DRAWING NON-PLANAR GRAPHS WITH CROSSING-FREE SUBGRAPHS*

Patrizio Angelini[1], Carla Binucci[2], **Giordano Da Lozzo[1]**,
Walter Didimo[2], Luca Grilli[2], Fabrizio Montecchiani[2],
Maurizio Patrignani[1], and Ioannis G. Tollis[3]

Graph Drawing 2013 - Bordeaux, France

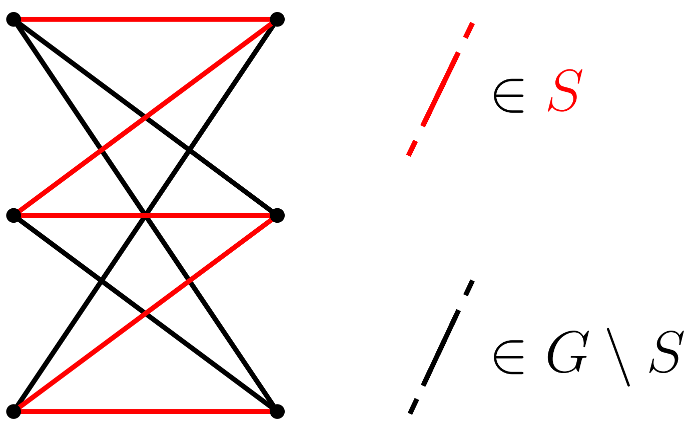
* Work on these results began at the *8th Bertinoro Workshop on Graph Drawing*

THE PROBLEM

Instance: a pair <*G*,*S*> *such that*

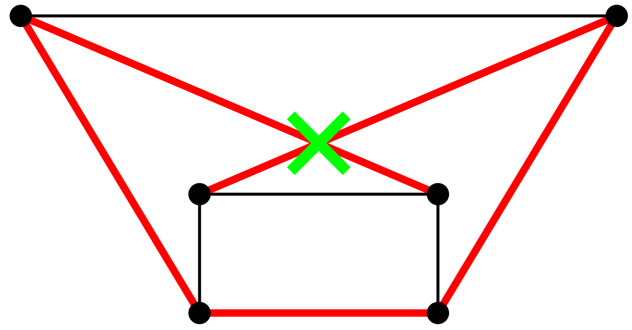
- *G is a* **non-planar** graph
- *S* is a **planar spanning** subgraph of *G*

Question: does there exist a **straight-line drawing** of *G* in which the edges of *S* not involved in any crossing?

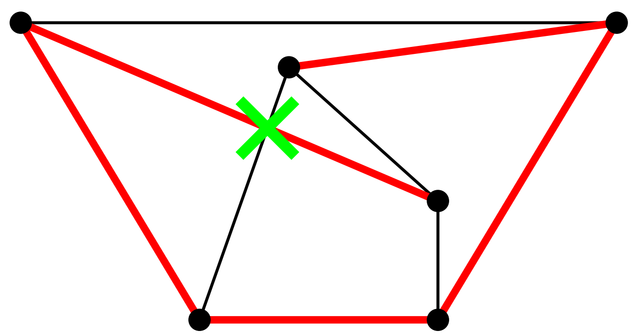


<*G*,*S*>
=
< $K_{3,3}$, **hamiltonian path** >

BAD CROSSINGS

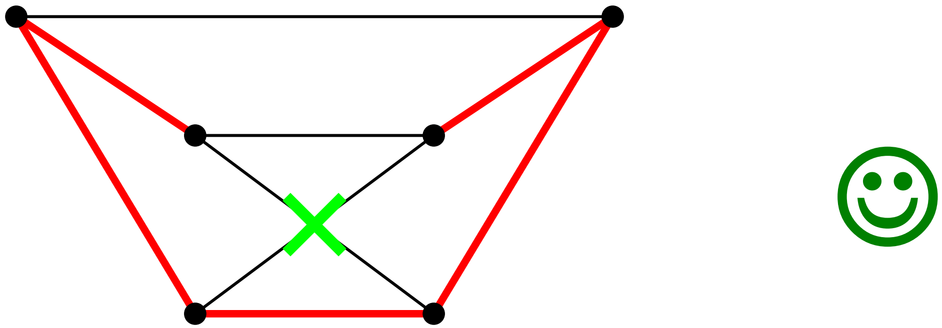


☹



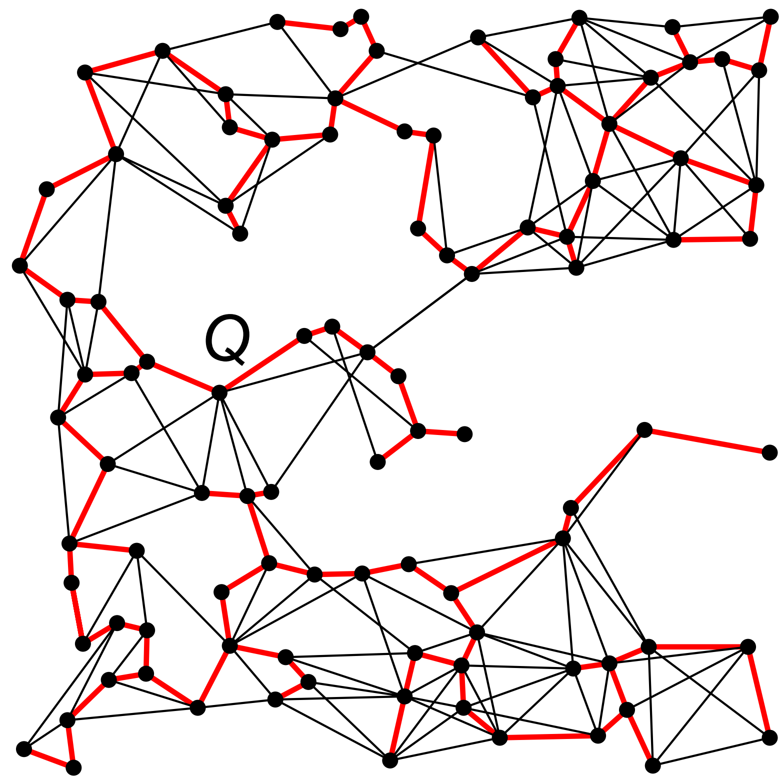
☹

GOOD CROSSINGS



MOTIVATIONS

- different groups of edges may have different **semantics/importance**
- a visual interface might attempt to display more important edges in a **planar way**
- try to maintain the **overview** of the whole graph



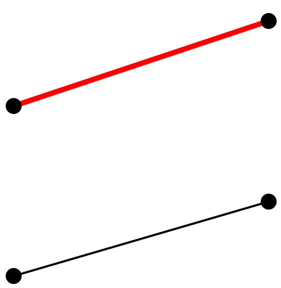
Routing Tree for node *Q*

TWO SETTINGS

Straight-line setting

1. edges of *S and of G\S* are straight-line segments

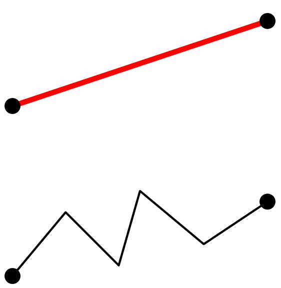
straight-line compatible drawings



Polyline setting

1. edges of *S* are straight-line segments
2. edges of *G\S* have at most k-bends

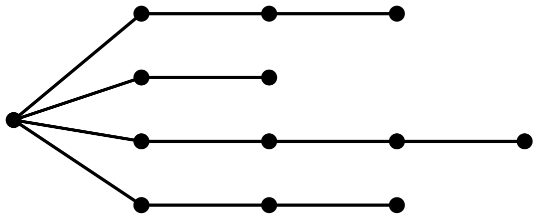
k-bend compatible drawings



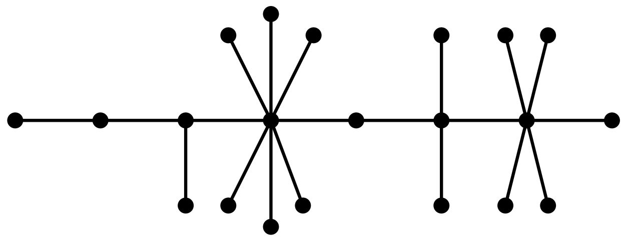
RESULTS: STRAIGHT-LINE SETTING

Always positive instances

1. spiders



2. caterpillars



3. BFS-trees

- every graph admits such planar spanning subgraphs

RESULTS: STRAIGHT-LINE SETTING

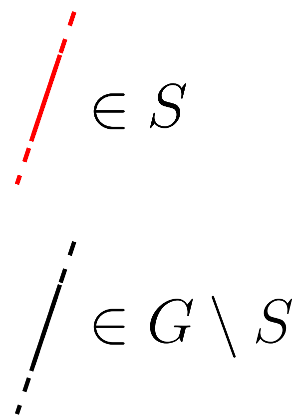
Negative instances

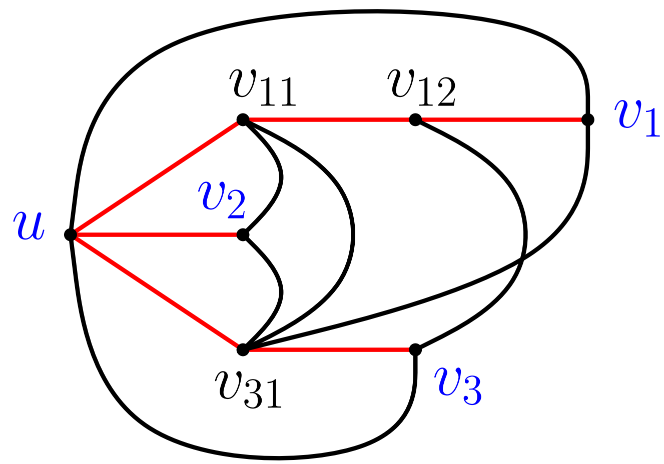
- even if S is a binary tree

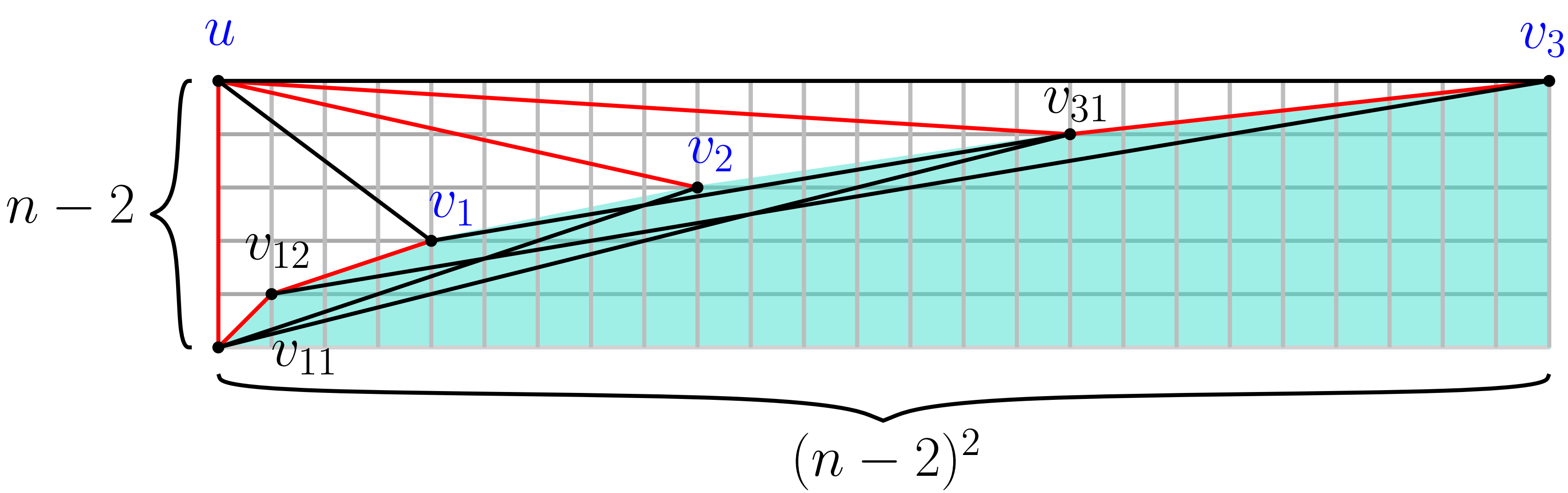
Efficient testing and drawing algorithm

- when S is triconnected

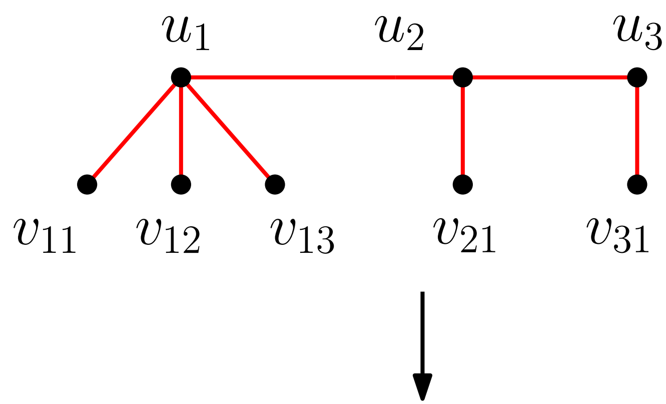
STRAIGHT-LINE: SPIDERS



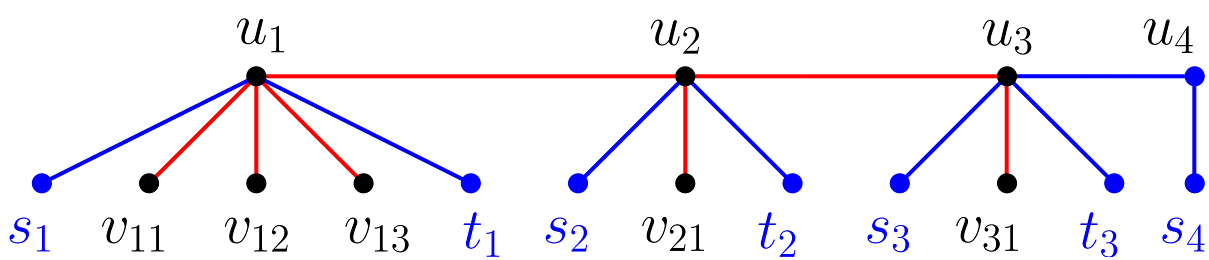




STRAIGHT-LINE: CATERPILLARS



Augmentation of *G* and *S* with dummy vertices and edges



Order *L*

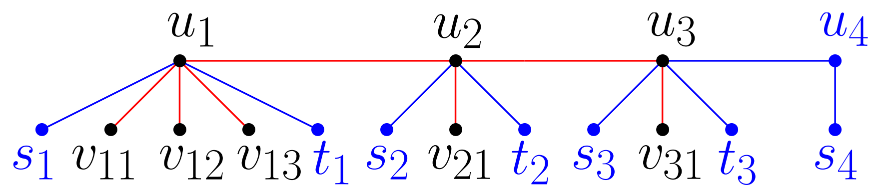
- suitable traversal of the augmented version of *S* started at u_1

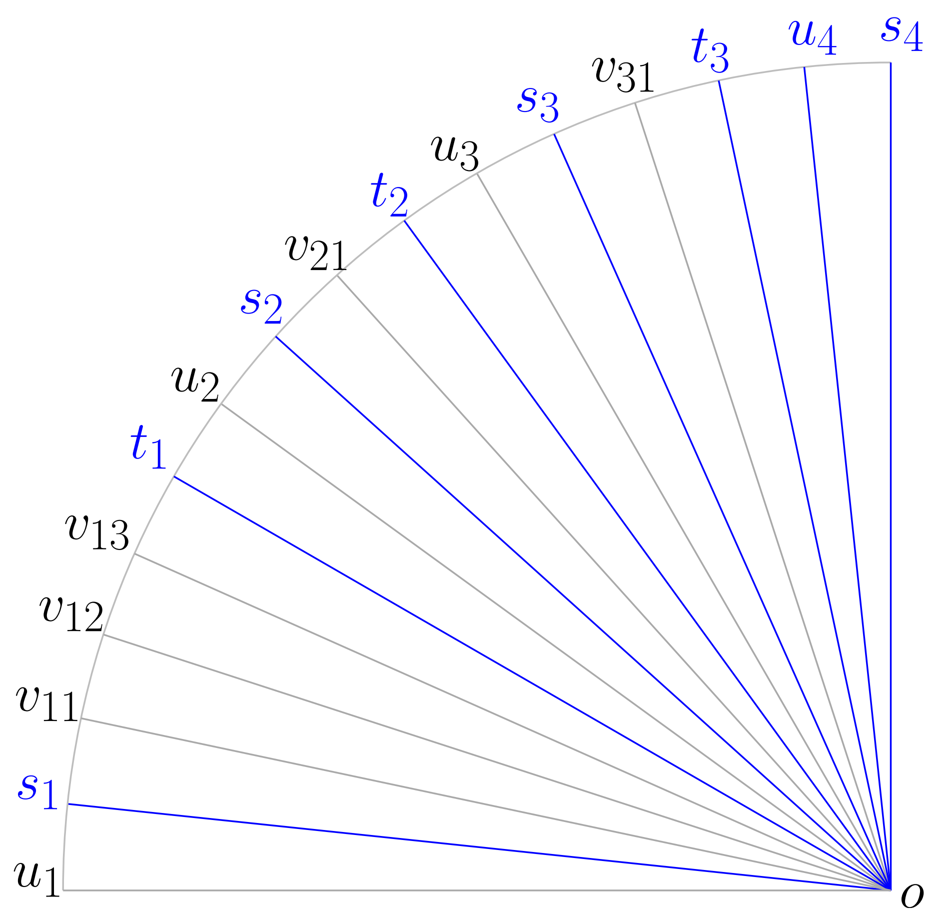
$L = \{u_1, s_1, v_{11}, v_{12}, v_{13}, t_1, u_2, s_2, v_{21}, t_2, u_3, s_3, v_{31}, t_3, u_4, s_4\}$

STRAIGHT-LINE: CATERPILLARS

CONSTRUCTION:

- consider a quarter of circumference *C*
- split *C* in *|L|-1* equally spaced sectors
- each vertex is suitably drawn along the rays delimiting each sector

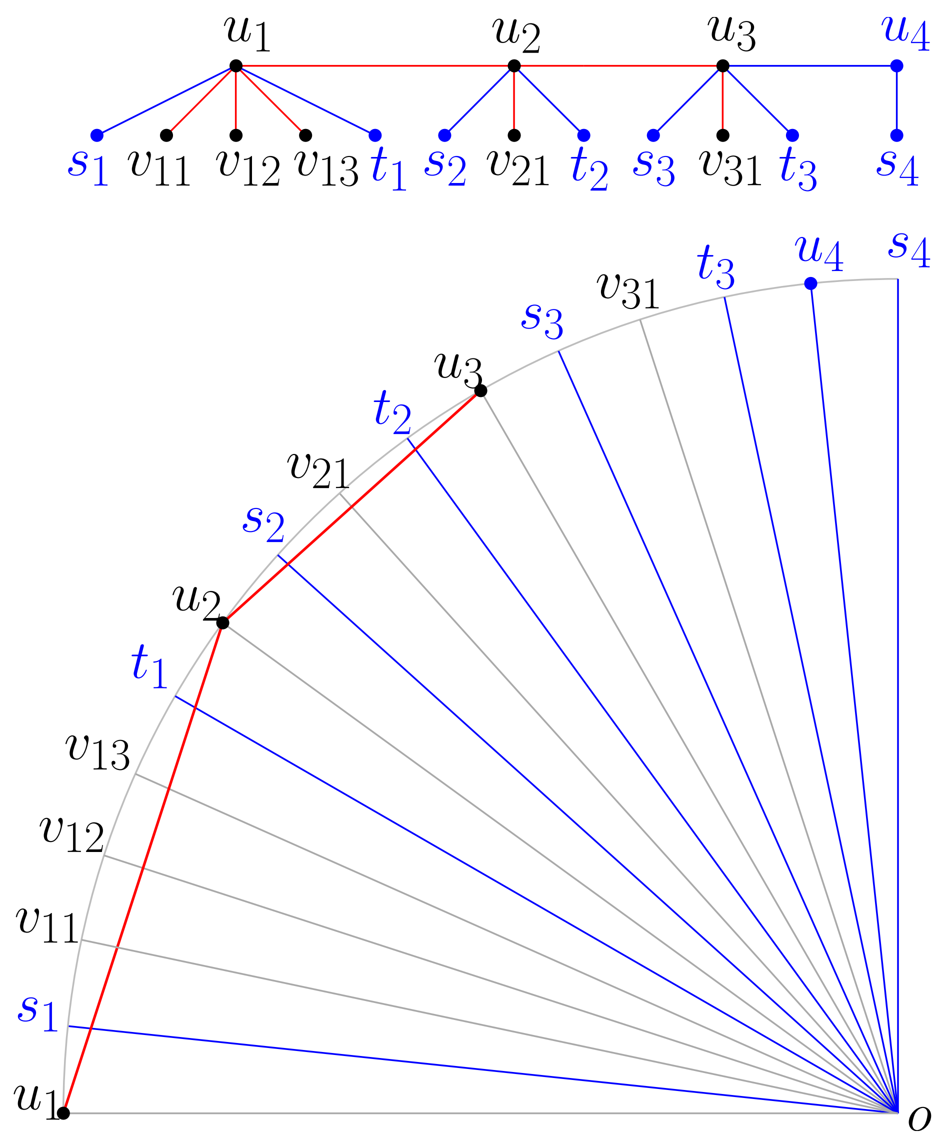




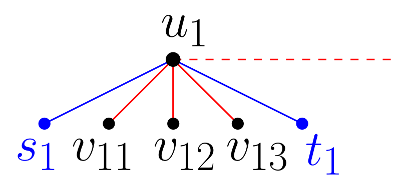
STRAIGHT-LINE: CATERPILLARS

CONSTRUCTION:

- vertices of the spine

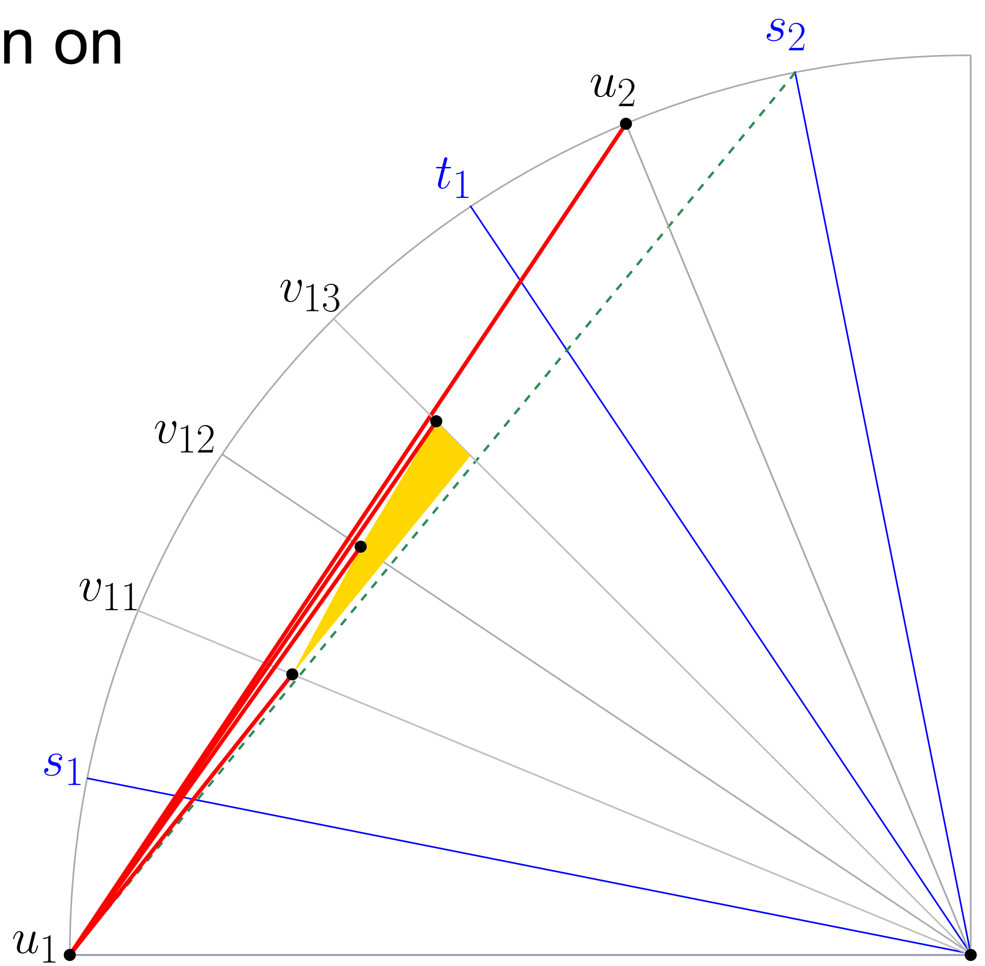


STRAIGHT-LINE: CATERPILLARS

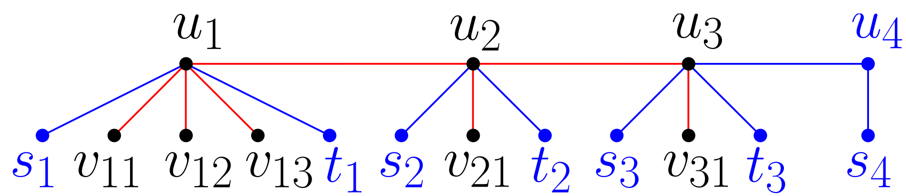


CONSTRUCTION:

- leaf vertices are drawn on a convex curve

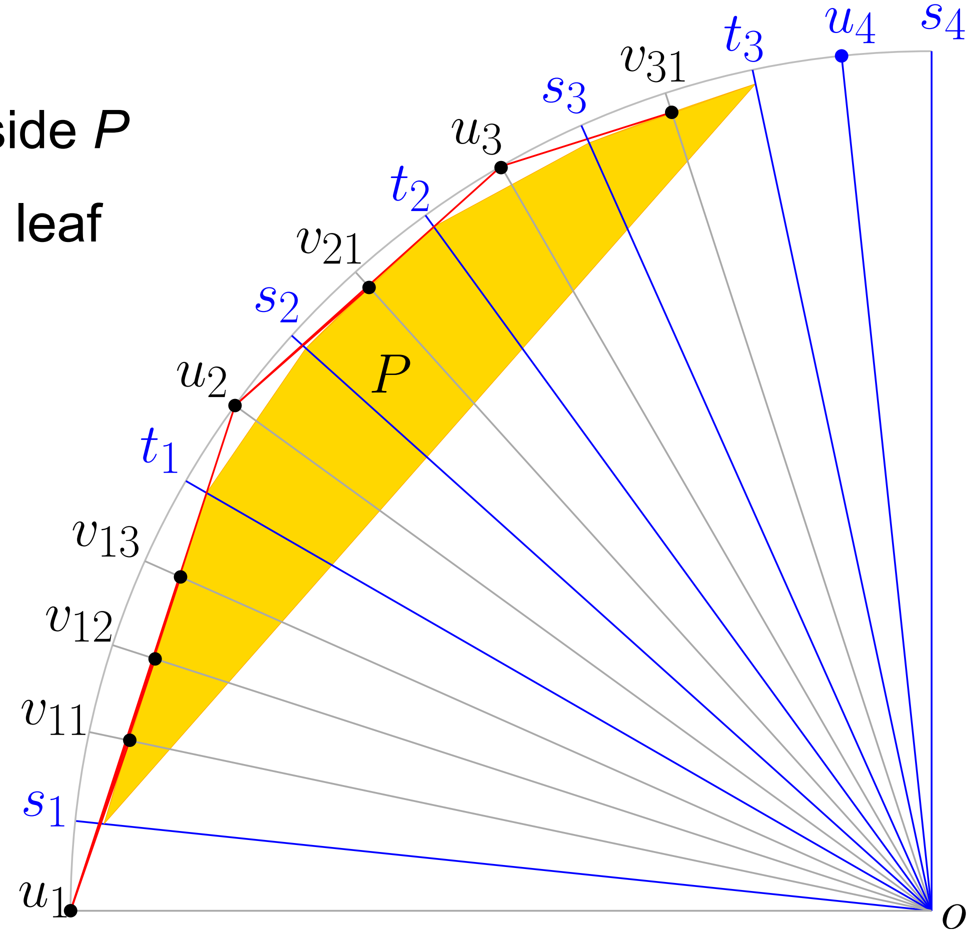


STRAIGHT-LINE: CATERPILLARS



S is crossing free:

1. *S* is drawn planar
2. edges between leaves inside *P*
3. edges between spine and leaf vertices do not cross *S*

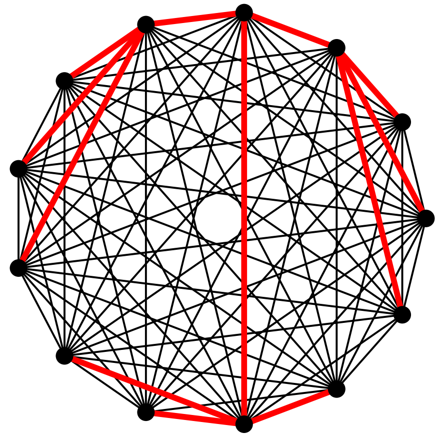
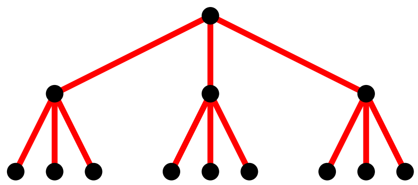


Nice property:

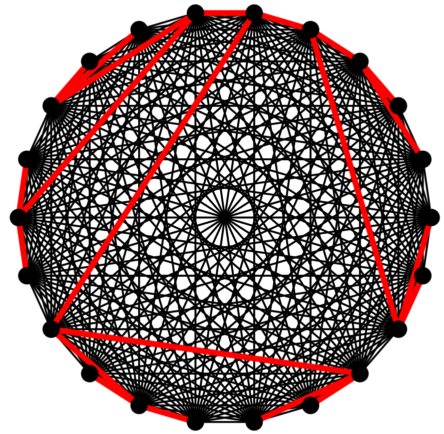
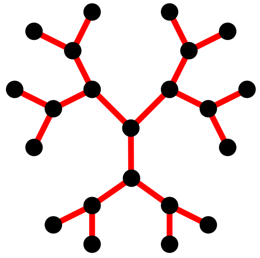
$d_{min}/d_{max} = \Omega(n^{-1})$

RESULTS: NEGATIVE INSTANCES

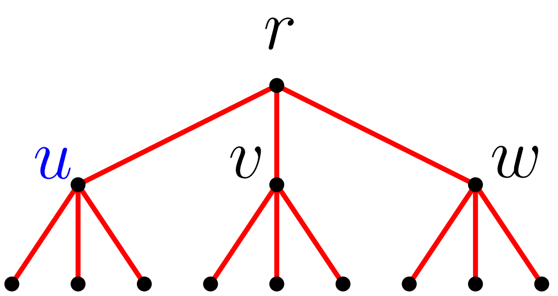
1. $G = K_{13}$ and S is the **complete rooted ternary tree**

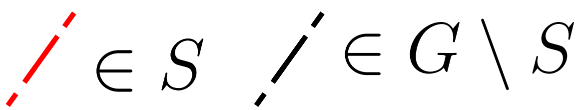



2. $G = K_{22}$ and S is the **complete unrooted binary tree**

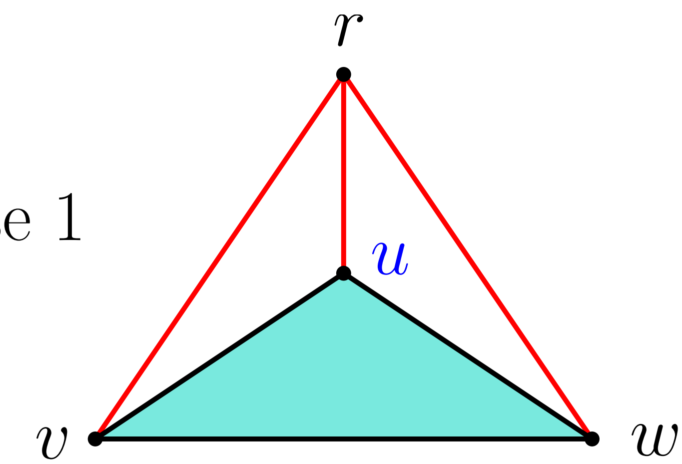



RESULTS: NEGATIVE INSTANCES

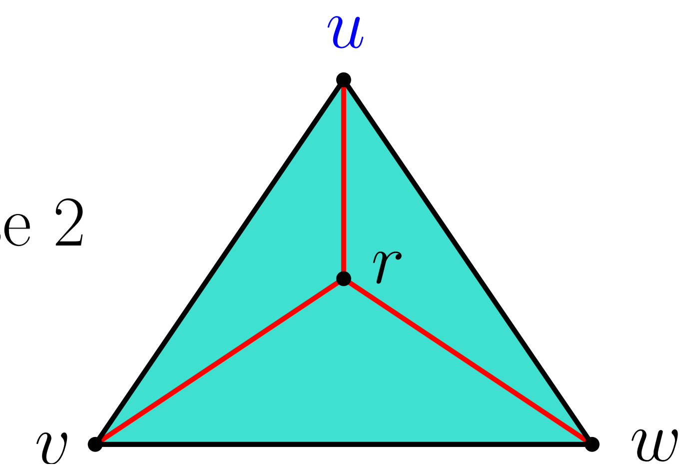




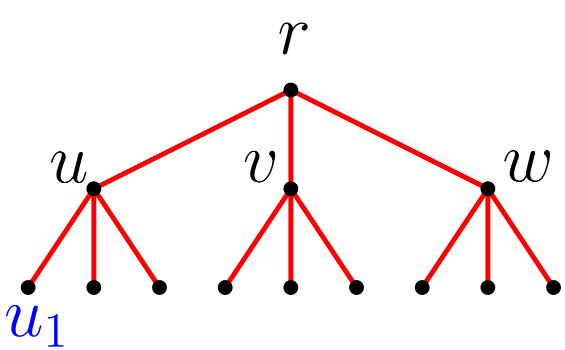
Case 1



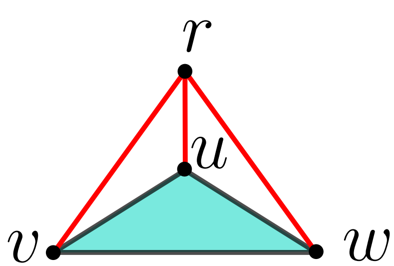
Case 2



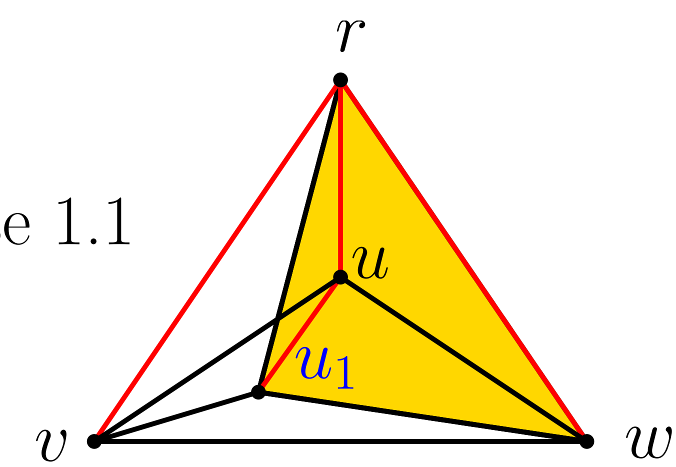
RESULTS: NEGATIVE INSTANCES



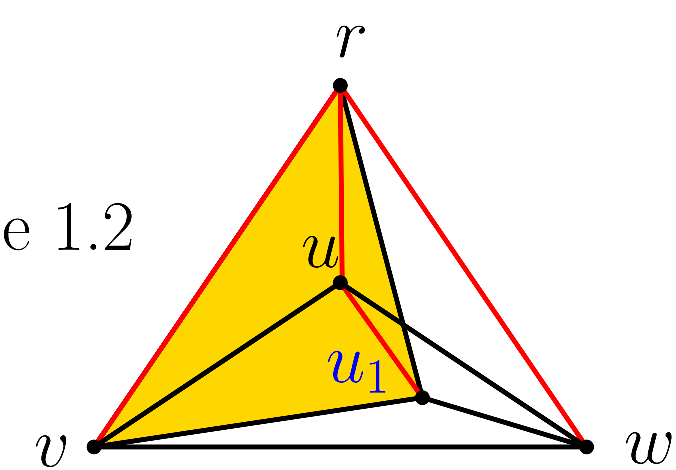
Case 1



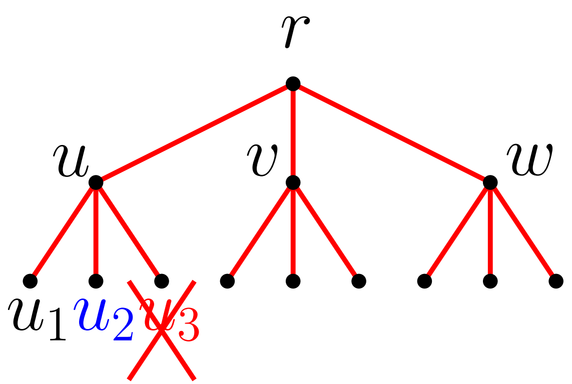
Case 1.1



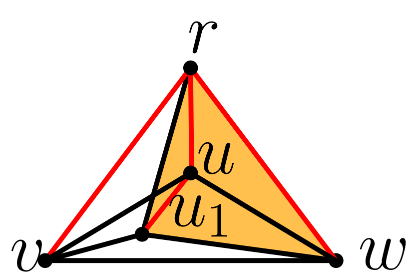
Case 1.2

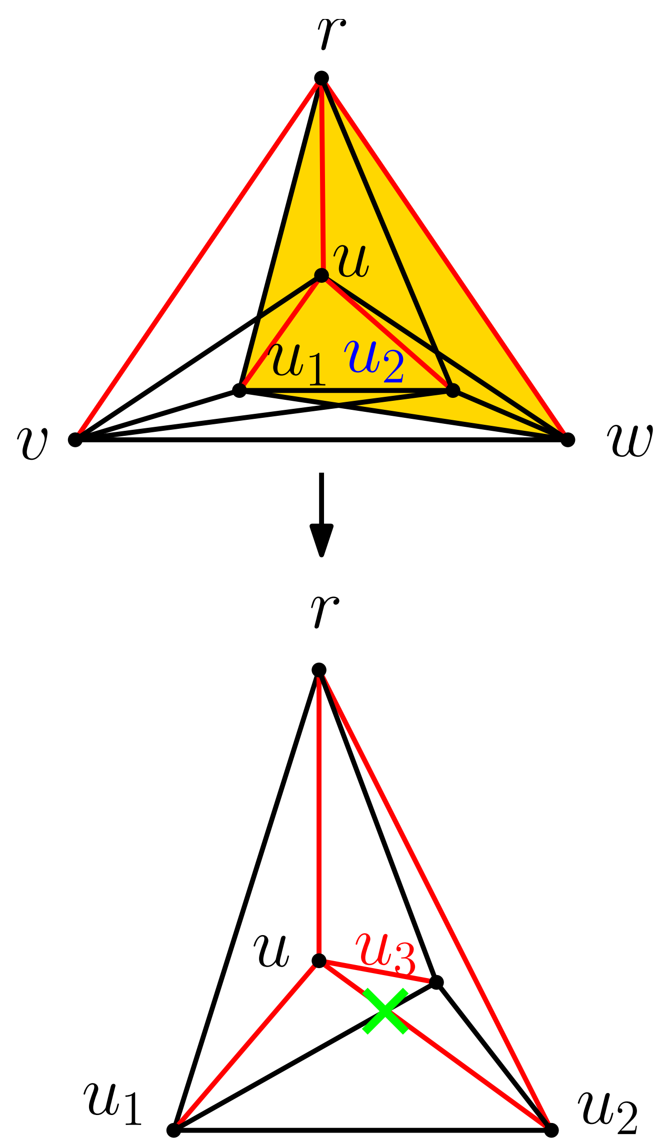


RESULTS: NEGATIVE INSTANCES

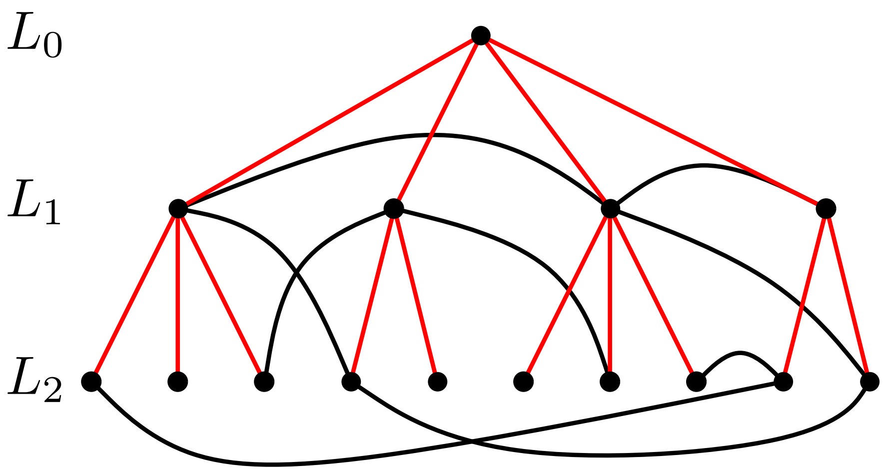


Case 1.1





RESULTS: BFS-TREES



KEY PROPERTY:
edges of $G \setminus S$ connect vertices belonging to the **same level** or to **consecutive levels**

- **construction similar to the caterpillar case**

☺

BFS-trees can be constructed in linear time

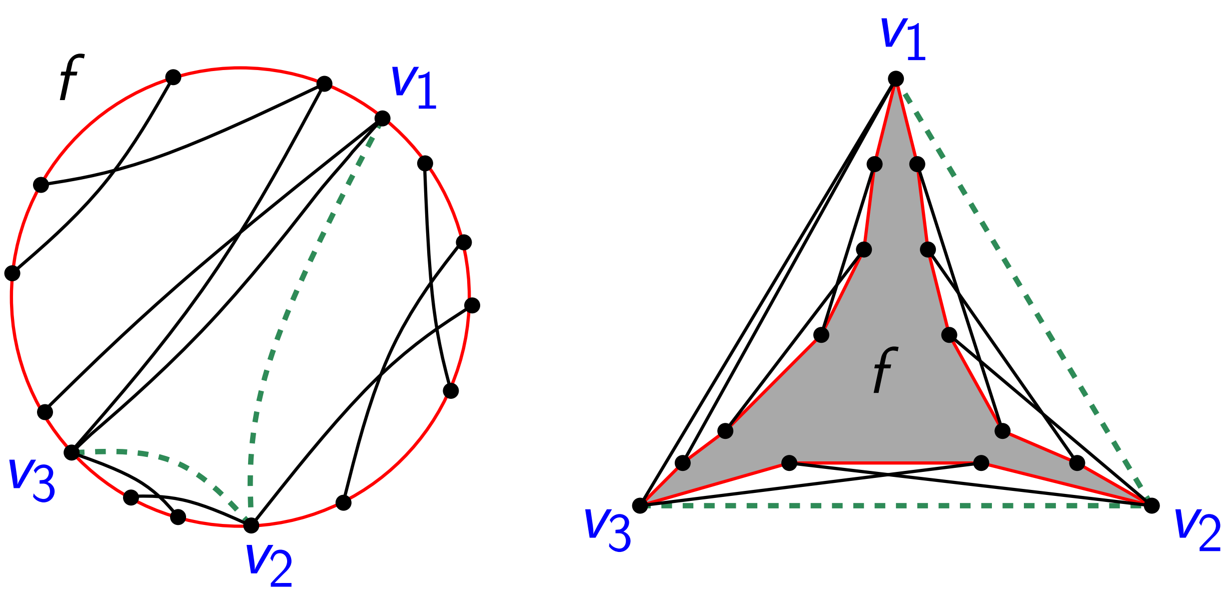
☹

the drawing may require $\Omega(2^n)$-area

STRAIGHT-LINE: TRICONNECTED GRAPHS

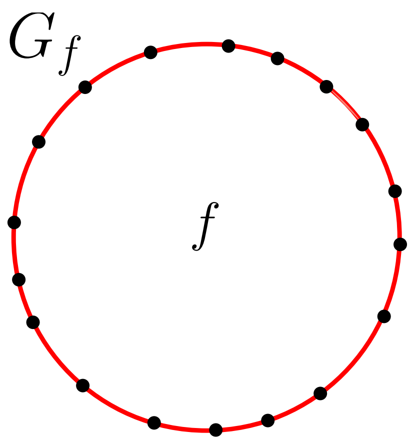
Necessary and sufficient conditions:

1. each **edge *e*** of ***G\S*** connects vertices belonging to the same face of *S*
2. there exists **a face *f* of *S*** containing **3 vertices** that do not separate the endpoints of any edge of ***G\S***



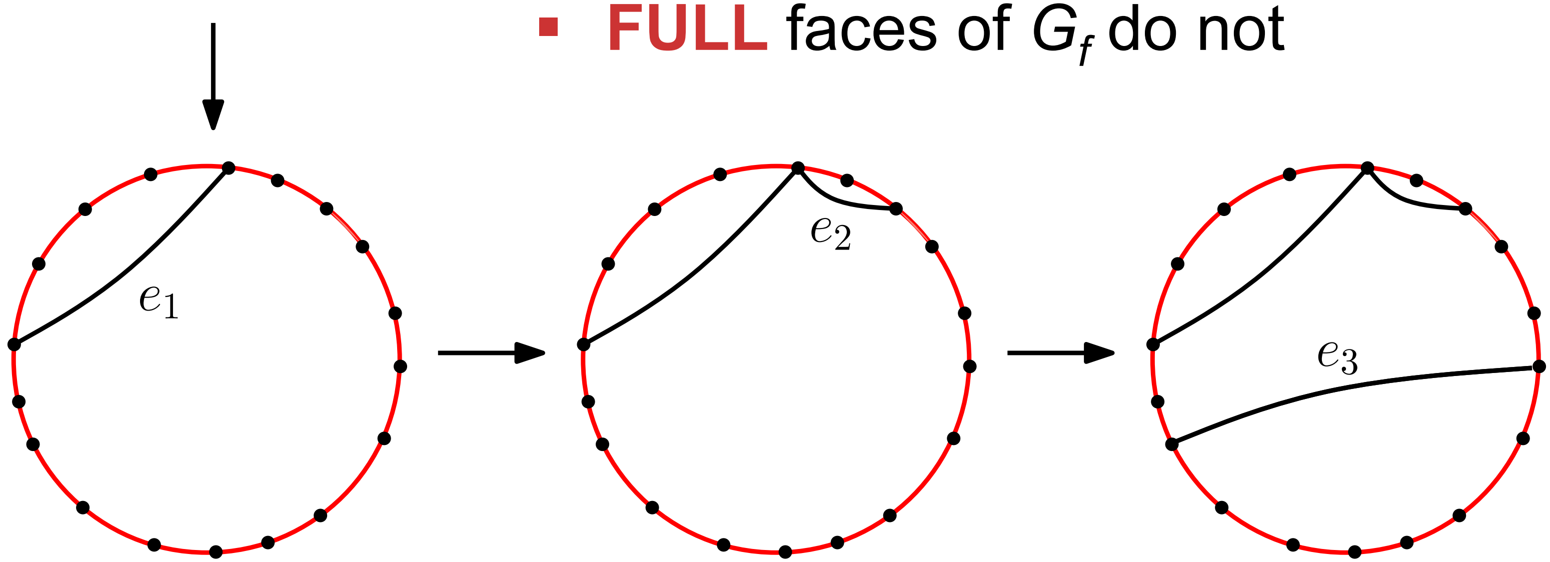
STRAIGHT-LINE: TRICONNECTED GRAPHS

- **Testing Condition 2:**



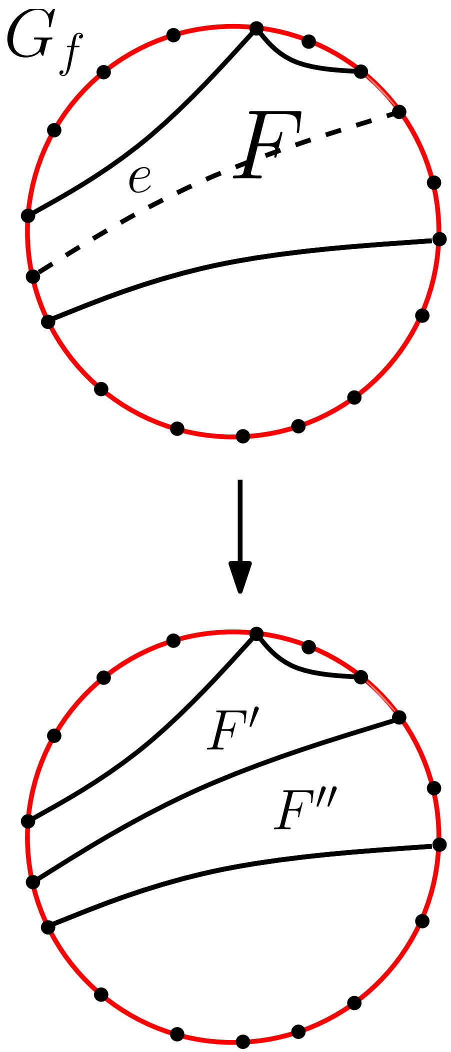
Auxiliary labelled biconnected outerplane graph G_f for each face f of S

- **EMPTY** faces of G_f contain vertices satisfying Cond 2
- **FULL** faces of G_f do not



STRAIGHT-LINE: TRICONNECTED GRAPHS

- **Testing Condition 2:**

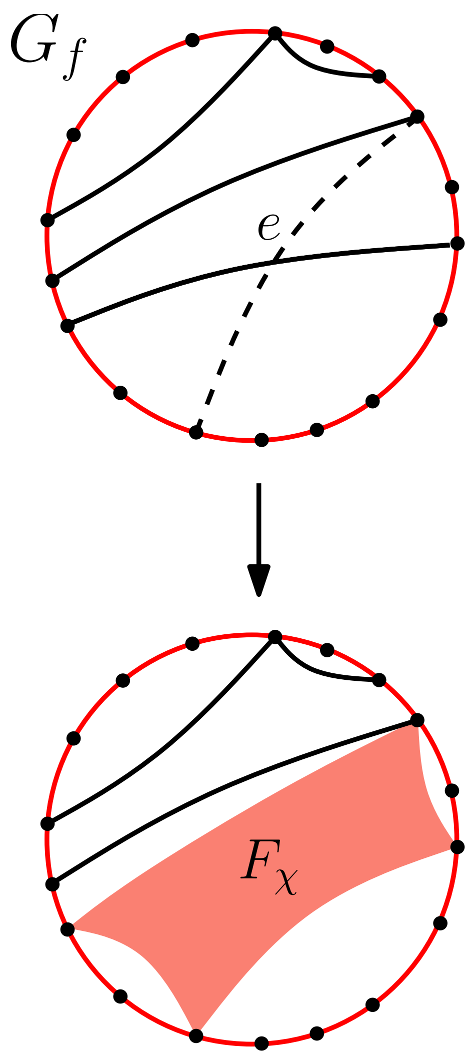


CASE 1: edge *e* **splits** a single EMPTY face *F* of G_f

UPDATE: replace *F* with **2 EMPTY** faces *F'* and *F''*

STRAIGHT-LINE: TRICONNECTED GRAPHS

- **Testing Condition 2:**



CASE 2: edge *e* **crosses** a set of inner edges of G_f

UPDATE: form a new **FULL** face F_x (and remove the previous EMPTY and FULL faces traversed)

G = (V,E) **S=(V,W)**

COND2: ***O*(|*E**W*|×|*V*|)-time**

RESULTS: POLYLINE SETTING

- grid K-bend compatible drawings of **Trees**

  - 1-bend

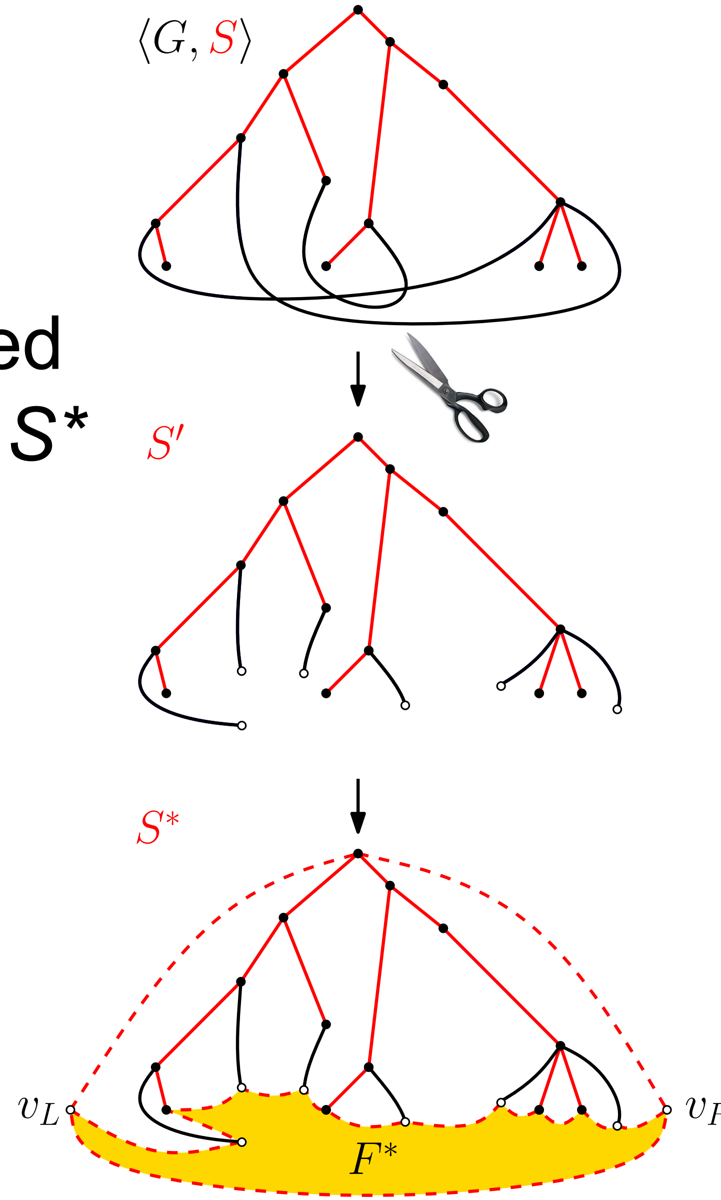
  - 3-bend

  - RAC 4-bend

right angle crossings

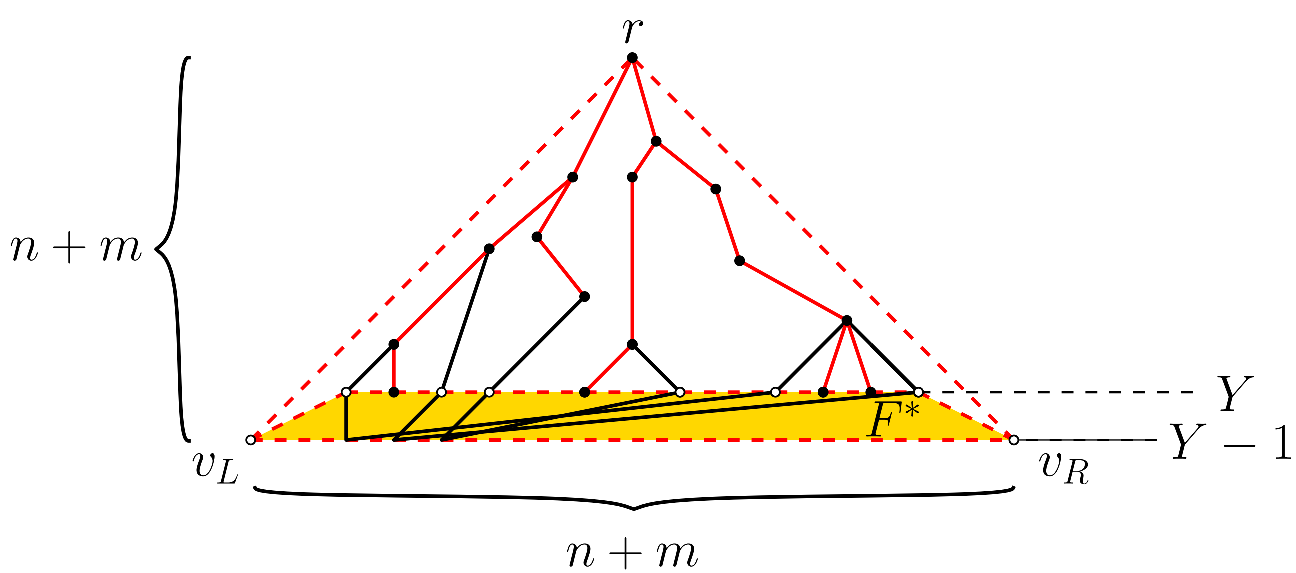
POLYLINE: 3-BEND TREES

- augment S to an embedded biconnected planar graph S^*
- dummy vertices belong to the same face F^*



POLYLINE: 3-BEND TREES

- **straight-line grid drawing** of *S** [Kant'96]
  - the leaf vertices of *S** have the same y-coordinate *Y*
- bend points for the edges *of G\S* have either y-coordinate *Y* or *Y-1*



OPEN PROBLEMS FOR FUTURE WORK

❑ STRAIGHT-LINE SETTING

- What is the **complexity** of the problem when *S* is not a spanning triconnected graph?

- Give a **characterization** of which spanning trees of a complete graph can be always realized.

❑ POLYLINE SETTING

- What is the optimal area requirement for **grid k-bend** compatible drawings when *S* is a **spanning trees**?

THANK YOU!