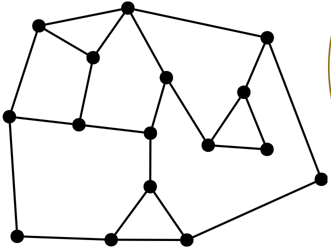
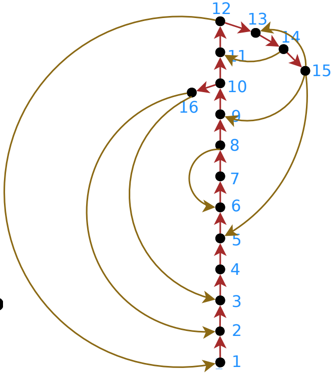




Efficient Planarity Testing

Mirko Kugelmeier

Motivation

- Planarity testing has many applications
- Planar graphs have interesting properties
- Planarity testing should be fast

Motivation

- Planarity testing has many applications
  - Circuit design
  - Chemistry: most molecules are representable as planar graphs
  - Highway design
  - ...
- Planar graphs have interesting properties
- Planarity testing should be fast

Motivation

- Planarity testing has many applications
  - Circuit design
  - Chemistry: most molecules are representable as planar graphs
  - Highway design
  - ...
- Planar graphs have interesting properties
  - Can be stored efficiently
  - Can be 4-colored
  - Many problems become easier (e.g. graph isomorphism in P)
  - ...
- Planarity testing should be fast

An Algorithm To Test Planarity

- Developed in 1974 by John Hopcroft and Robert Tarjan
- Extensive use of depth-first search
- Works by embedding paths in the graph into the plane one-by-one

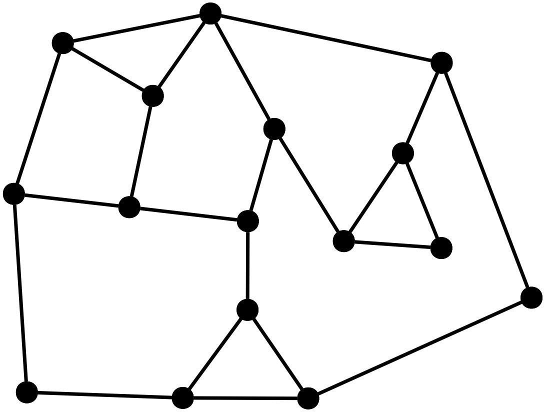
An Algorithm To Test Planarity

- Developed in 1974 by John Hopcroft and Robert Tarjan
- Extensive use of depth-first search
- Works by embedding paths in the graph into the plane one-by-one
  - Dismiss graphs with $|E| > 3|V| - 3$
  - Divide graph into biconnected components
  - Convert component into palm tree and find paths
  - Find cycle, delete it and check planarity of disconnected segments by embedding them in the plane.

Structuring

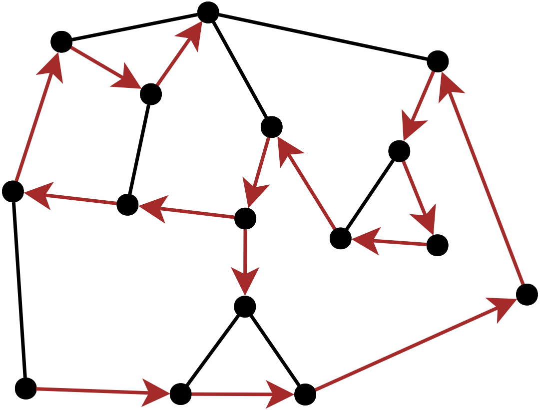
- Graph must be structured to allow embedding algorith to work
- Depth-first search
  - Numbers vertices
  - Imposes direction on edges

Structuring



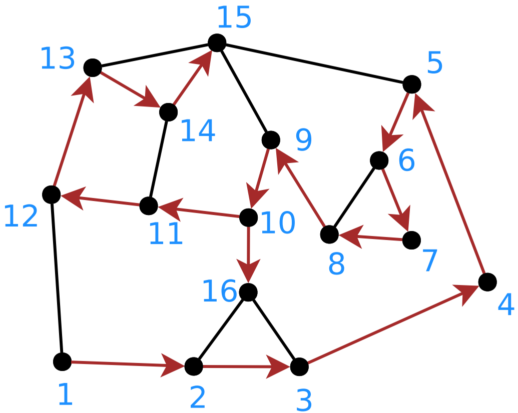
To determine the planarity of this graph we must first structure it.

Depth-first search



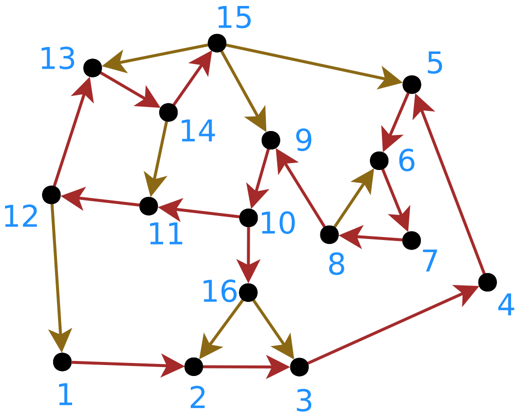
A Depth-first search imposes a first structure on the graph.

Depth-first search



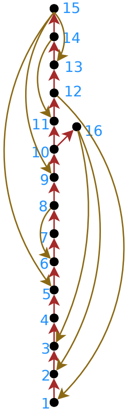
We number the vertices in search order of the DFS.

Depth-first search



The DFS partitions edges into tree arcs and fronds.

Depth-first search

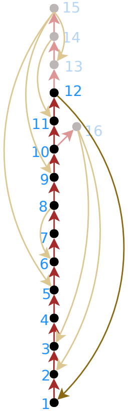


This gives the graph a palm tree structure.

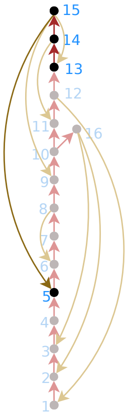
Pathfinding

- Later embedding steps must be ordered in a specific way
- We determine paths in the graph with special properties
- Each Path consists of a frond and zero or more tree arcs
- Paths can be found in $O(V + E)$ with another DFS.

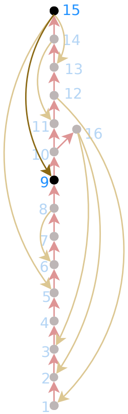
Pathfinding



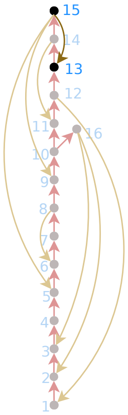
Pathfinding



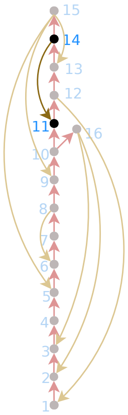
Pathfinding



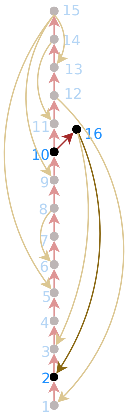
Pathfinding



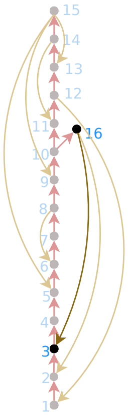
Pathfinding



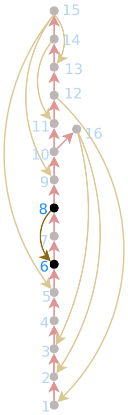
Pathfinding



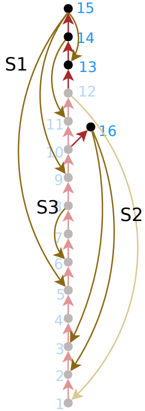
Pathfinding



Pathfinding



Pathfinding

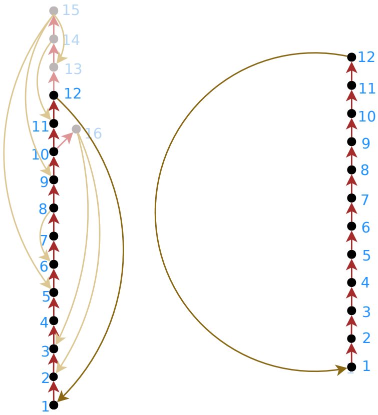


The graph falls into segments by removing the first generated path.

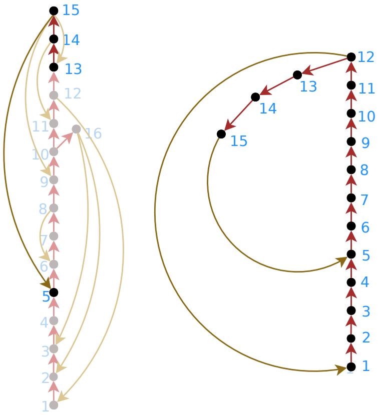
Embedding

- Paths are embedded in the order they are found.
- Segments are embedded in order with recursive application for 'sub-segments'
- Segments might need to be moved around in order to embed newer paths
- If reordering is impossible the graph is non-planar

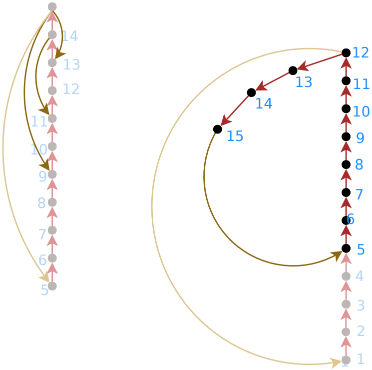
Embedding



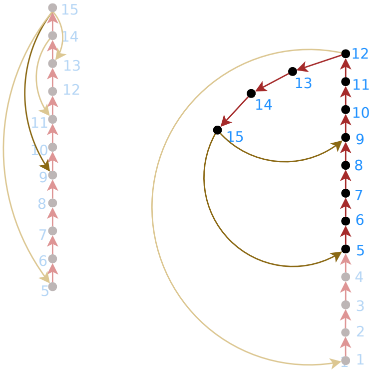
Embedding



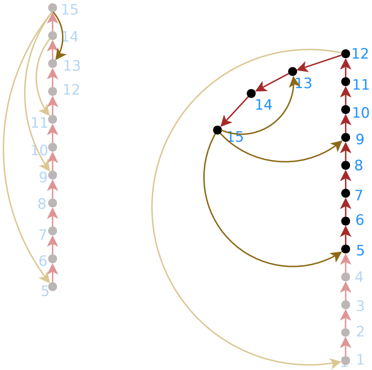
Embedding



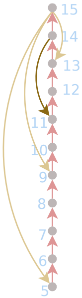
Embedding

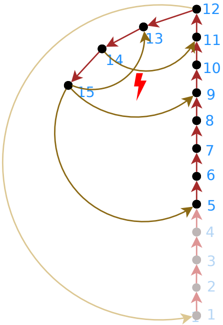


Embedding

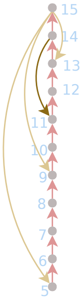


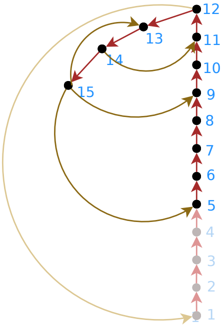
Embedding



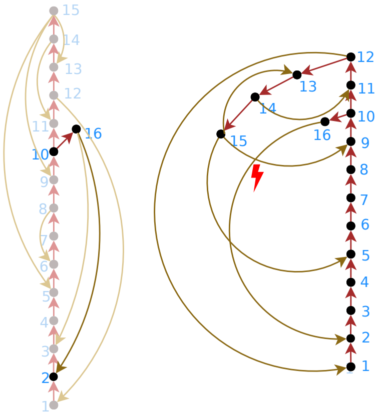


Embedding

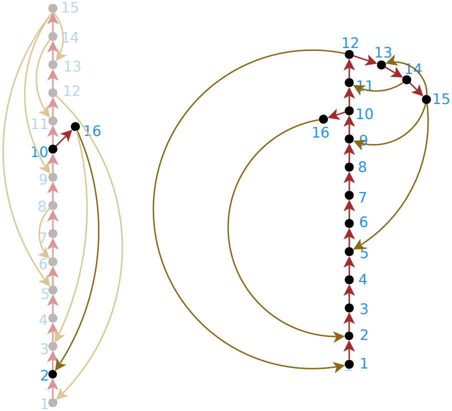




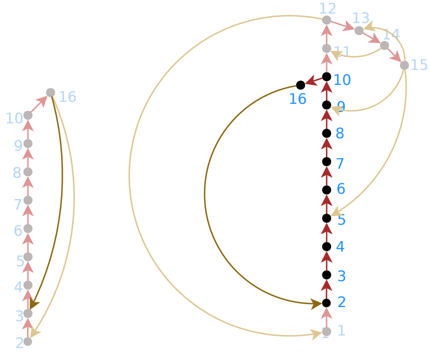
Embedding



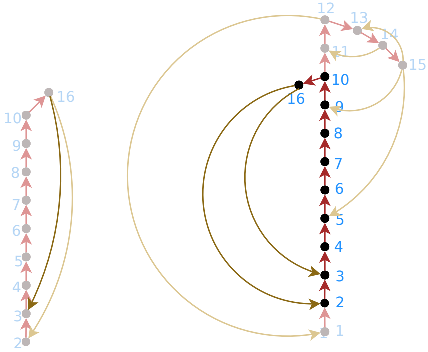
Embedding



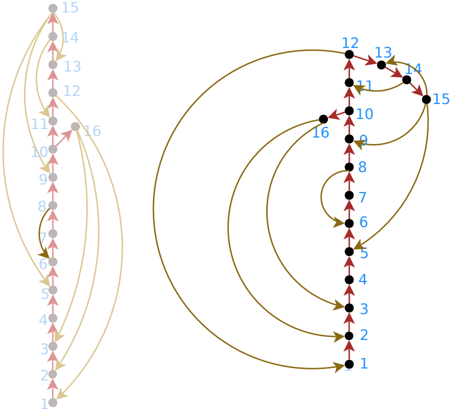
Embedding



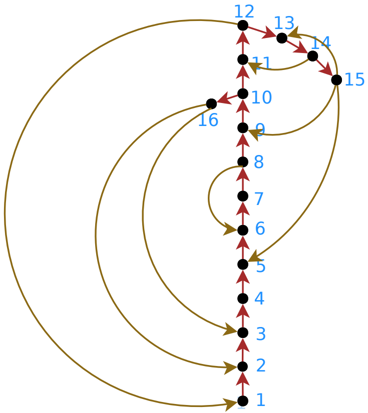
Embedding



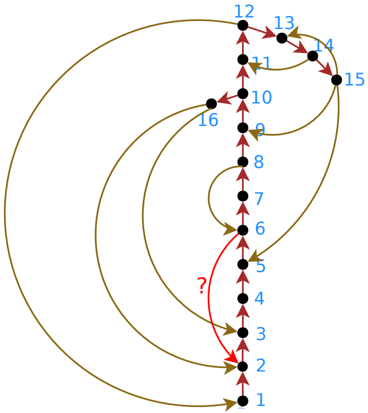
Embedding



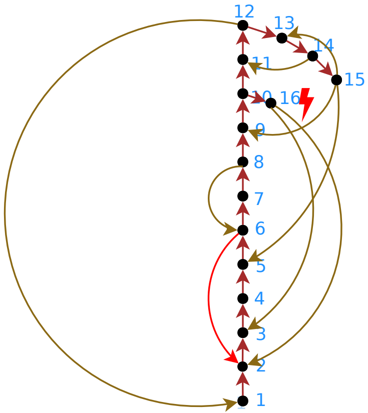
Embedding



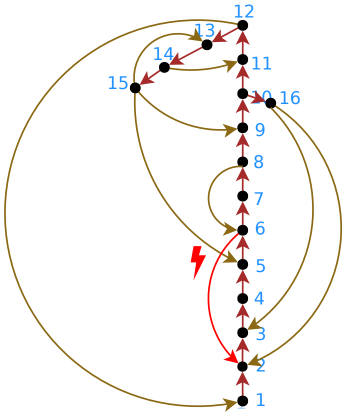
Non-Planar



Non-Planar



Non-Planar



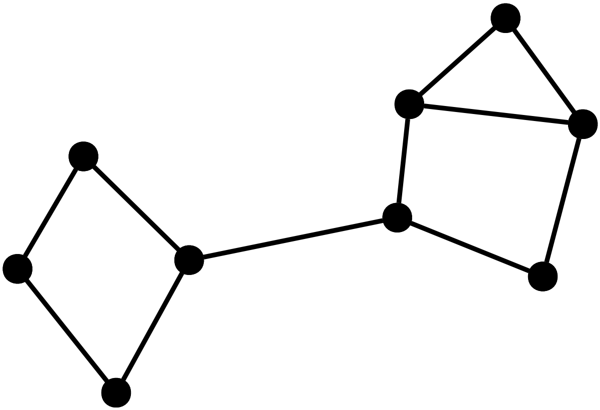
Complexity

- Division into biconnected components: O(V + E)
- Structuring the graph with DFS: O(V + E)
- Pathfinding with DFS: O(V + E)
- Embedding the paths: O(V + E)
- All graphs with $|E| > 3|V| - 3$ are none-planar
- Therefore planarity testing is O(V)

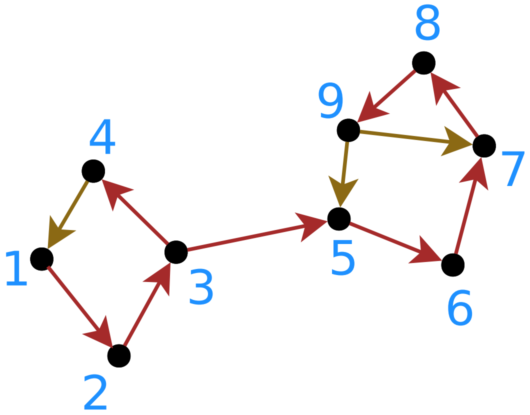
Biconnected Components

- A graph is biconnected if the graph is still connected after removing any vertex
- Every graph can be divided into a set of biconnected components
- Components can be found with a DFS

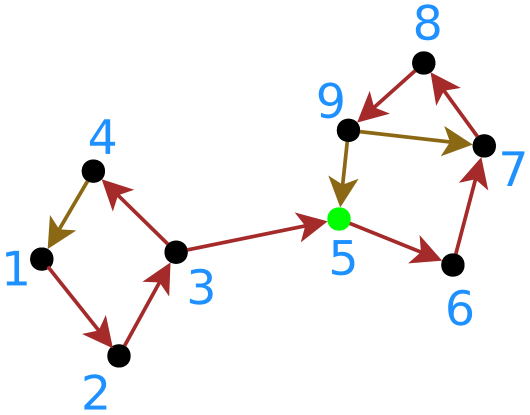
Finding Biconnected Components



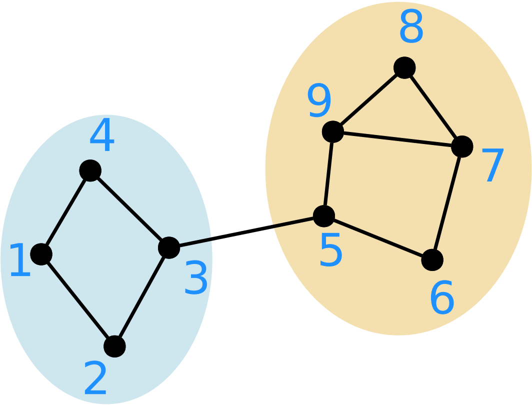
Finding Biconnected Components



Finding Biconnected Components



Finding Biconnected Components



References

- Hopcroft, John, and Tarjan, Robert. "Efficient planarity testing." Journal of the ACM (JACM) 21.4 (1974): 549-568.
- Hopcroft, John, and Tarja, Robert. "A V^2 algorithm for determining isomorphism of planar graphs." Inform. Processing Letters, 1, 1 (1971) 32-34
- Mehlhorn, Kurt, and Petra Mutzel. "On the embedding phase of the Hopcroft and Tarjan planarity testing algorithm." Algorithmica 16.2 (1996): 233-242.