Planar Graphs

and how to find them

Brendan Halstead

July 21, 2020

Outline

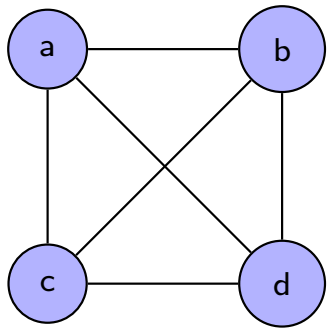
1. Definitions and Examples
2. Kuratowski's Theorem
3. DMP Algorithm
4. Other Surfaces?

What is a planar graph?

Planar embedding: drawing of a graph with no edges crossing
Planar graph: has a planar embedding

What is a planar graph?

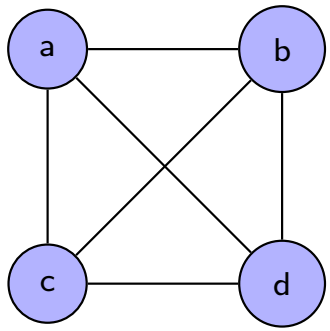
Planar embedding: drawing of a graph with no edges crossing
Planar graph: has a planar embedding



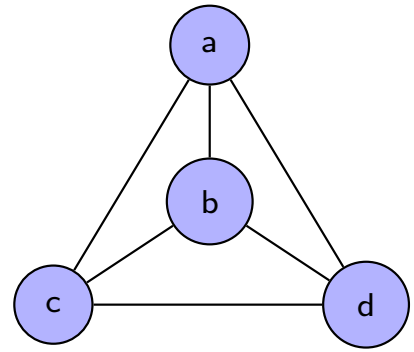
Nonplanar embedding

What is a planar graph?

Planar embedding: drawing of a graph with no edges crossing
Planar graph: has a planar embedding



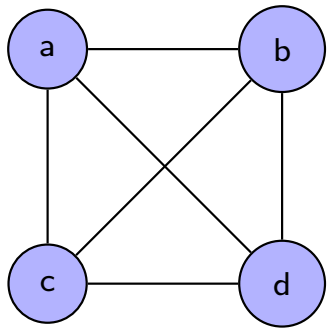
Nonplanar embedding



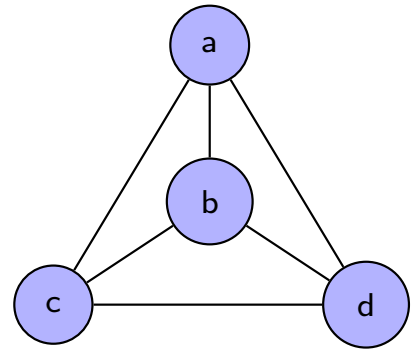
Planar embedding

What is a planar graph?

Planar embedding: drawing of a graph with no edges crossing
Planar graph: has a planar embedding



Nonplanar embedding

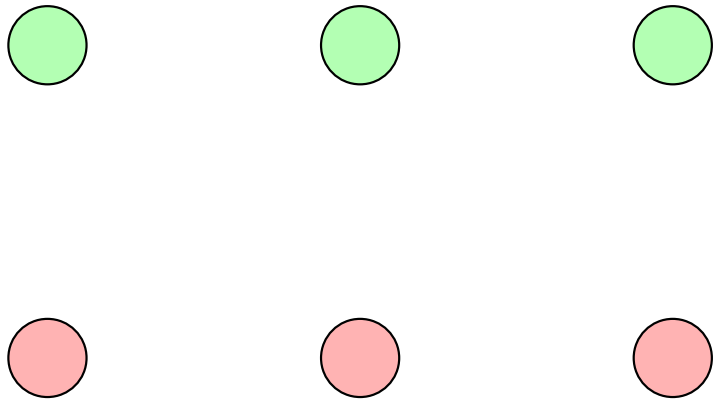


Planar embedding

K_4 is planar!

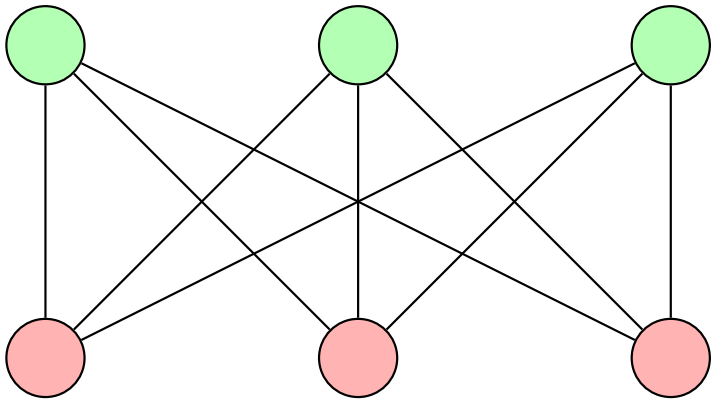
Another example

The complete bipartite graph $K_{3,3}$



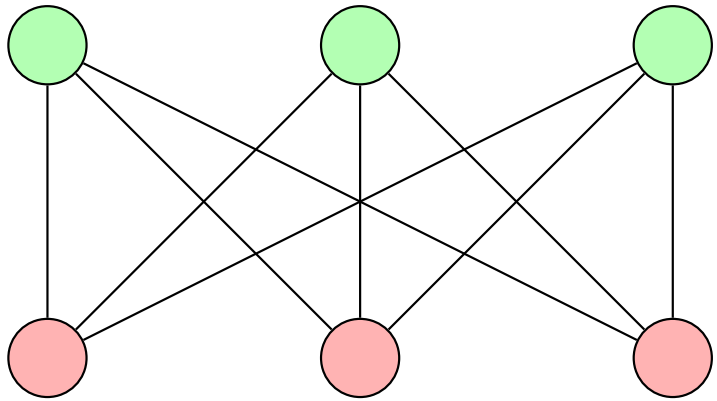
Another example

The complete bipartite graph $K_{3,3}$



Another example

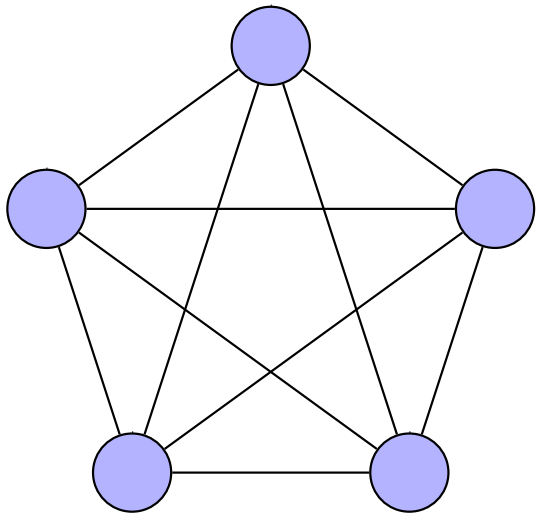
The complete bipartite graph $K_{3,3}$



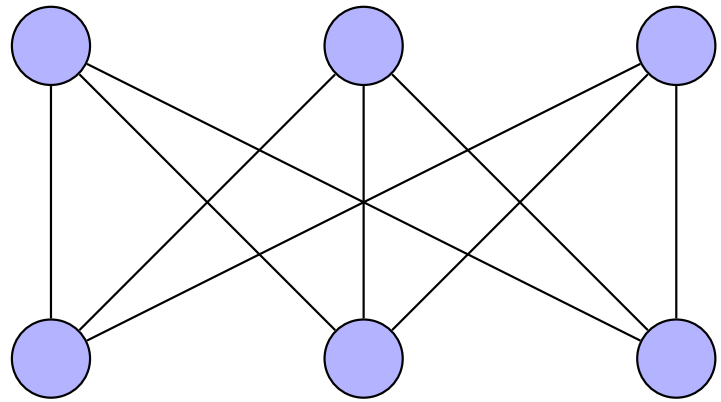
$K_{3,3}$ is **not** planar.

Kuratowski's Theorem

Theorem: A graph is planar if and only if it contains no subgraphs homeomorphic to K_5 or $K_{3,3}$.



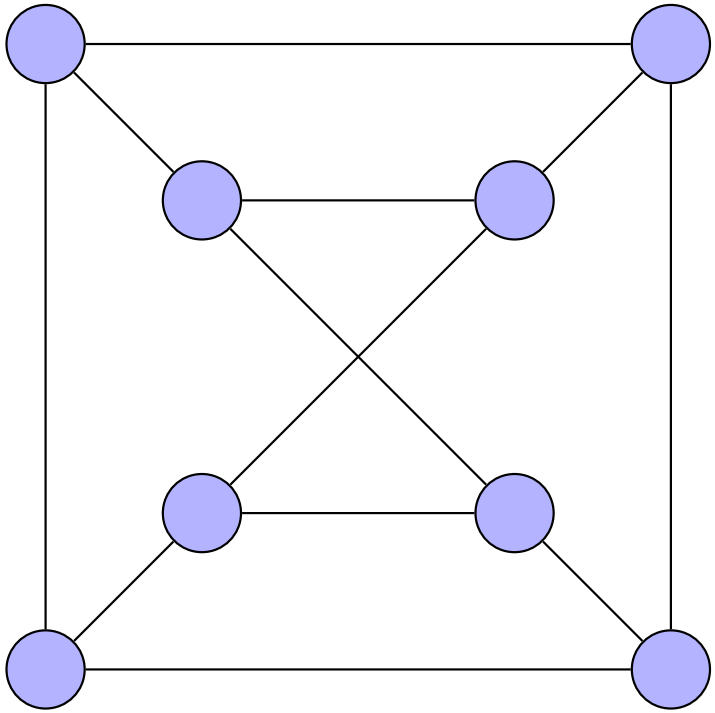
K_5



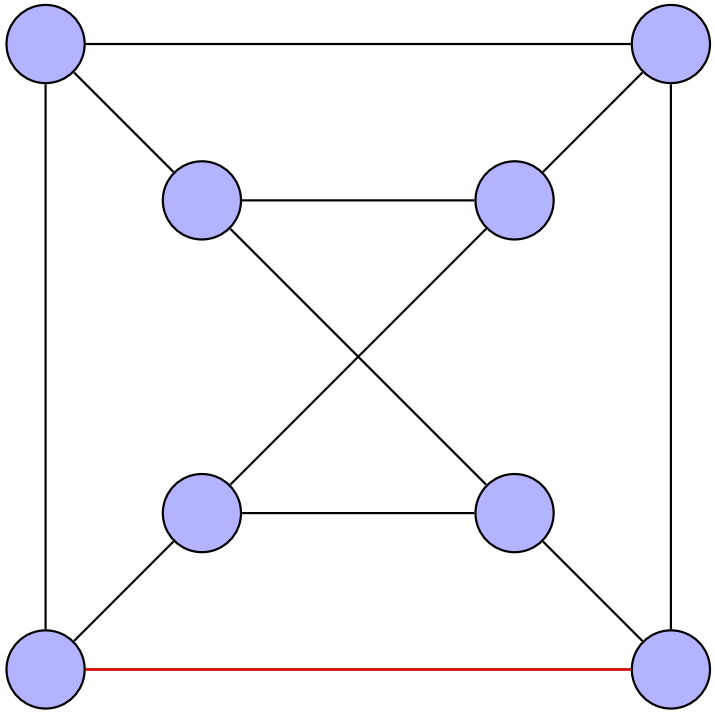
$K_{3,3}$

Homeomorphic to = has same basic shape as

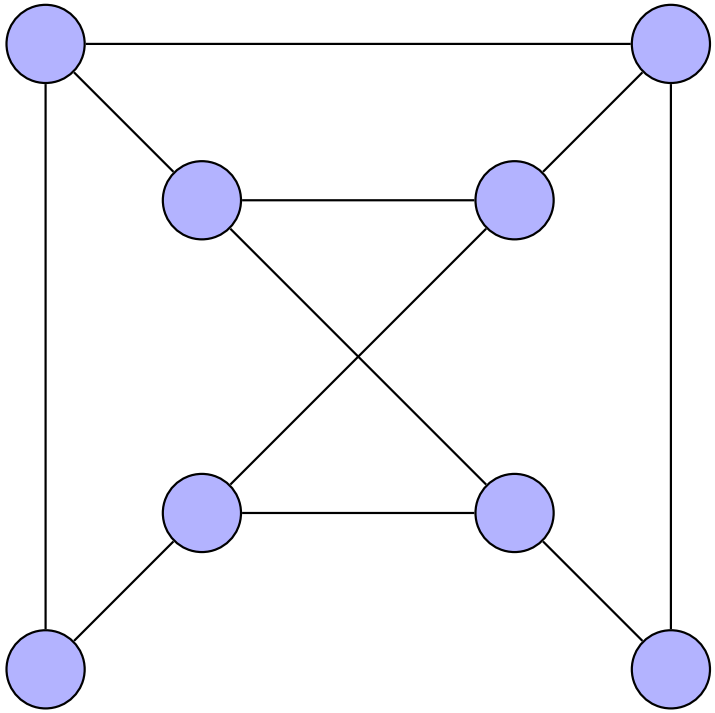
Example



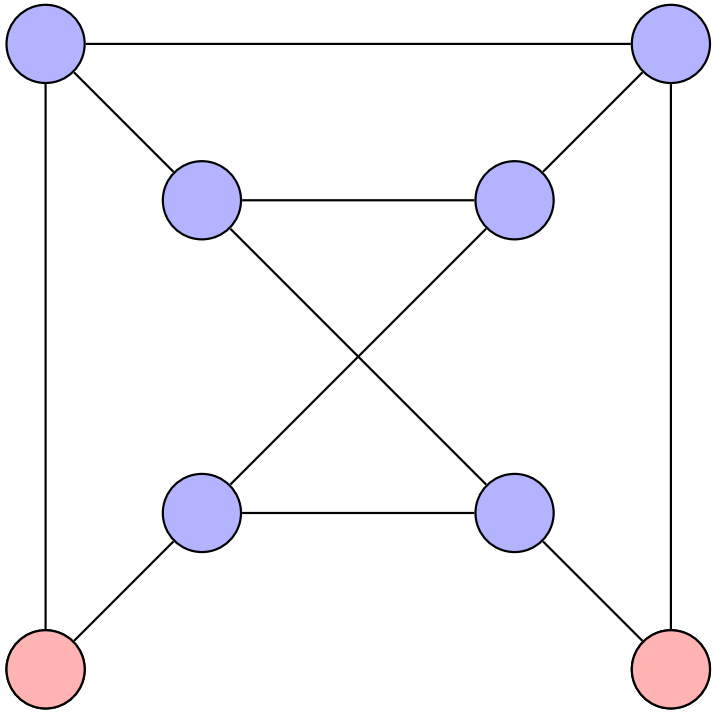
Example



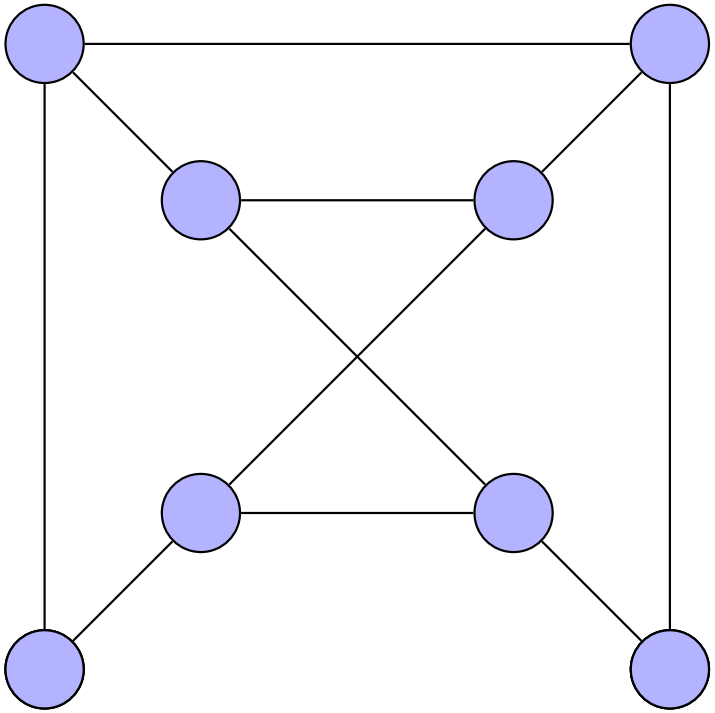
Example

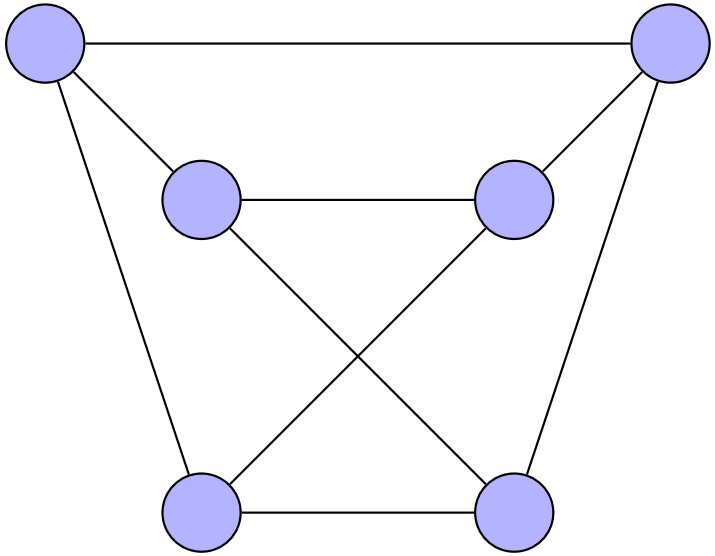


Example

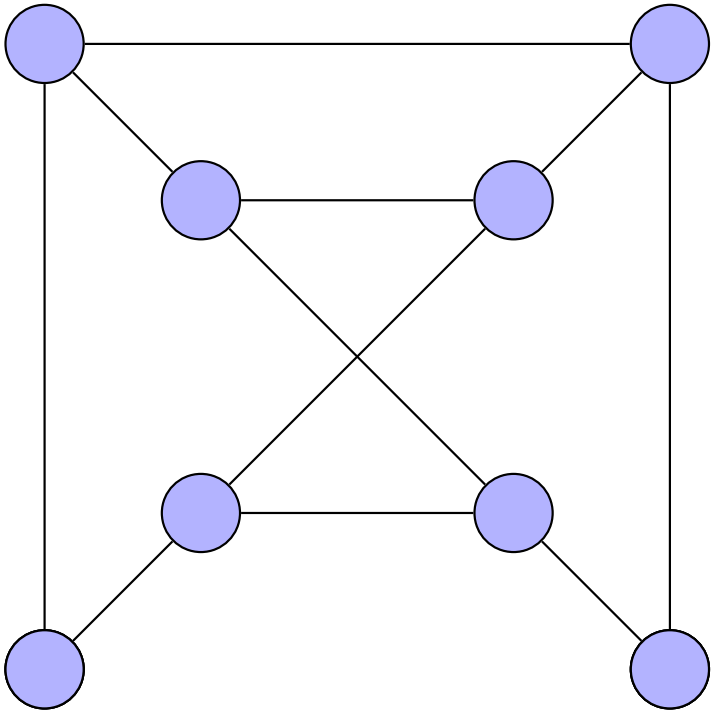


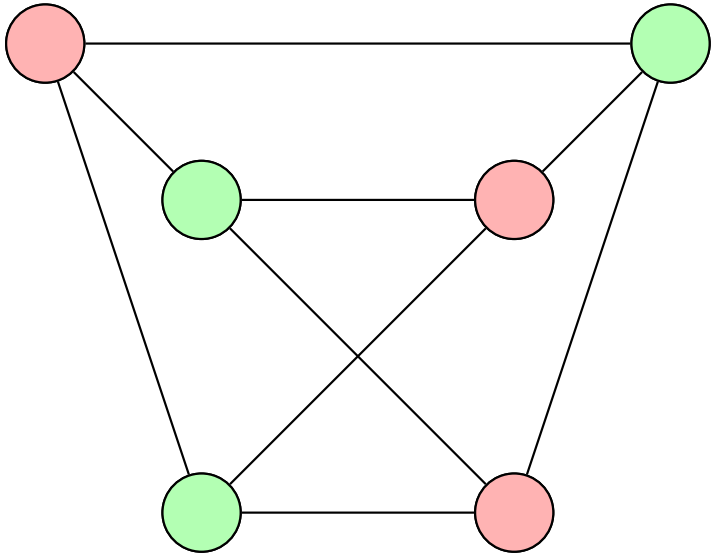
Example



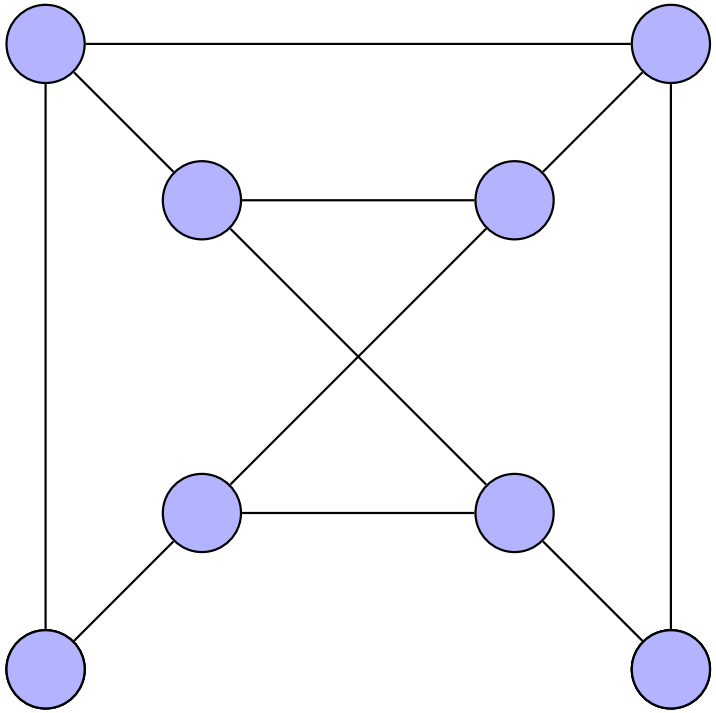


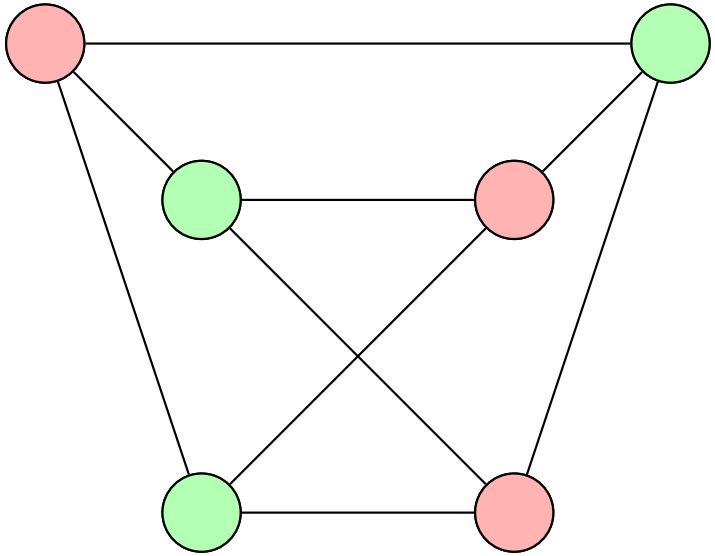
Example

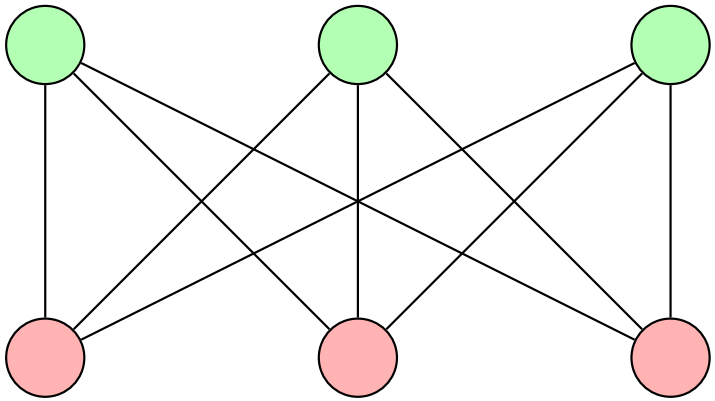




Example







Background

Published by Demoucron, Malgrange and Pertuiset in 1964

Works by adding segments to a subgraph until graph is complete or segment is blocked

Not the most efficient, but simplest to understand

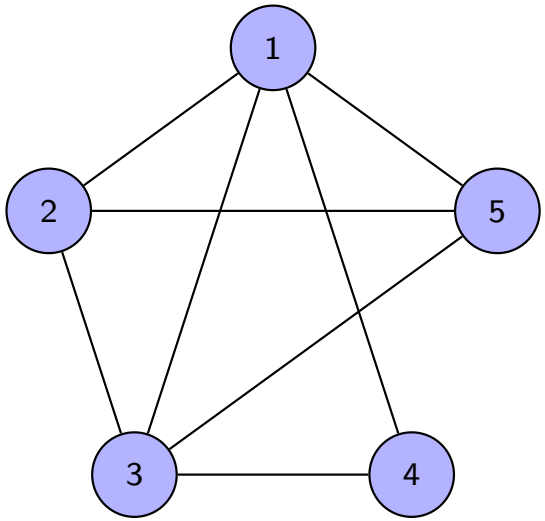
Requires pre-processing

DMP Planarity Algorithm

DMP Algorithm

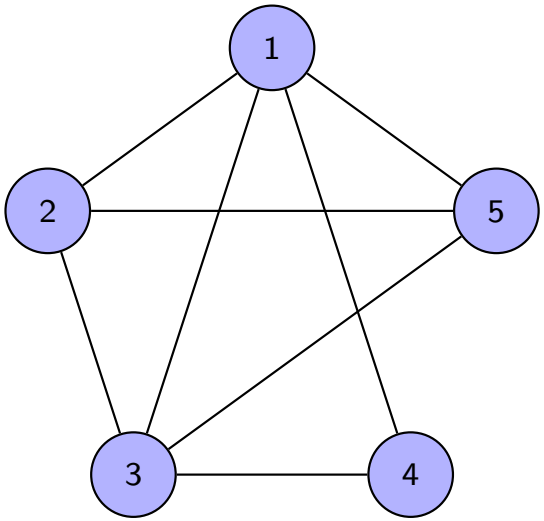
Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G

DMP Planarity Algorithm

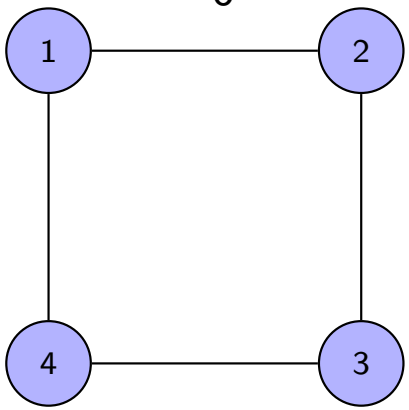


DMP Planarity Algorithm

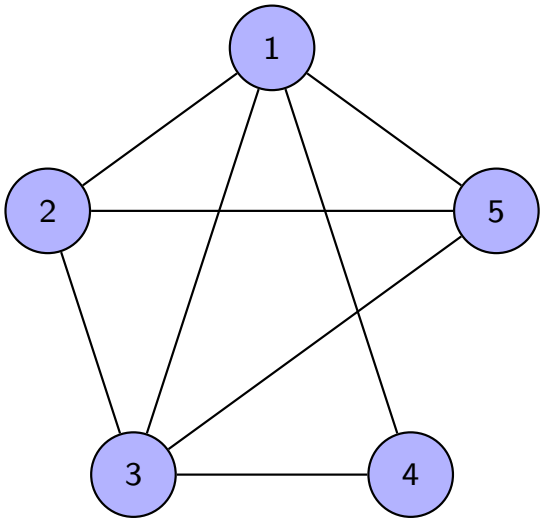
G

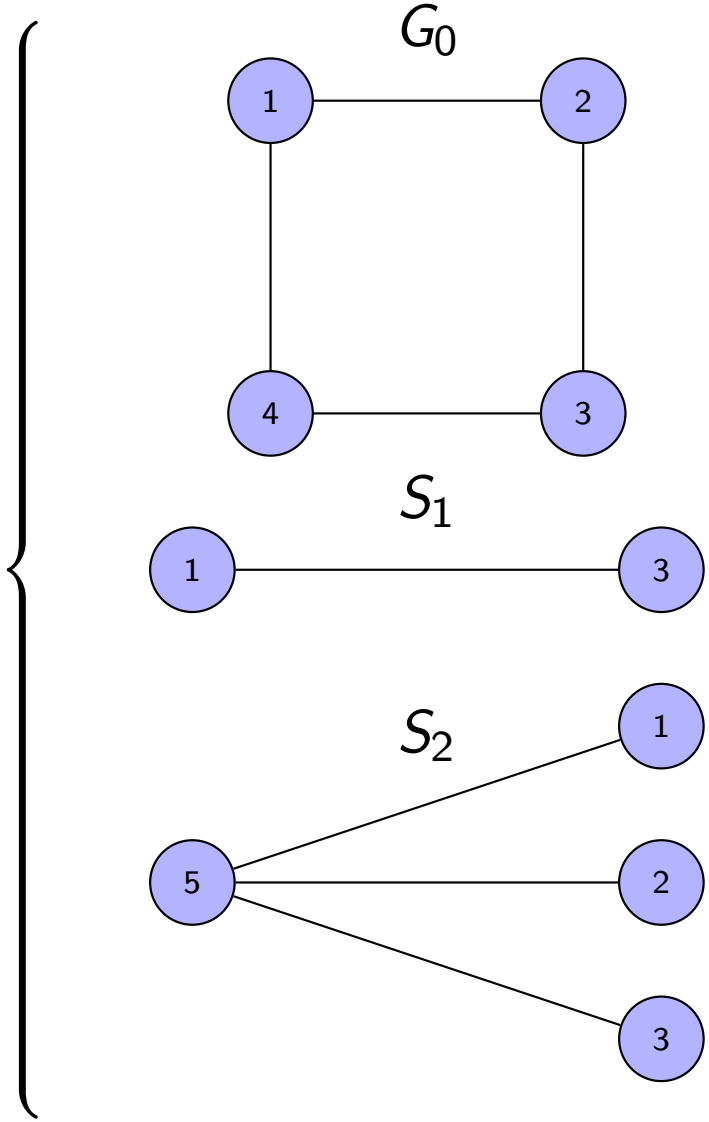


G_0



DMP Planarity Algorithm



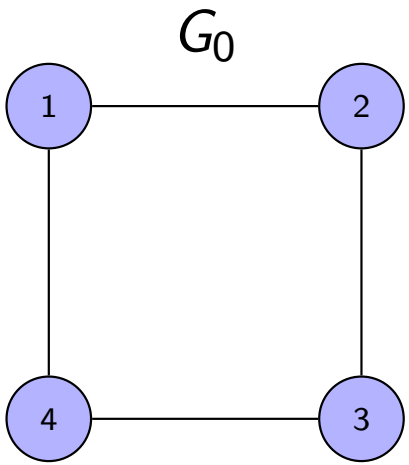


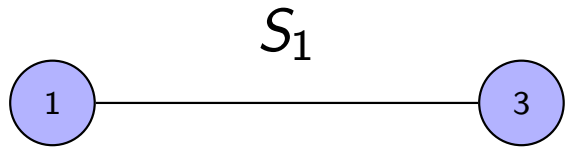
DMP Planarity Algorithm

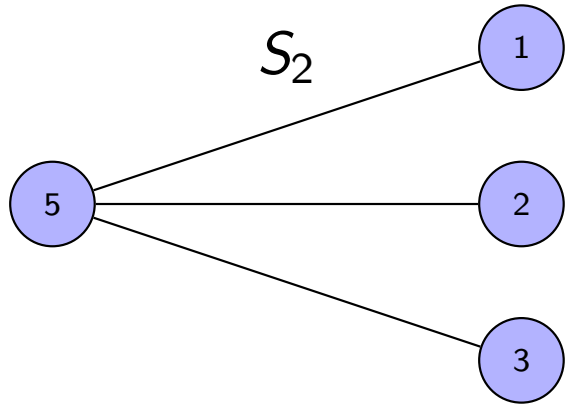
DMP Algorithm

Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G
while $G_j \neq G$ **do**
    if *any segment is blocked* **then**
        return FALSE
    end
end

DMP Planarity Algorithm





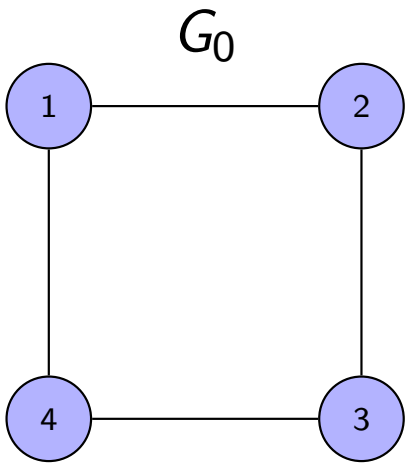


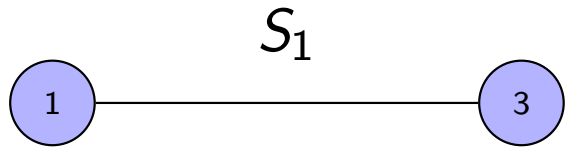
DMP Planarity Algorithm

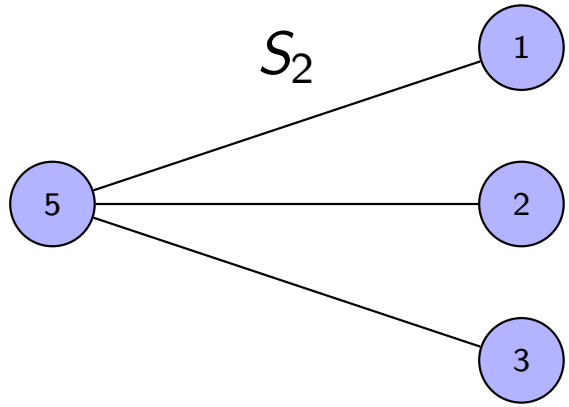
DMP Algorithm

Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G
while $G_j \neq G$ **do**
    if *any segment is blocked* **then**
        return FALSE
    end
    if *a segment is forced* **then**
        B := that segment
    end
    else
        B := any segment
    end
end

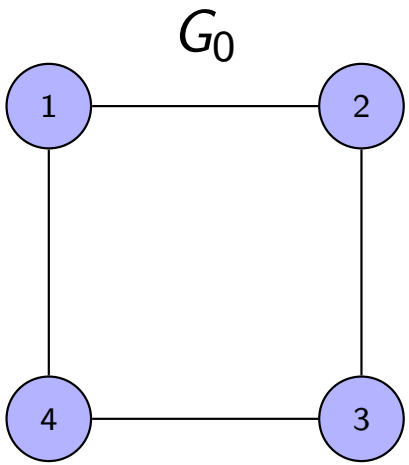
DMP Planarity Algorithm

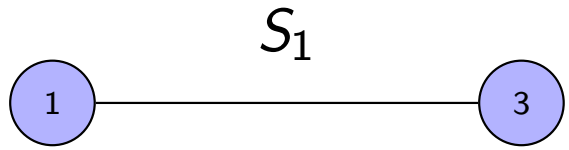


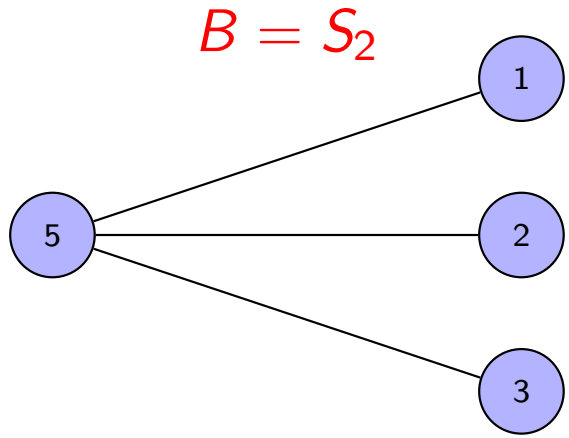




DMP Planarity Algorithm





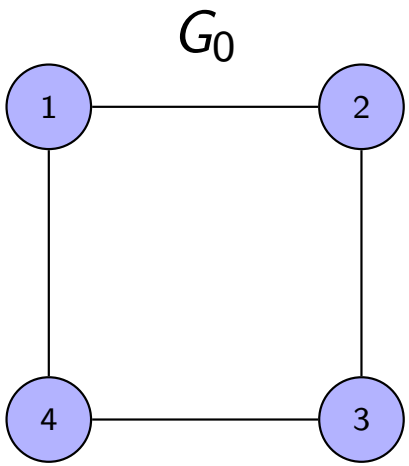


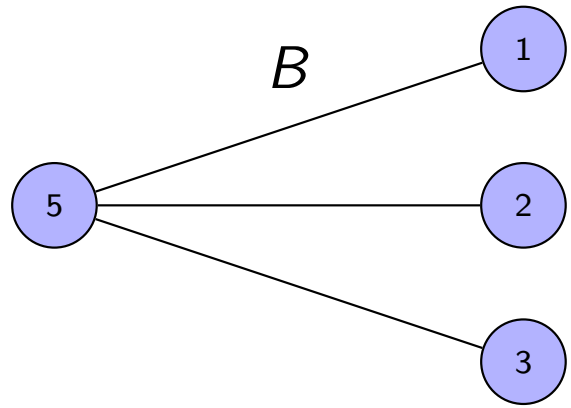
DMP Planarity Algorithm

DMP Algorithm

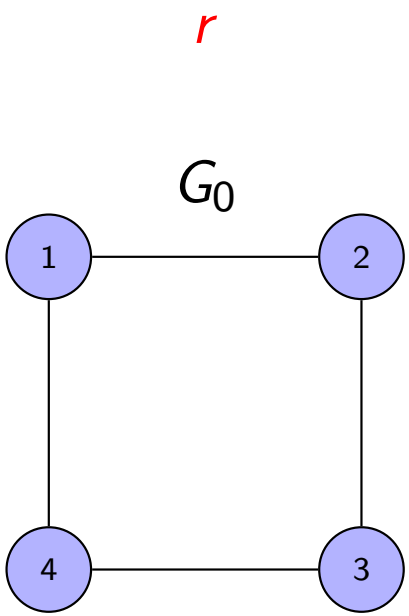
Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G
while $G_j \neq G$ **do**
    if *any segment is blocked* **then**
        return FALSE
    end
    if *a segment is forced* **then**
        B := that segment
    end
    else
        B := any segment
    end
    r := region whose boundary contacts B
end

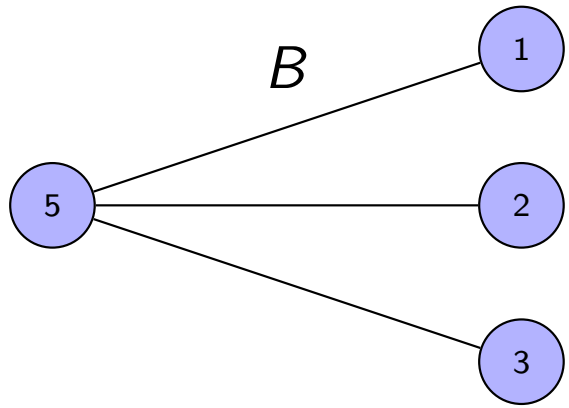
DMP Planarity Algorithm





DMP Planarity Algorithm



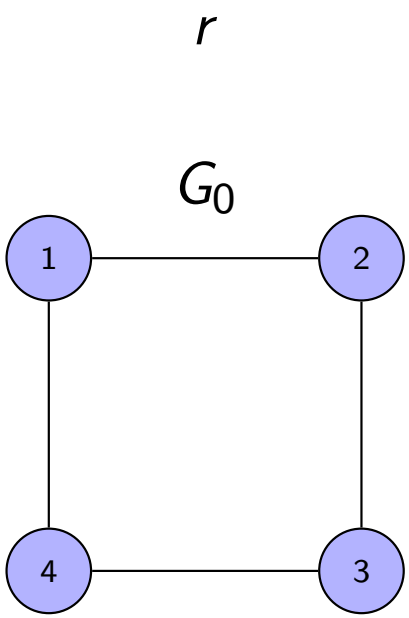


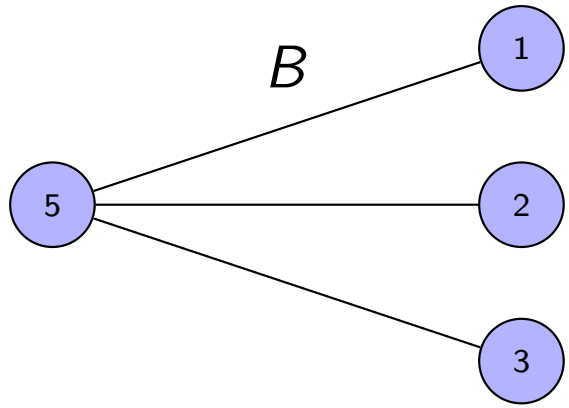
DMP Planarity Algorithm

DMP Algorithm

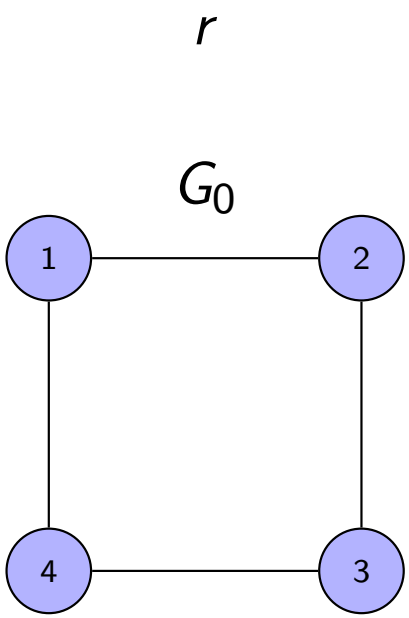
Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G
while $G_j \neq G$ **do**
    if *any segment is blocked* **then**
        return FALSE
    end
    if *a segment is forced* **then**
        B := that segment
    end
    else
        B := any segment
    end
    r := region whose boundary contacts B
    P := path between two contact points of B
end

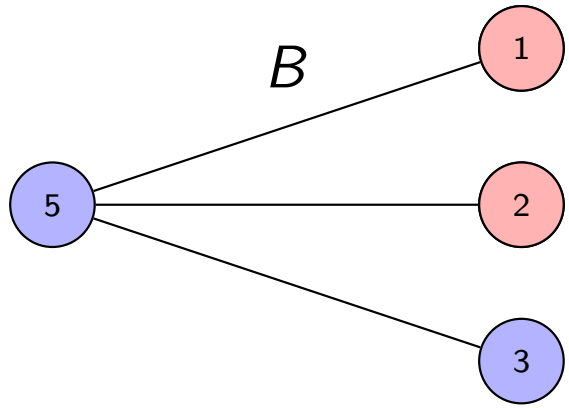
DMP Planarity Algorithm



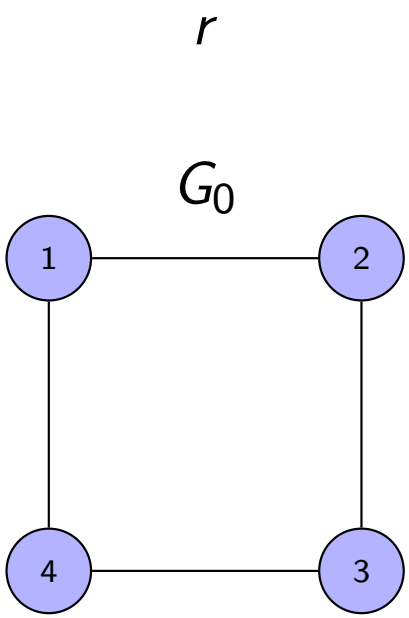


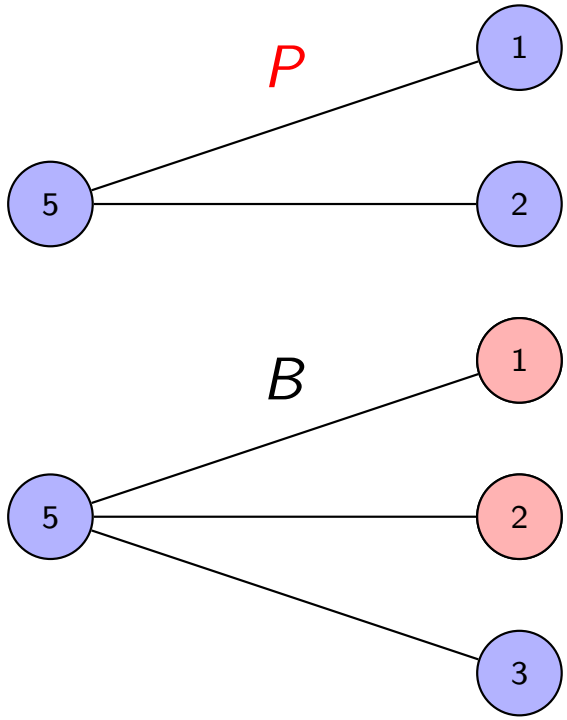
DMP Planarity Algorithm





DMP Planarity Algorithm



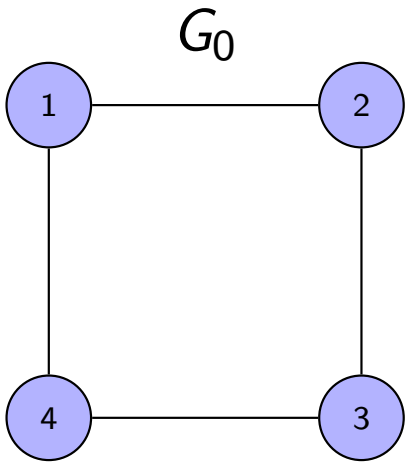


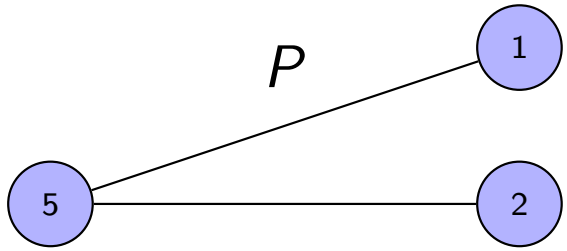
DMP Planarity Algorithm

DMP Algorithm

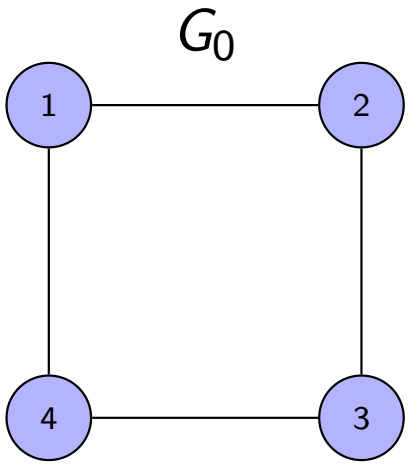
Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G
while $G_j \neq G$ **do**
    if *any segment is blocked* **then**
        return FALSE
    end
    if *a segment is forced* **then**
        B := that segment
    end
    else
        B := any segment
    end
    r := region whose boundary contacts B
    P := path between two contact points of B
    G_{j+1} := drawing of P in r
end

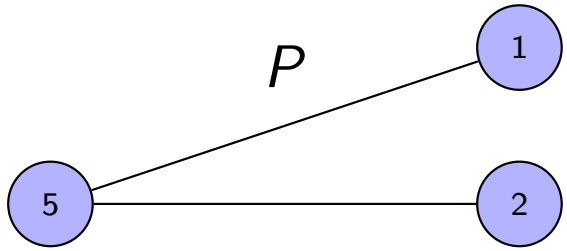
DMP Planarity Algorithm

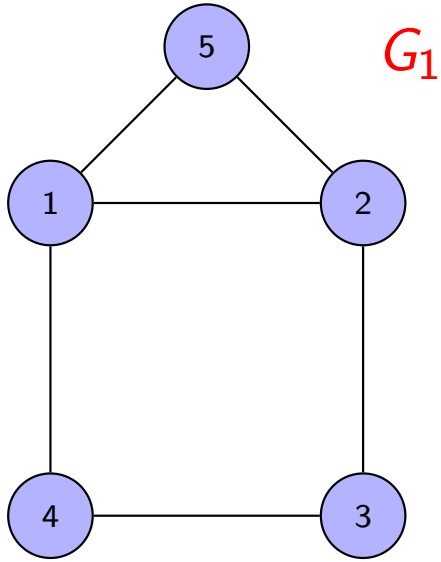




DMP Planarity Algorithm

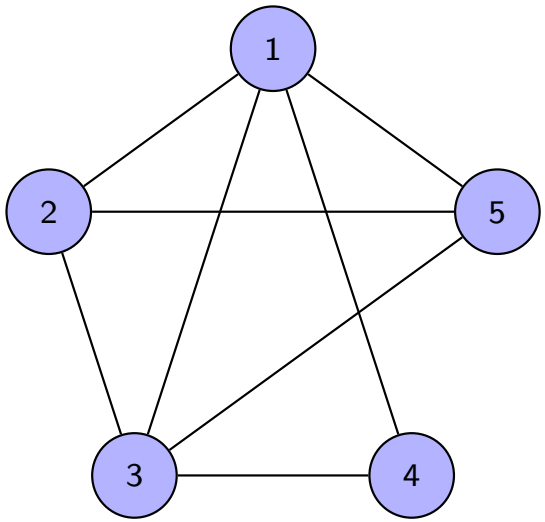


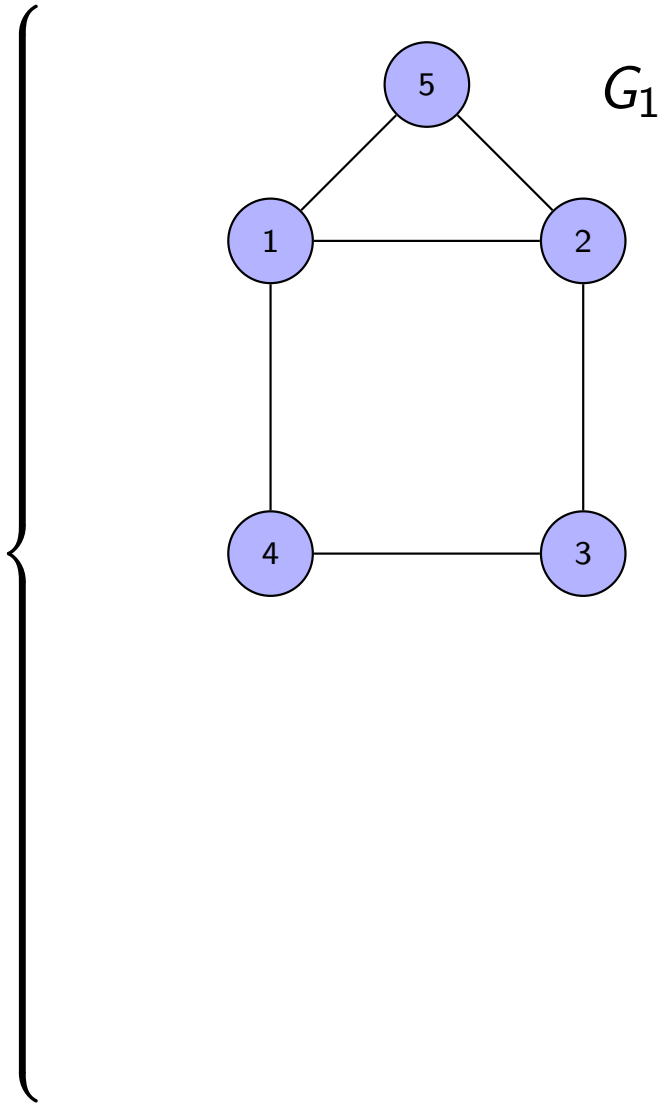




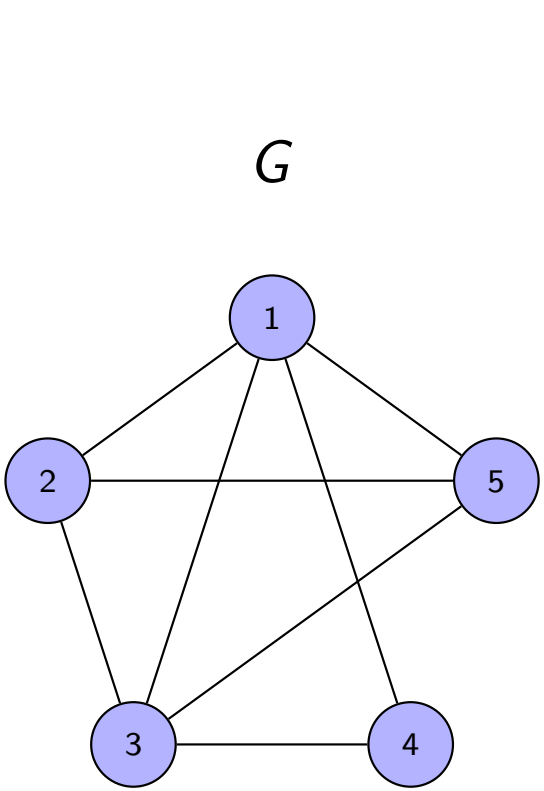
DMP Planarity Algorithm

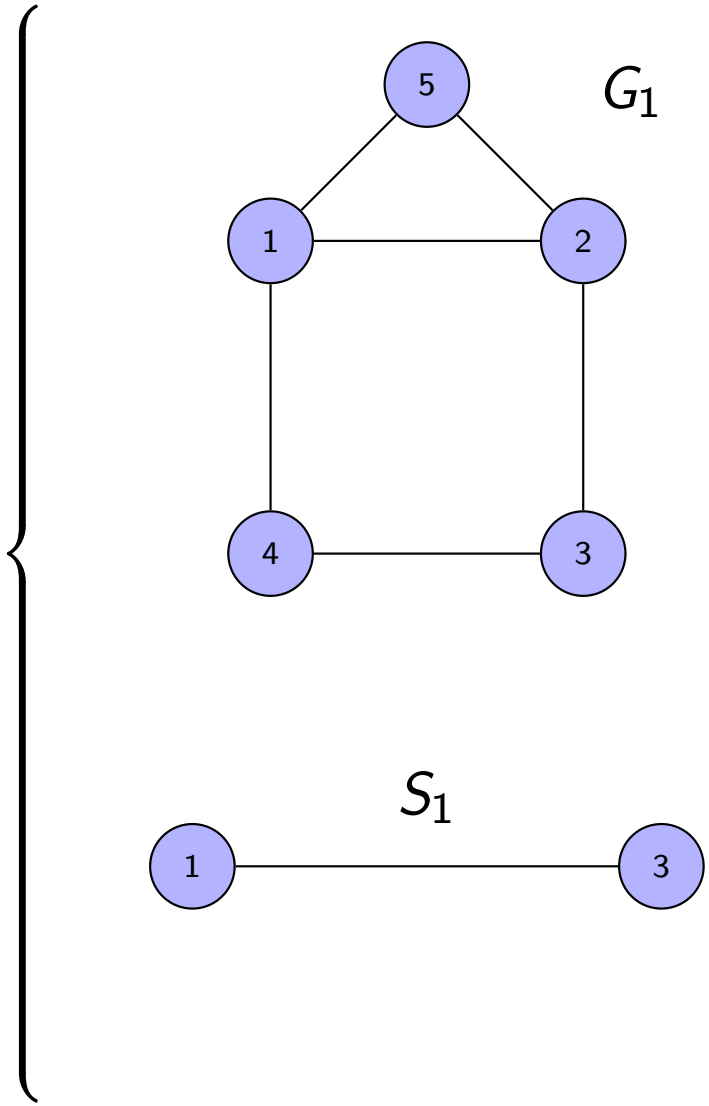
G



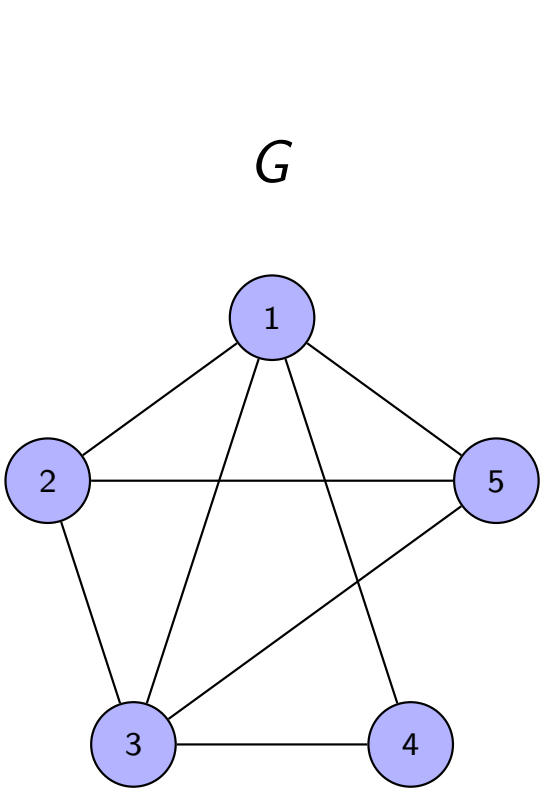


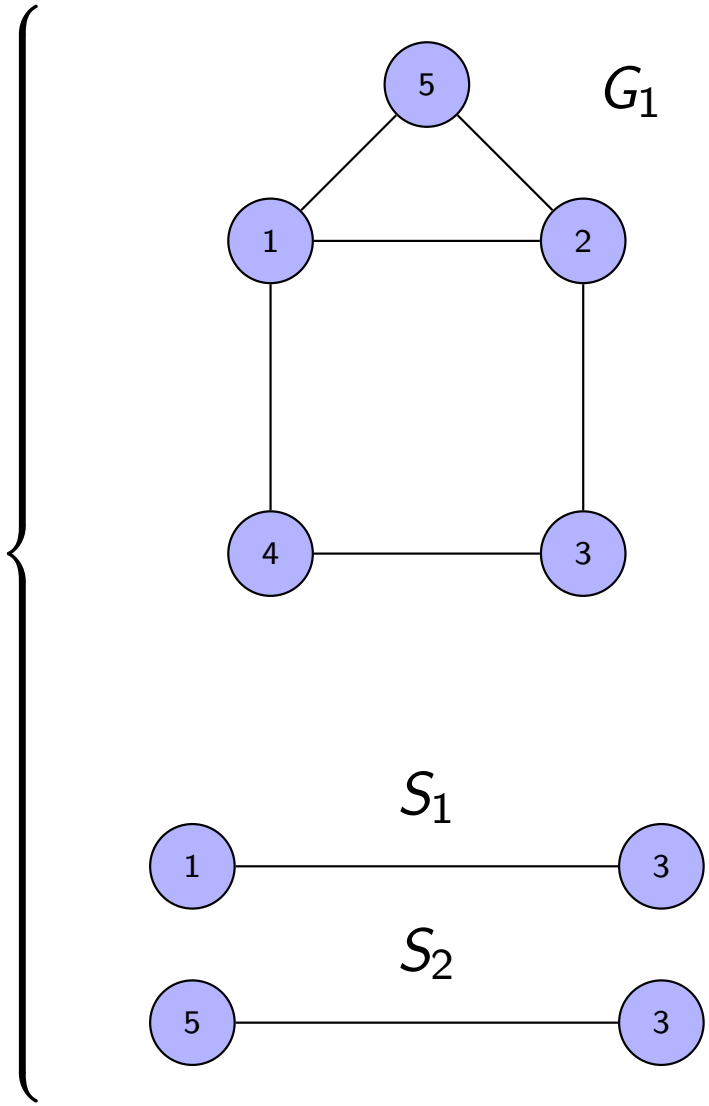
DMP Planarity Algorithm





DMP Planarity Algorithm



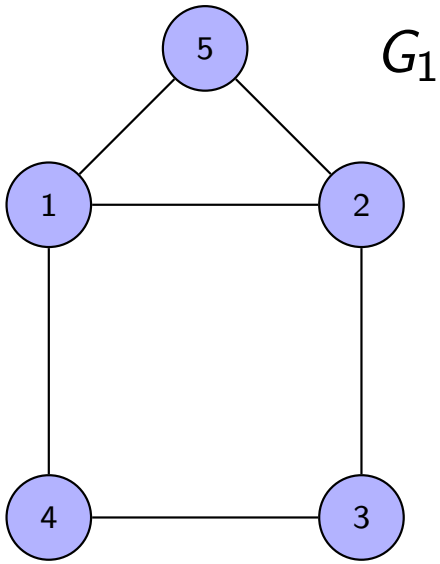


DMP Planarity Algorithm

DMP Algorithm

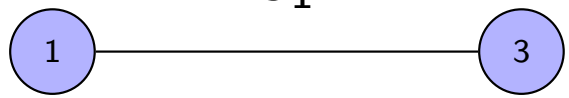
Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G
while $G_j \neq G$ **do**
    if *any segment is blocked* **then**
        return FALSE
    end
    if *a segment is forced* **then**
        B := that segment
    end
    else
        B := any segment
    end
    r := region whose boundary contacts B
    P := path between two contact points of B
    G_{j+1} := drawing of P in r
end

DMP Planarity Algorithm

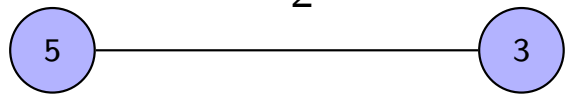


G_1

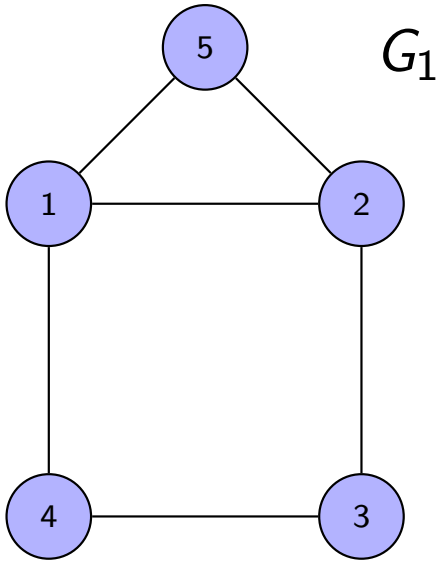
S_1



S_2

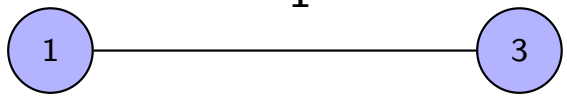


DMP Planarity Algorithm

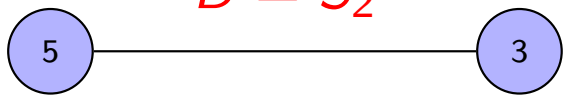


G_1

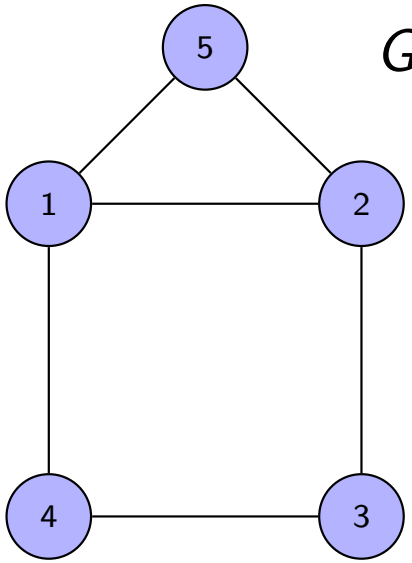
S_1



$B = S_2$

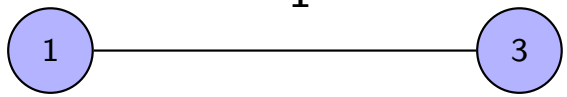


DMP Planarity Algorithm

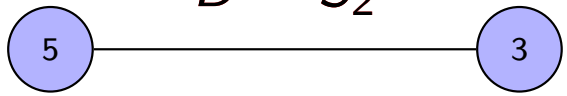


G_1 r

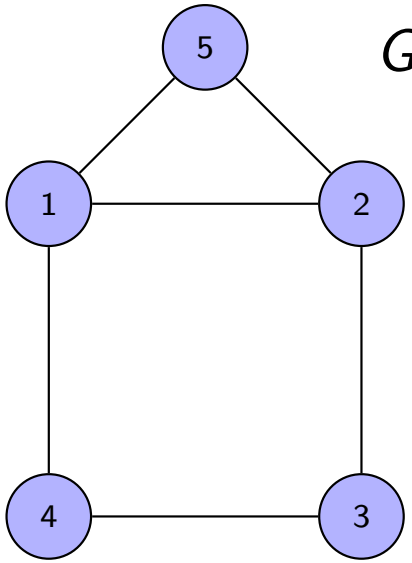
S_1



$B = S_2$

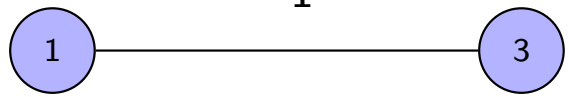


DMP Planarity Algorithm

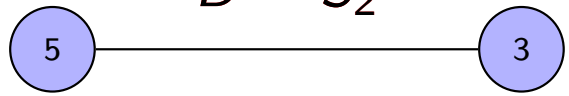


G_1 r

S_1

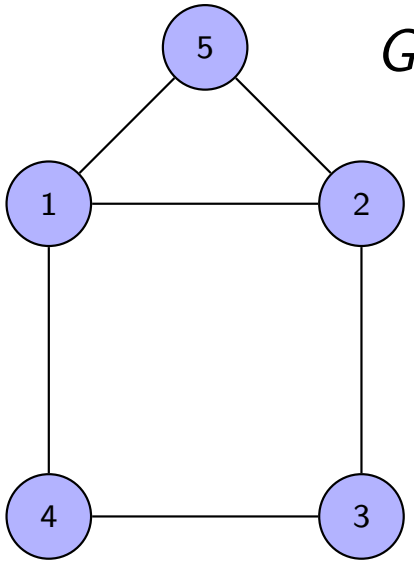


$B = S_2$

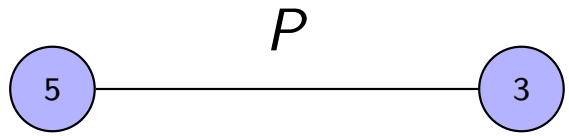


P

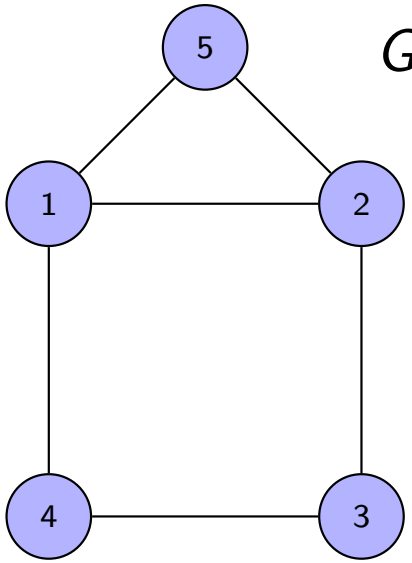
DMP Planarity Algorithm



G_1 r

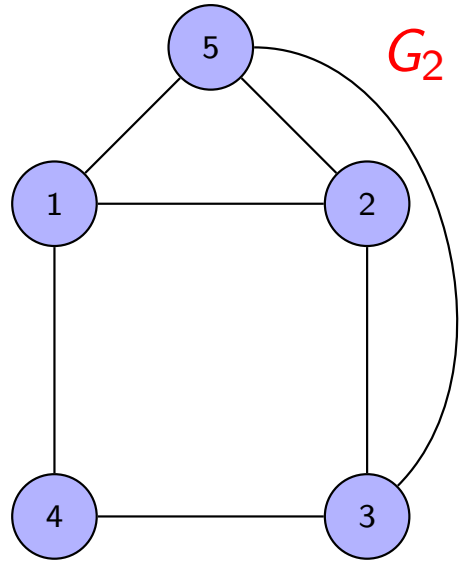


DMP Planarity Algorithm



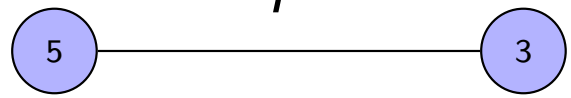
G_1

r



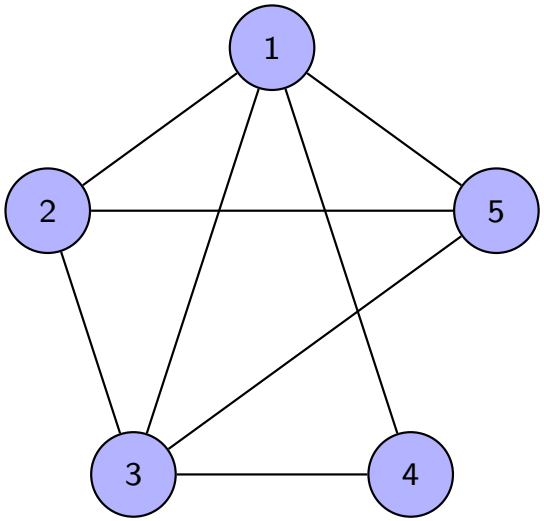
G_2

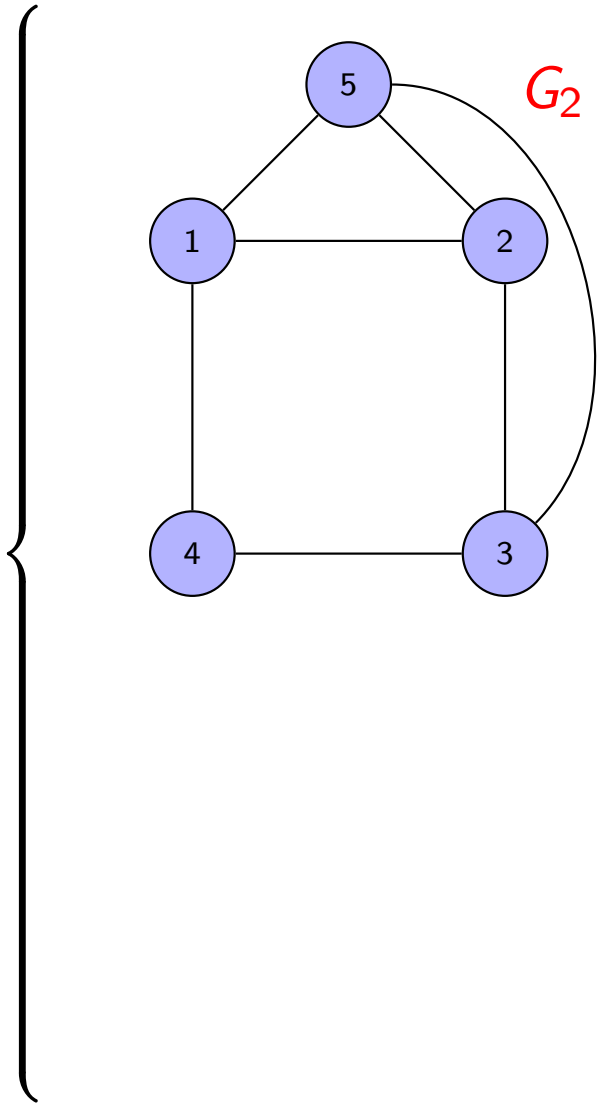
P



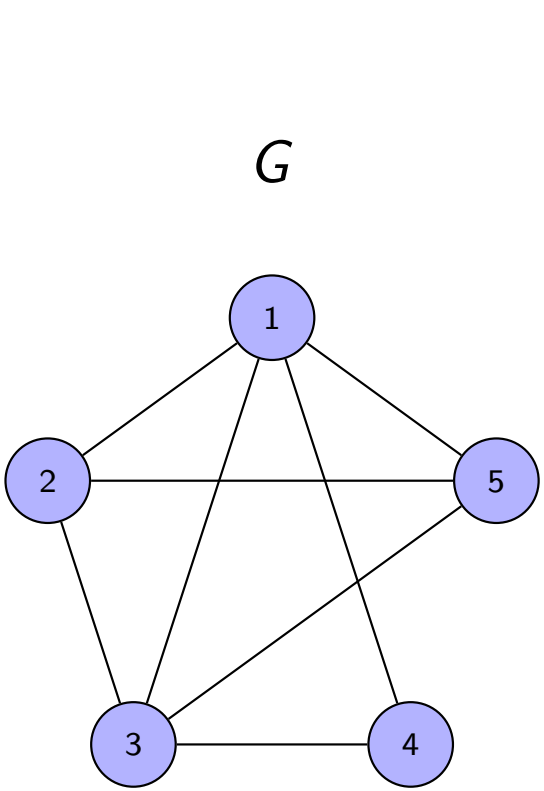
DMP Planarity Algorithm

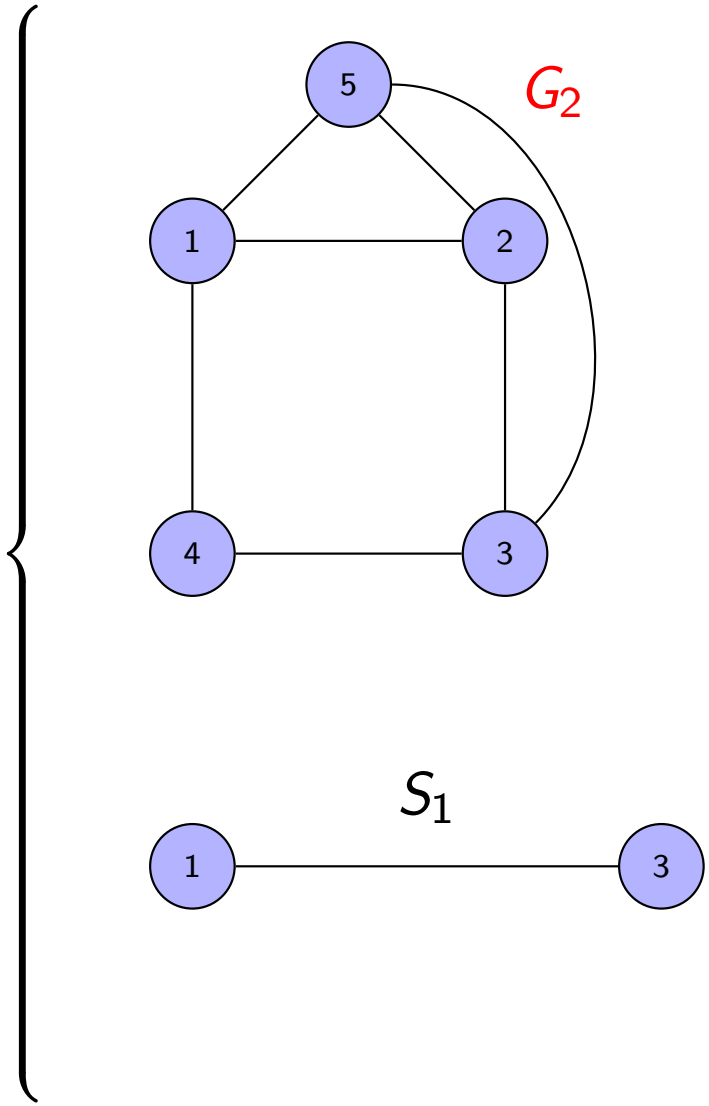
G



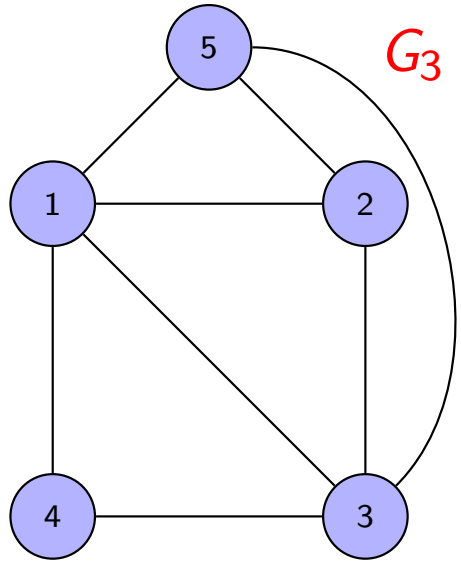


DMP Planarity Algorithm

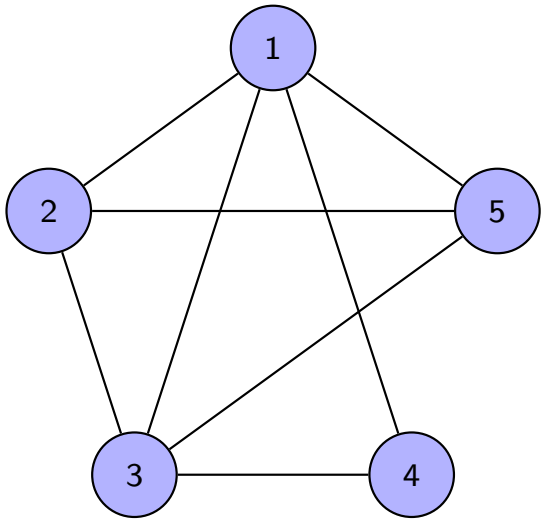


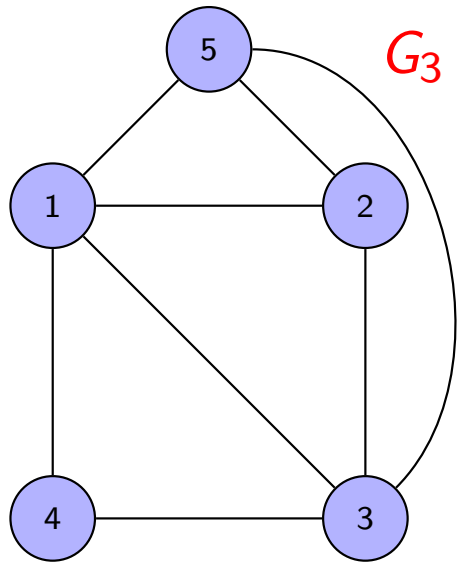


DMP Planarity Algorithm



DMP Planarity Algorithm





DMP Planarity Algorithm

DMP Algorithm

Input: A 2-connected graph G
Output: A planar embedding or FALSE
G_0 := any cycle in G
while $G_j \neq G$ **do**
    if *any segment is blocked* **then**
        return FALSE
    end
    if *a segment is forced* **then**
        B := that segment
    end
    else
        B := any segment
    end
    r := region whose boundary contacts B
    P := path between two contact points of B
    G_{j+1} := drawing of P in r
end
return G_j

What about other surfaces?

Sphere?

What about other surfaces?

Sphere? same as plane

Annulus?

What about other surfaces?

Sphere? same as plane

Annulus? same as plane

Torus?

What about other surfaces?

Sphere? same as plane

Annulus? same as plane

Torus? different!