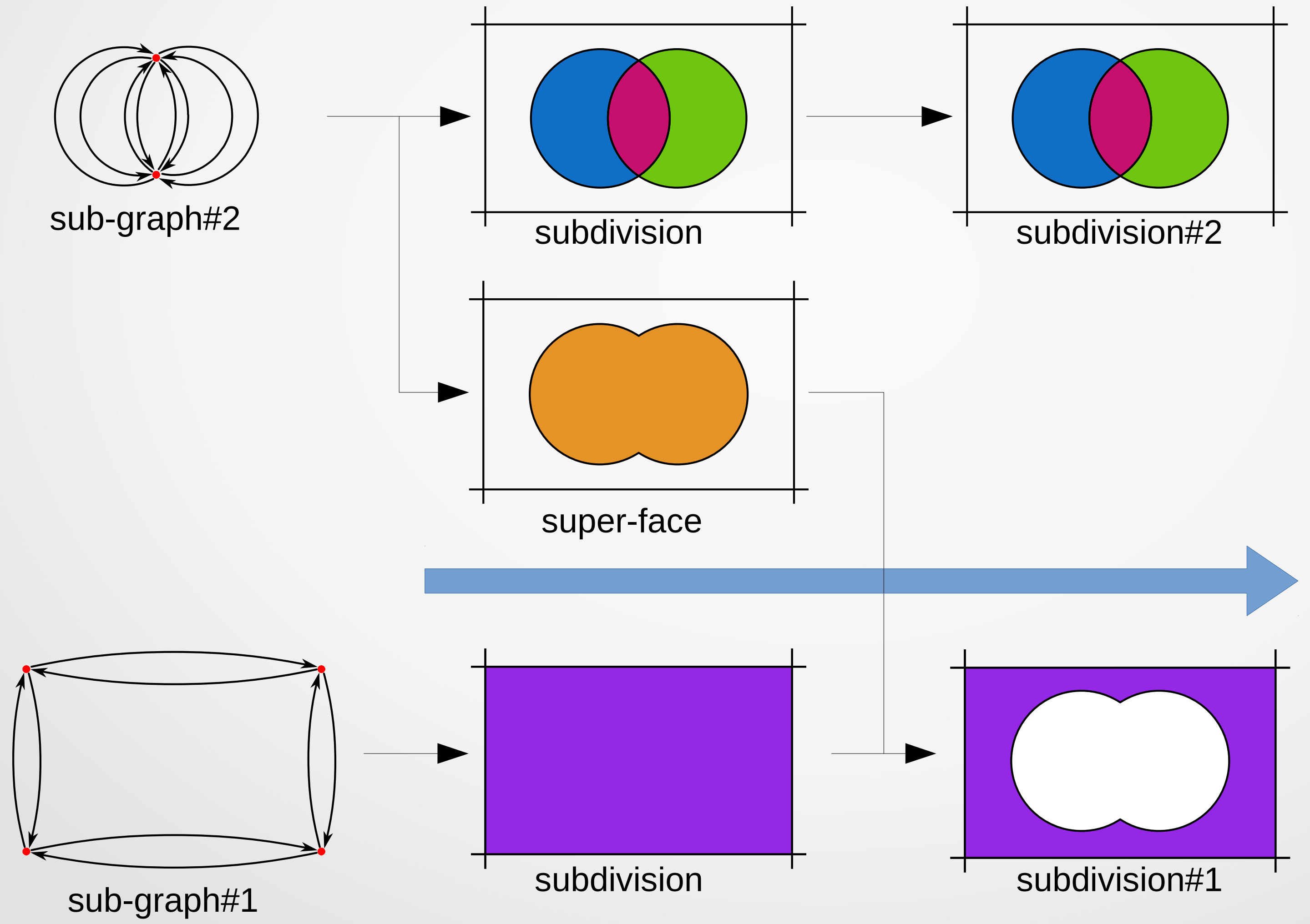
Subdivision

Detecting overlay (and super-face)



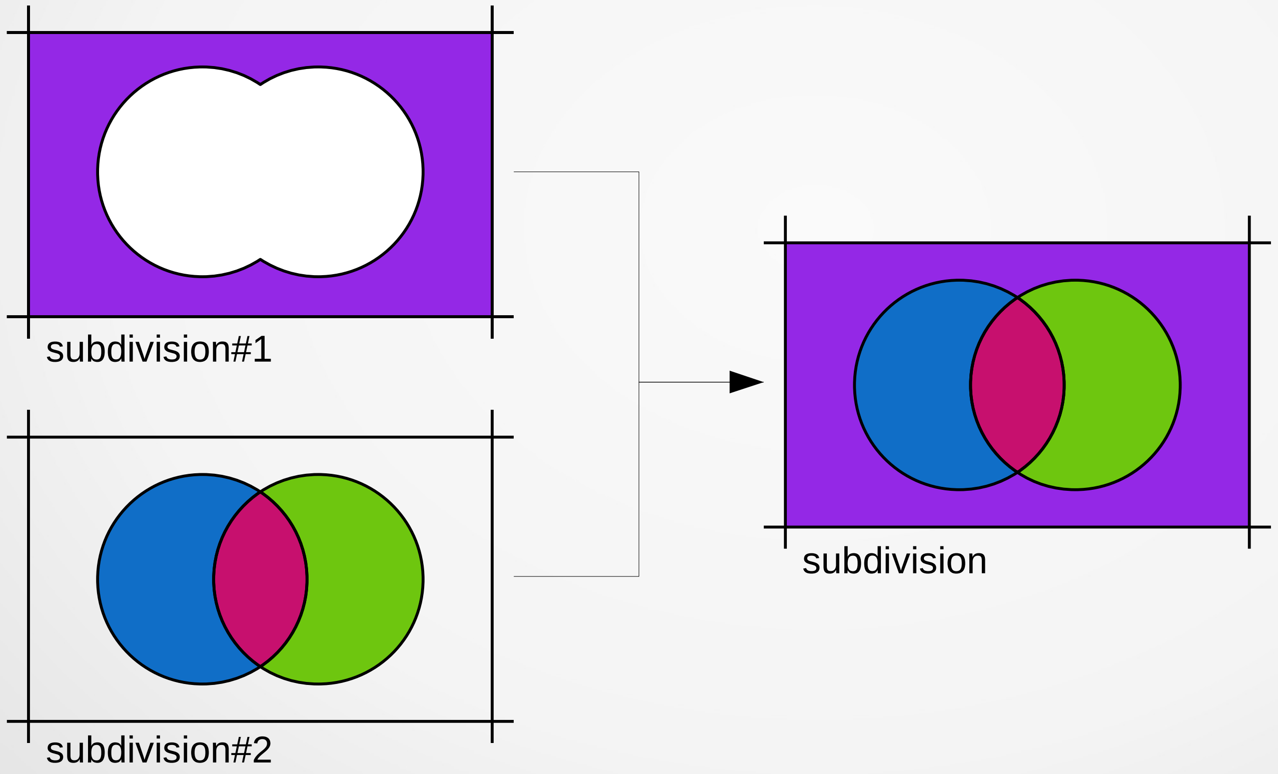
Subdivision

Updating faces according to overlay and super-faces.

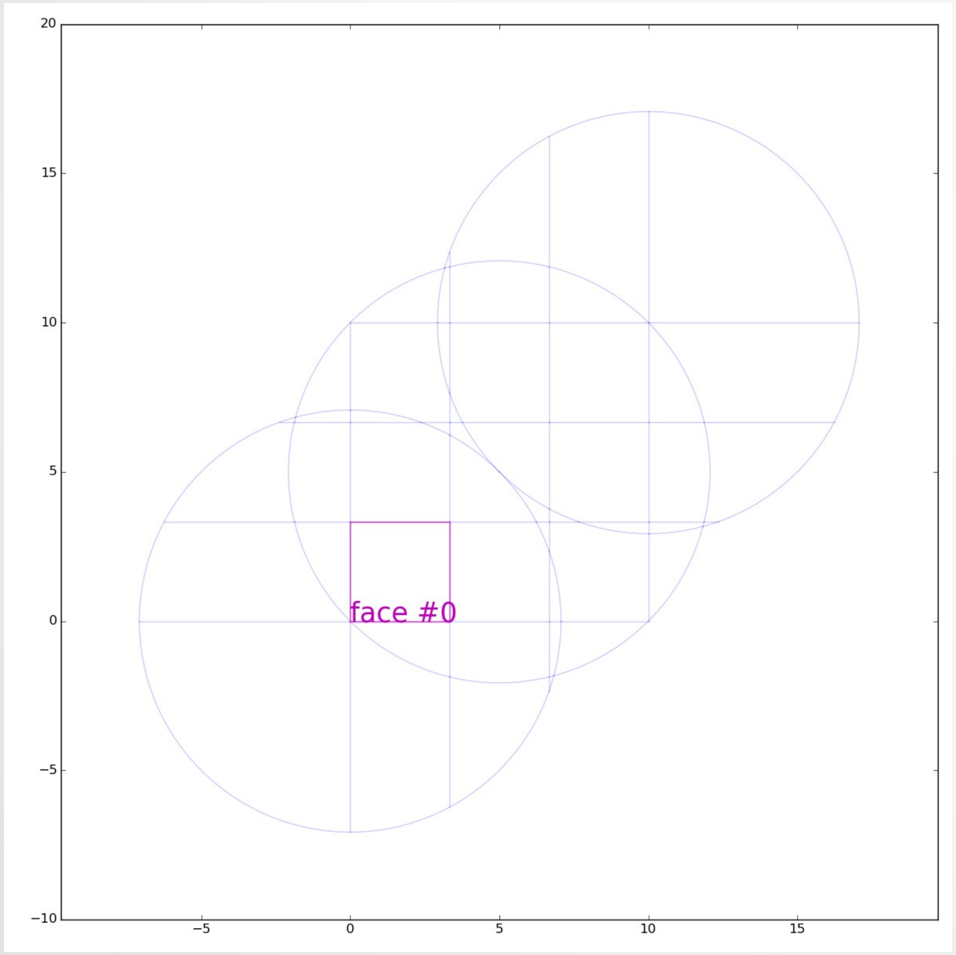


Subdivision

Merging sub-graphs



Implementation



Private repo available at:
https://github.com/saeedghsh/subdivision/

Dependencies:

- Python >=2.6
- numpy >= 1.10.2
- sympy >= 1.0
- networkx >= 1.10
- matplotlib >= 1.4.3

Summary

Contributions:

- Extending the subdivision algorithm beyond straight-line
- A prototype of the implementation

TODO:

- implementation efficiency – speed
- degenerate cases
- subdivision overlay (i.e. intersection)

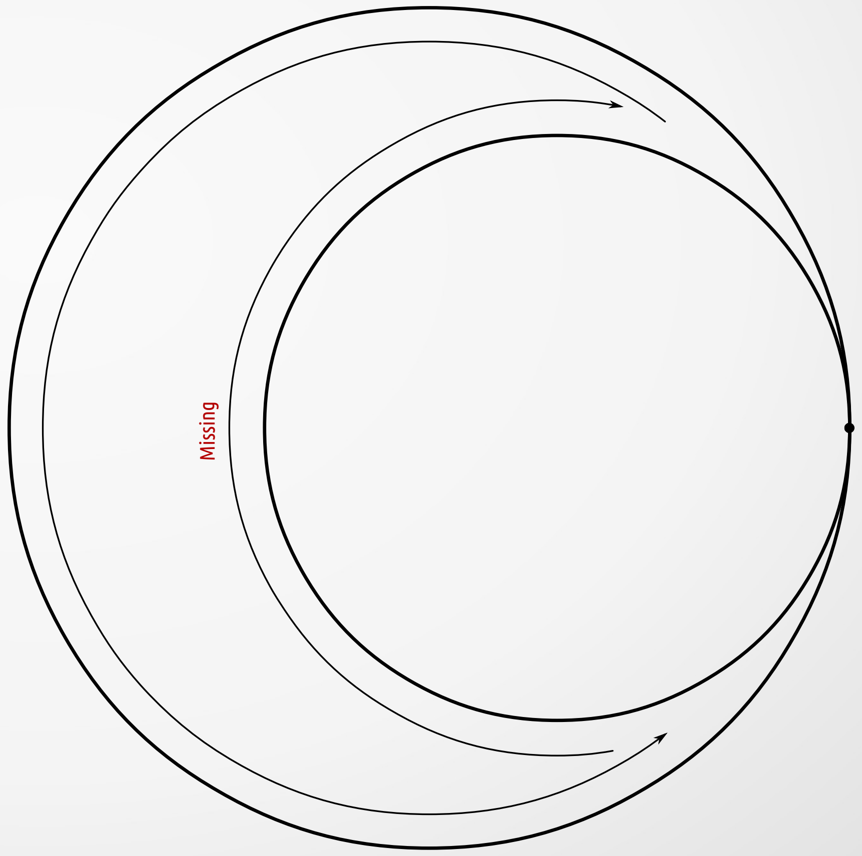
Summary

Contributions:

- Extending the subdivision algorithm beyond straight-line
- A prototype of the implementation

TODO:

- implementation efficiency – speed
- **degenerate cases**
- subdivision overlay (i.e. intersection)



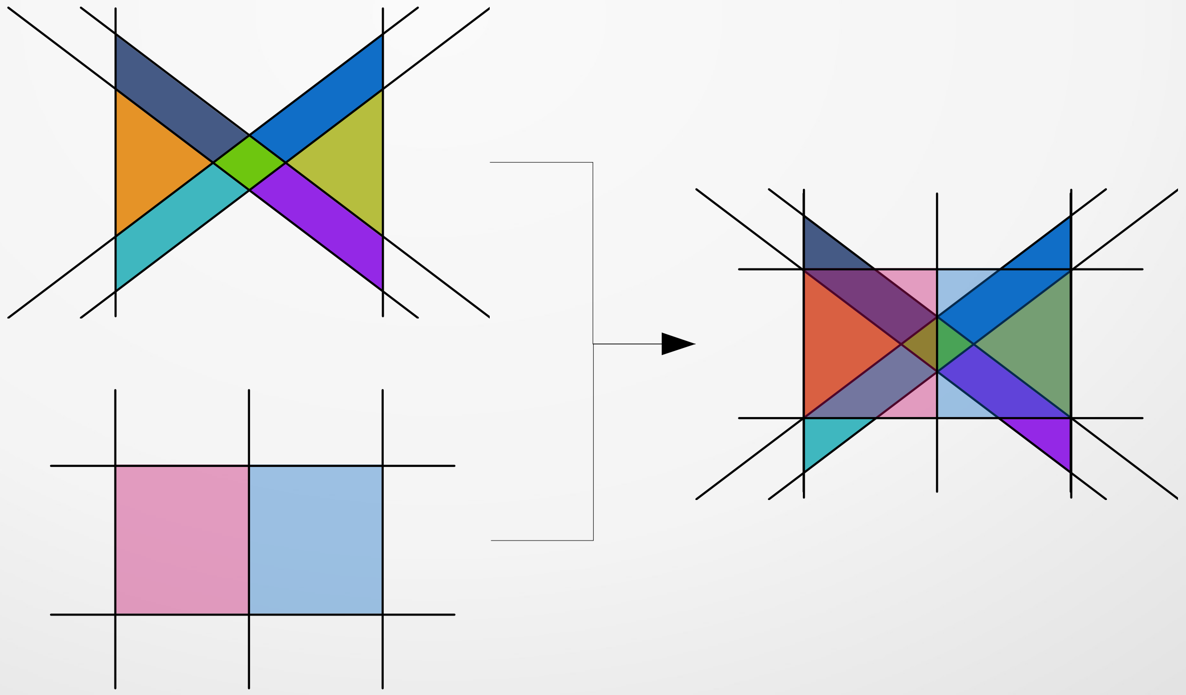
Summary

Contributions:

- Extending the subdivision algorithm beyond straight-line
- A prototype of the implementation

TODO:

- implementation efficiency – speed
- degenerate cases
- **subdivision overlay**



Future work

Challenges

- Identifying degenerate cases
- Dynamic subdivision
- Subdivision overlay – intersection, ...

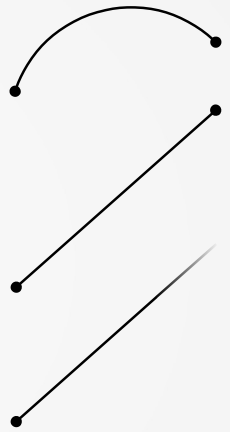
Future work

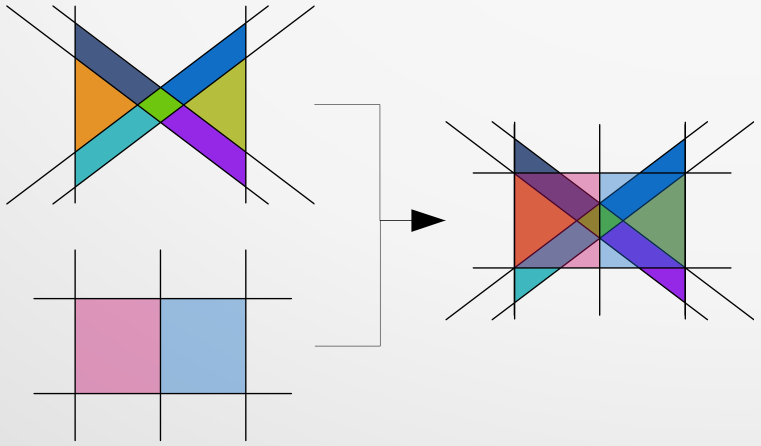
Challenges

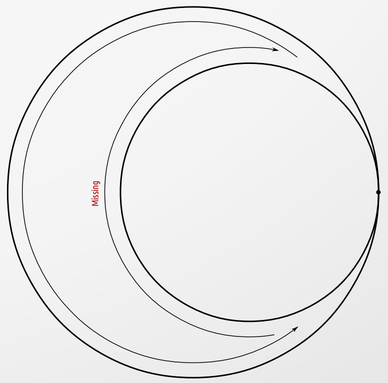
- Identifying degenerate cases
- Dynamic subdivision
- Subdivision overlay – intersection, ...

Implementation

- Handling degenerate cases,
- Including more practical curves
  - ✓ rays (half-line) and line segments
  - ✓ arcs, ellipses, …
- Extending the implementation to include “subdivision overlay”,





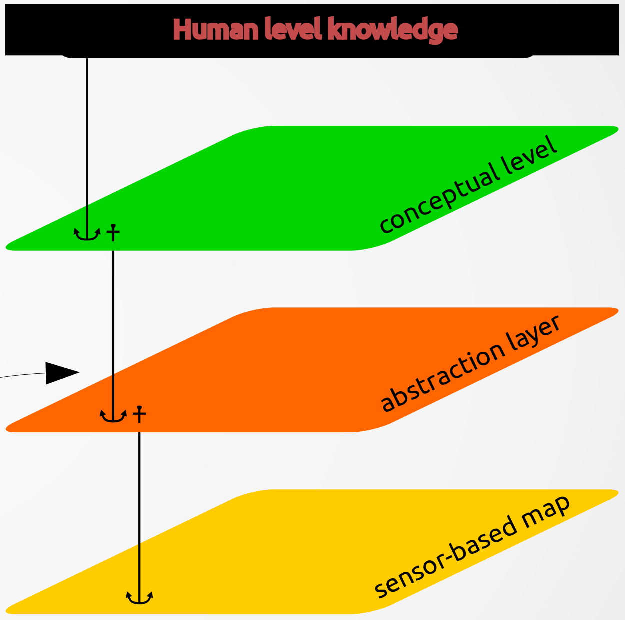


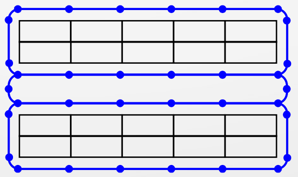
Future work

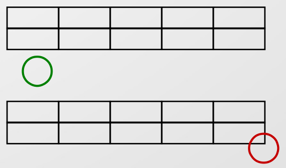
Challenges
- Identifying degenerate cases
- Dynamic subdivision
- Subdivision overlay – intersection, ...

Implementation
- Handling degenerate cases,
- Including more practical curves
  - ✓ rays (half-line) and line segments
  - ✓ arcs, ellipses, …
- Extending the implementation to include "subdivision overlay",

In Application
- Deploying in a multi-layer semantic map, (also documenting a proper API)
- Integrating robotics related feature,
  - ✓ collision detection,
  - ✓ path planning libraries,...







Thank you