Polynomial kernel for face cover on non-embedded planar graphs

Thekla Hamm, Sukanya Pandey, Krisztina Szilágyi

Co-funded by the European Union

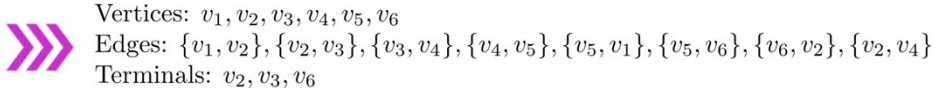
MINISTRY OF EDUCATION, YOUTH AND SPORTS

ROBOPROX

Part 1: Face cover

Face cover

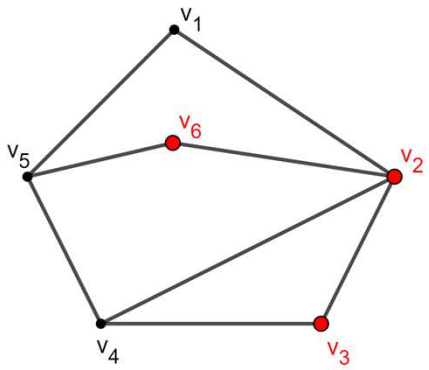
- Input: planar graph $G = (V, E)$, set of terminals $T \subseteq V$
- Output: size of smallest face cover – set of faces covering T



Face cover

- Input: planar graph $G = (V, E)$, set of terminals $T \subseteq V$
- Output: size of smallest face cover – set of faces covering T

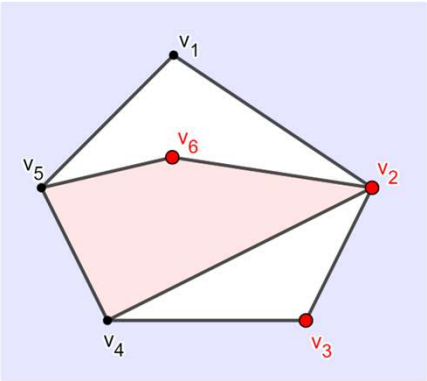
Vertices: $v_1, v_2, v_3, v_4, v_5, v_6$
Edges: $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_5\}, \{v_5, v_1\}, \{v_5, v_6\}, \{v_6, v_2\}, \{v_2, v_4\}$
Terminals: v_2, v_3, v_6



Face cover

- Input: planar graph $G = (V, E)$, set of terminals $T \subseteq V$
- Output: size of smallest face cover – set of faces covering T

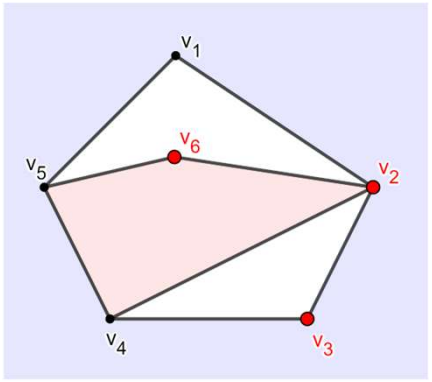
Vertices: $v_1, v_2, v_3, v_4, v_5, v_6$
Edges: $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_5\}, \{v_5, v_1\}, \{v_5, v_6\}, \{v_6, v_2\}, \{v_2, v_4\}$
Terminals: v_2, v_3, v_6

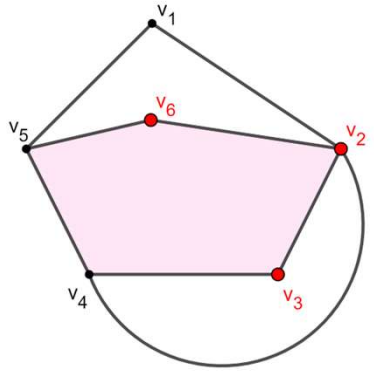


Face cover

- Input: planar graph $G = (V, E)$, set of terminals $T \subseteq V$
- Output: size of smallest face cover – set of faces covering T

Vertices: $v_1, v_2, v_3, v_4, v_5, v_6$
Edges: $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_5\}, \{v_5, v_1\}, \{v_5, v_6\}, \{v_6, v_2\}, \{v_2, v_4\}$
Terminals: v_2, v_3, v_6





Previous results

- Motivation: parameterization by face cover number
  - Steiner tree, Multiway cut, Shortest paths…

- k= size of face cover
- $O(c^k \cdot n)$ algorithm [BM'88]
- Restricted version – fixed embedding:
  - $O\left(2^{10.1\sqrt{k}} + n^2\right)$ [KT'10]
  - Linear kernel [GST'17]
- WLOG assume that the input graph is biconnected

Part 2: SPQR-trees

SPQR-trees

Background

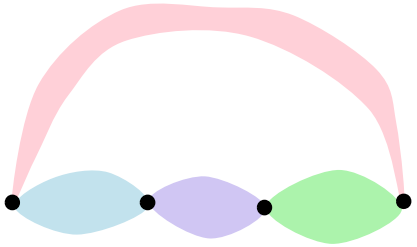
- 3-connected graph = need to delete at least 3 vertices to disconnect it
- Theorem [Whitney ‘33]: A 3-connected planar graph has a unique (combinatorial) embedding

SPQR trees

- Describing all embeddings of (biconnected) planar graphs
- Building blocks: 3-connected components and single edges

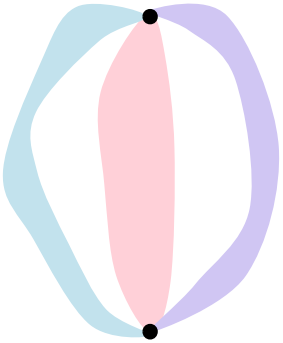
+ connections between them

Connections



Series

S

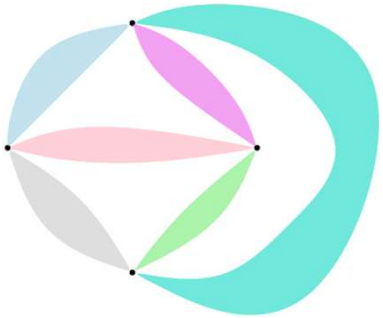


Parallel

P

(Boring)

Q

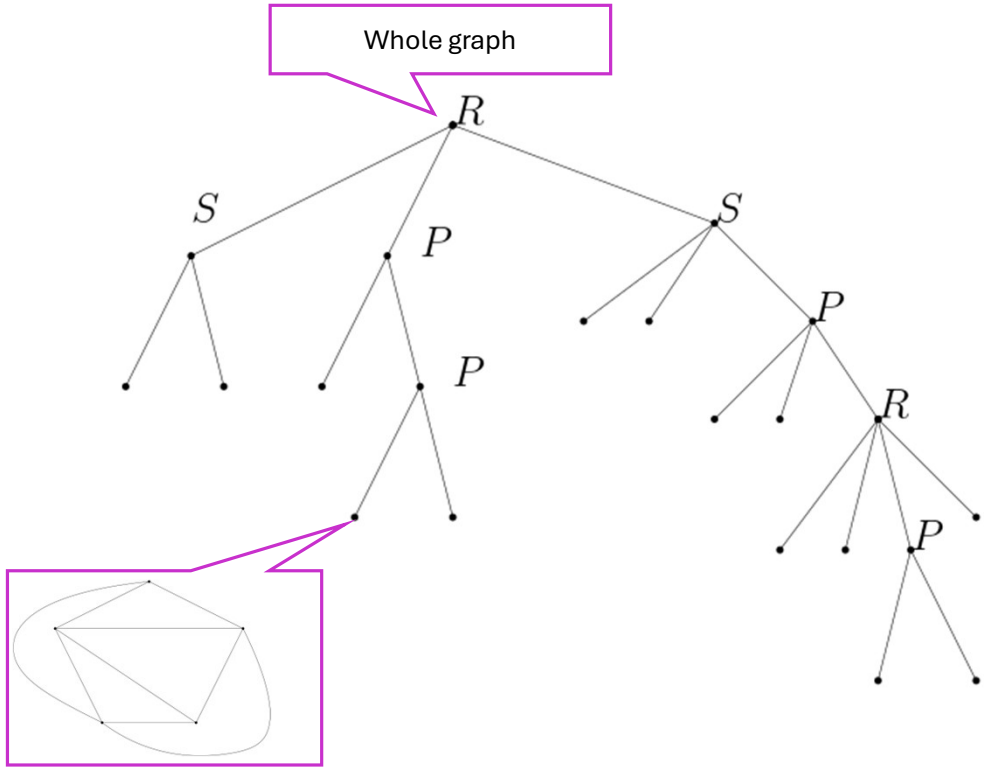


Rigid

R

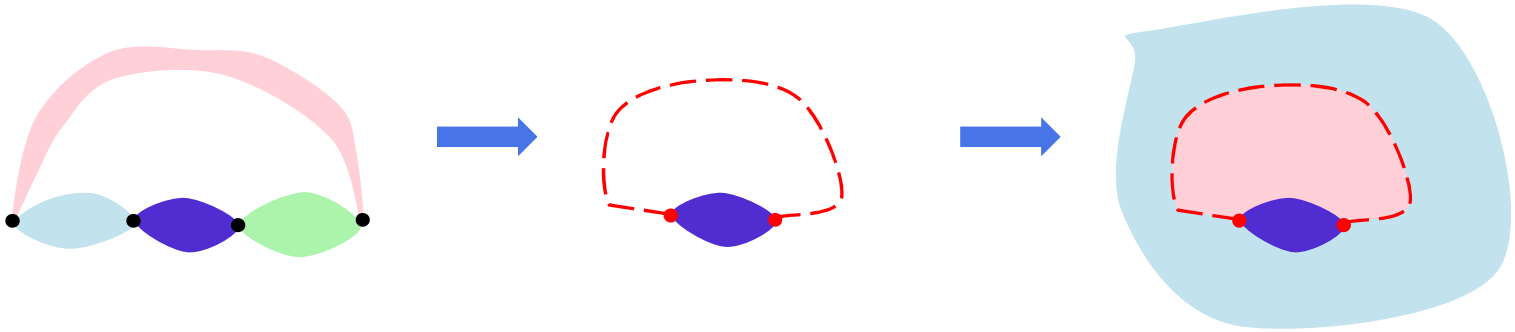
SPQR tree

- Vertices: nodes
- Leaves: 3-connected or edges
- Graph associated to each node



Vocabulary

Node skeleton



Corner vertices,
corner edge

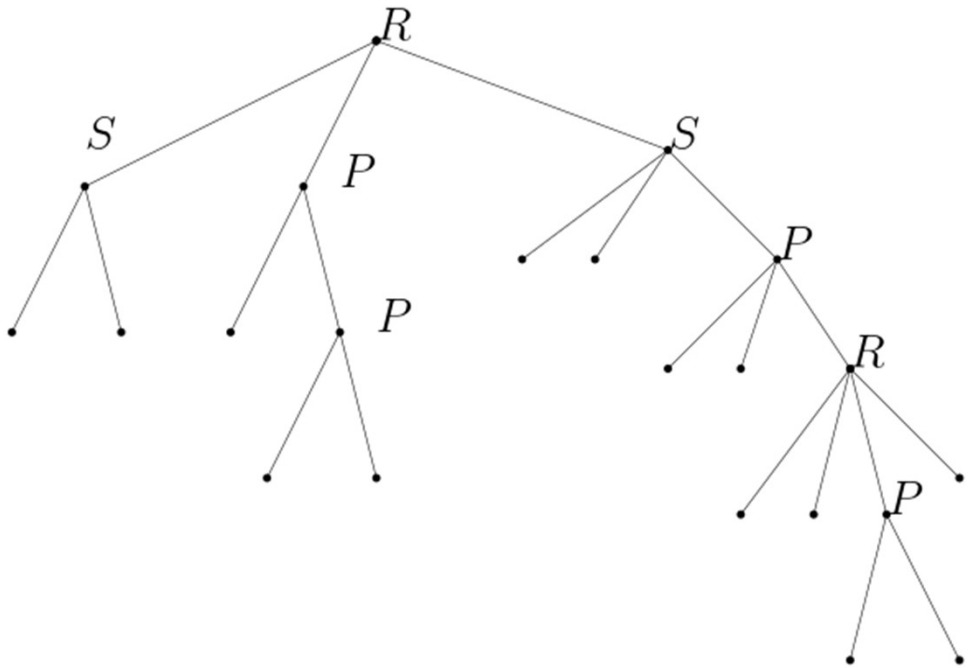
External faces

Part 3: Kernelization

Kernelization

- Create a smaller instance without losing anything important
  - Want: smaller graph with the same face cover
  - k = size of smallest face cover
  - Small = polynomial in k

Step 0:
construct
SPQR tree

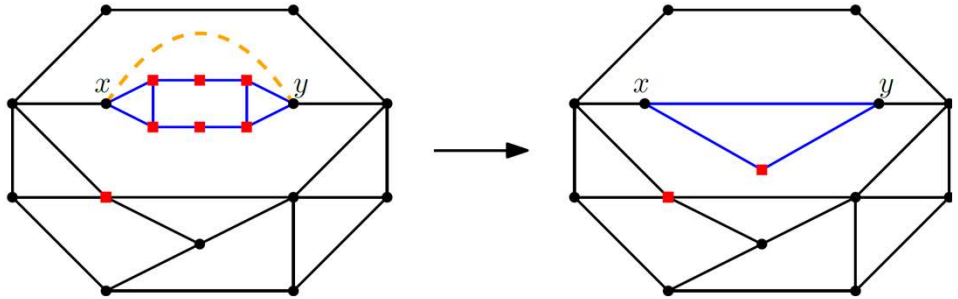


First attempt

- Replace subgraph by a smaller one with the same face cover number

First attempt

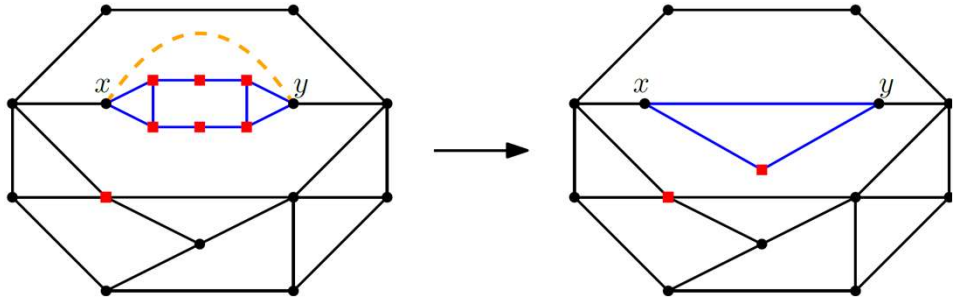
- Replace subgraph by a smaller one with the same face cover number
- Face cover number drops from 2 to 1



First attempt

- Replace subgraph by a smaller one with the same face cover number
- Face cover number drops from 2 to 1
- Distinguish between internal (private) and external faces

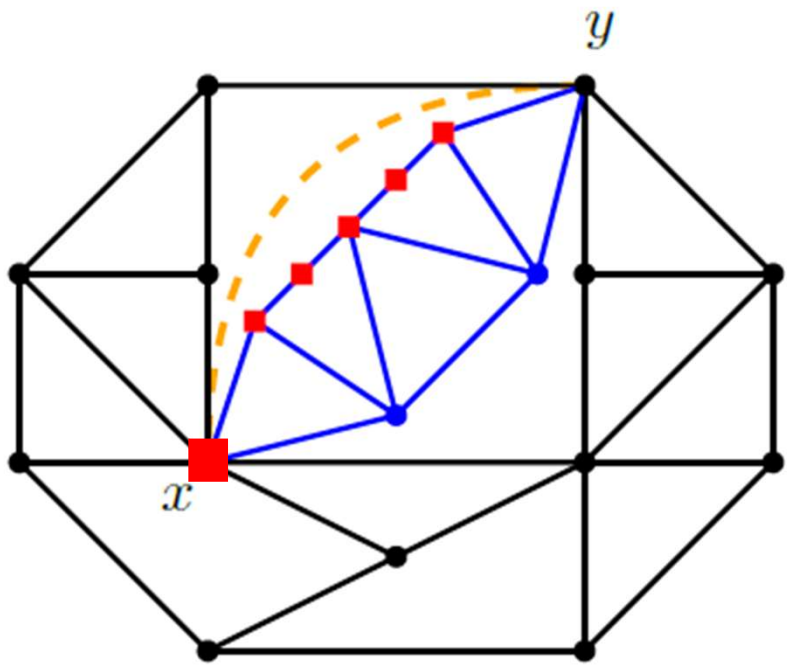
⇒Face cover of the replacement graph using 0/1/2 external faces has to be the same as the original



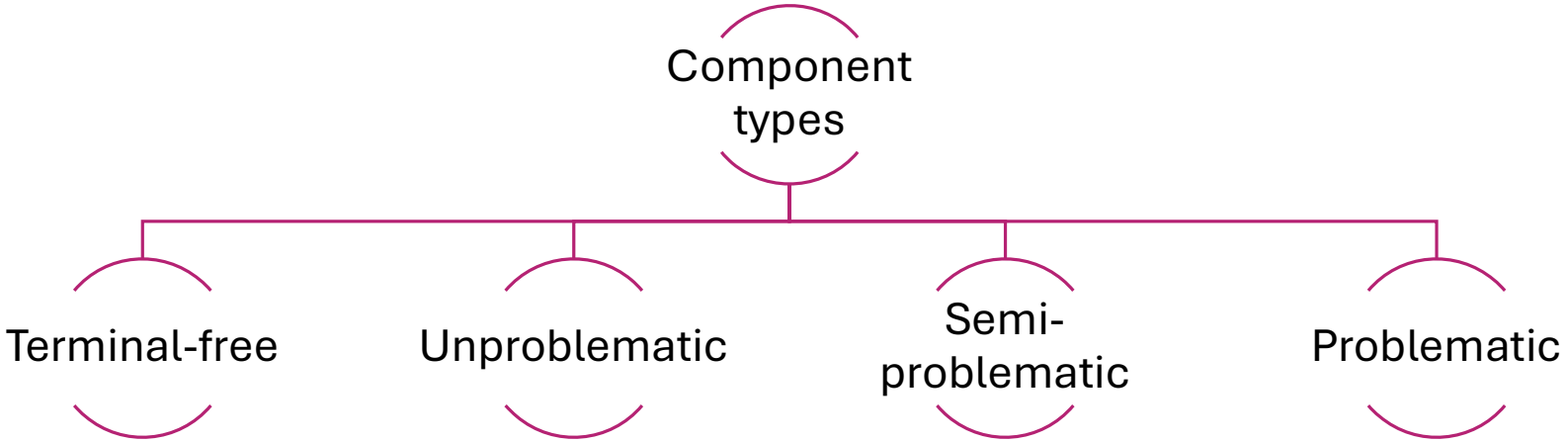
Second attempt

- We might want to cover the corner vertices from “outside”

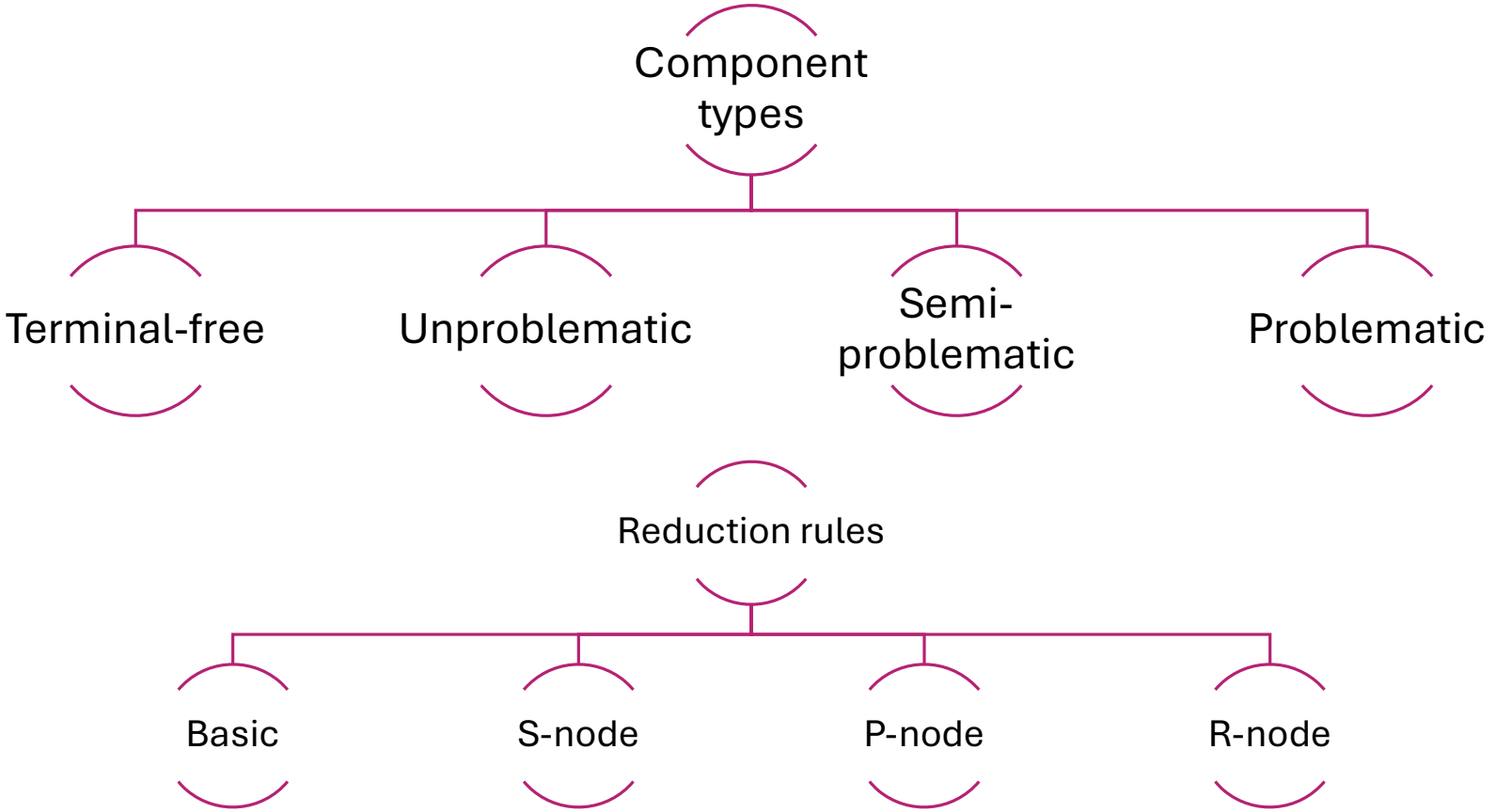
⇒ Compare face covers where corner vertices are not covered



Our approach

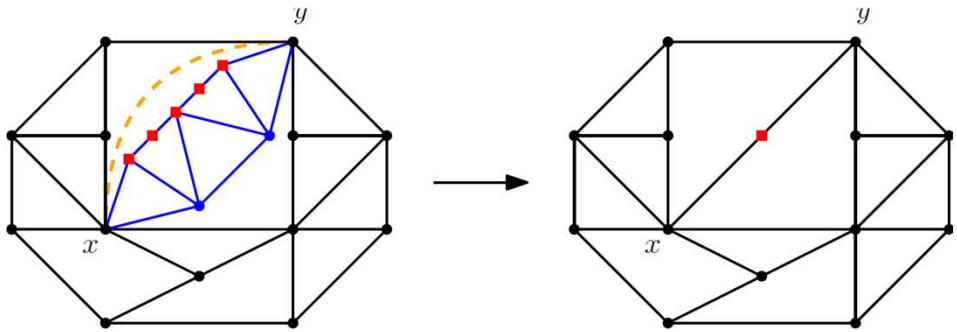


Our approach



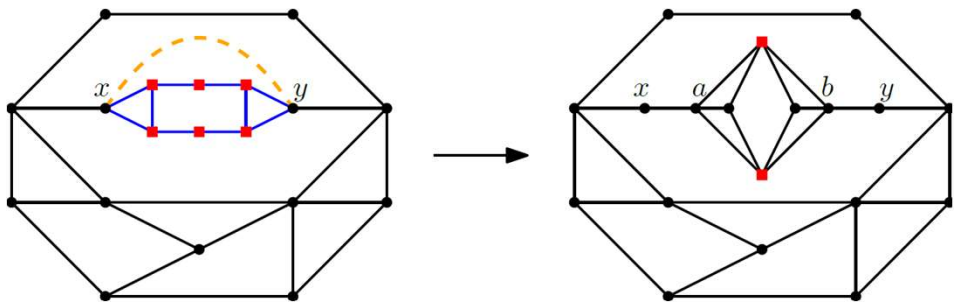
Terminal-free and unproblematic components

- Terminal-free: no terminals except possibly corner vertices
  - Replace by an edge
- Unproblematic: can be covered by one external face
  - Replace by terminal subdivided edge



Semi-problematic and problematic components

- Semi-problematic: can be covered by two external or one internal face
  - Replace by gadget below
- Problematic: everything else



Reduction Rules – S-node and P-node

- Look at which faces can be reused
- Bound the number of each component type
- Delete/contract all but $O(k)$ components

Reduction Rules – R-node

- Lemma: If a face has $\geq 3k + 1$ terminals, it has to be in the face cover

$\Longrightarrow$ if a face has $> 3k + 1$ terminals, turn all but $3k + 1$ into non-terminals

Reduction Rules – R-node

- Lemma: If a face has $\geq 3k + 1$ terminals, it has to be in the face cover

$\Longrightarrow$ if a face has $> 3k + 1$ terminals, turn all but $3k + 1$ into non-terminals

- Destroy 3-connectivity:
  - Delete edges incident to useless faces
  - Merge faces if they contain the same, one terminal/corner vertex
  - Replace some long paths by a single edge

Reduction Rules – R-node

- Lemma: If a face has $\geq 3k + 1$ terminals, it has to be in the face cover

$\Longrightarrow$ if a face has $> 3k + 1$ terminals, turn all but $3k + 1$ into non-terminals

- Destroy 3-connectivity:
  - Delete edges incident to useless faces
  - Merge faces if they contain the same, one terminal/corner vertex
  - Replace some long paths by a single edge
- Rigidize the embedding

Conclusion and open questions

- We get a $O(k^3)$ kernel
- Previous results:
  - Special case: fixed embedding and $T = V$ – linear kernel
  - Subexponential algorithm for fixed embedding
- Open:
  - Linear kernel for non-embedded version?
  - Subexponential algorithm for non-embedded version?