On the Queue Number of Planar Graphs

GD2021

Michael A. Bekos[1] Martin Gronemann[2] Chrysanthi N. Raftopoulou[3]

1. University of Tübingen, Germany
2. Osnabrück University,Germany
3. National Technical University of Athens, Greece

Tübingen, 16/09/2021

Definitions

Linear Layout

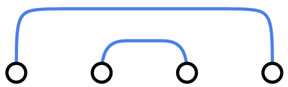
- vertices along the spine
- edges on pages

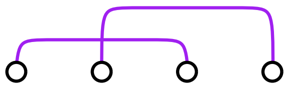
Definitions

Linear Layout

- vertices along the spine
- edges on pages


necklace

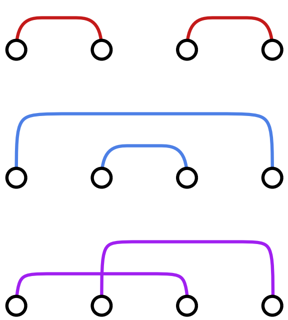

rainbow


twist

Definitions

Linear Layout

- vertices along the spine
- edges on pages

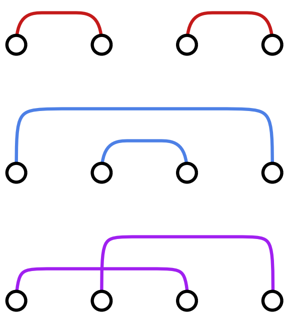


	queue layout
necklace	yes
rainbow	no
twist	yes

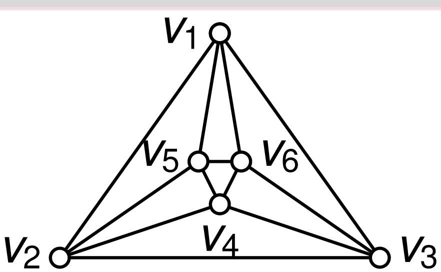
Definitions

Linear Layout

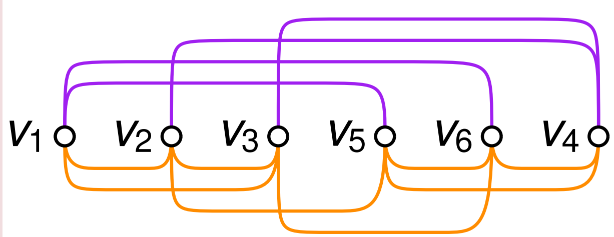
- vertices along the spine
- edges on pages



	queue layout
necklace	yes
rainbow	no
twist	yes



octahedron

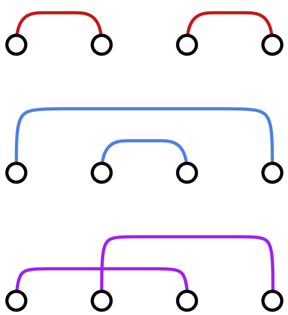


queue layout

Definitions

Linear Layout

- vertices along the spine
- edges on pages



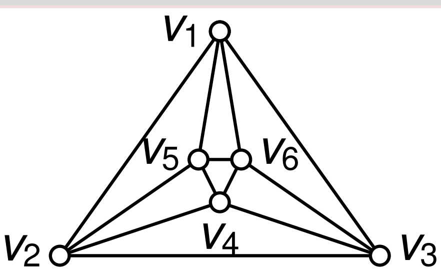
	queue layout
necklace	yes
rainbow	no
twist	yes

Queue number of graph

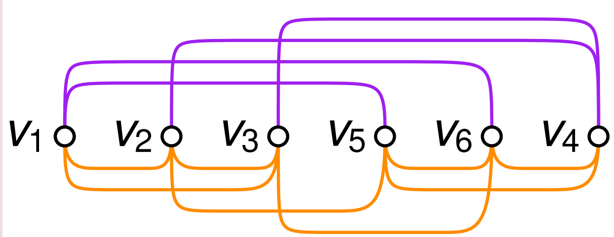
minimum number of pages
over all queue layouts

Queue number of graph family

maximum queue number
over all elements



octahedron



queue layout

On the Queue Number Of Planar Graphs
Michael A. Bekos, Martin Gronemann, Chrysanthi N. Raftopoulou

Problem: Queue number of Planar Graphs

Theorem[1]:

Planar graphs have queue number at most 49.

[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood *Planar graphs have bounded queue-number* (2020)

Problem: Queue number of Planar Graphs

Theorem[1]:

Planar graphs have queue number at most 49.

Improvement:

Planar graphs have queue number at most 42.

[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood *Planar graphs have bounded queue-number* (2020)

Problem: Queue number of Planar Graphs

Theorem[1]:

Planar graphs have queue number at most 49.

Improvement:

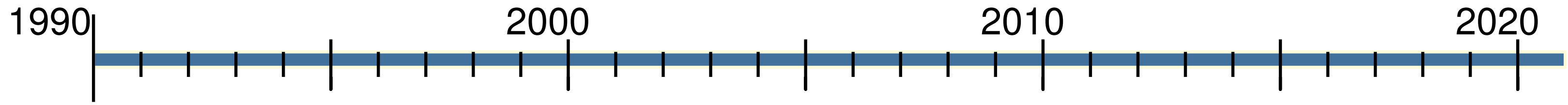
Planar graphs have queue number at most 42.

Idea:

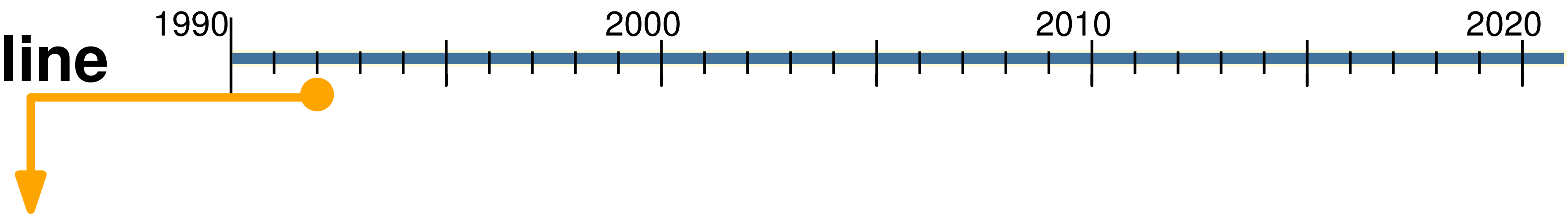
- Exploit planarity
- Modify existing technique

[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood *Planar graphs have bounded queue-number* (2020)

Timeline



Timeline



1992 [1]

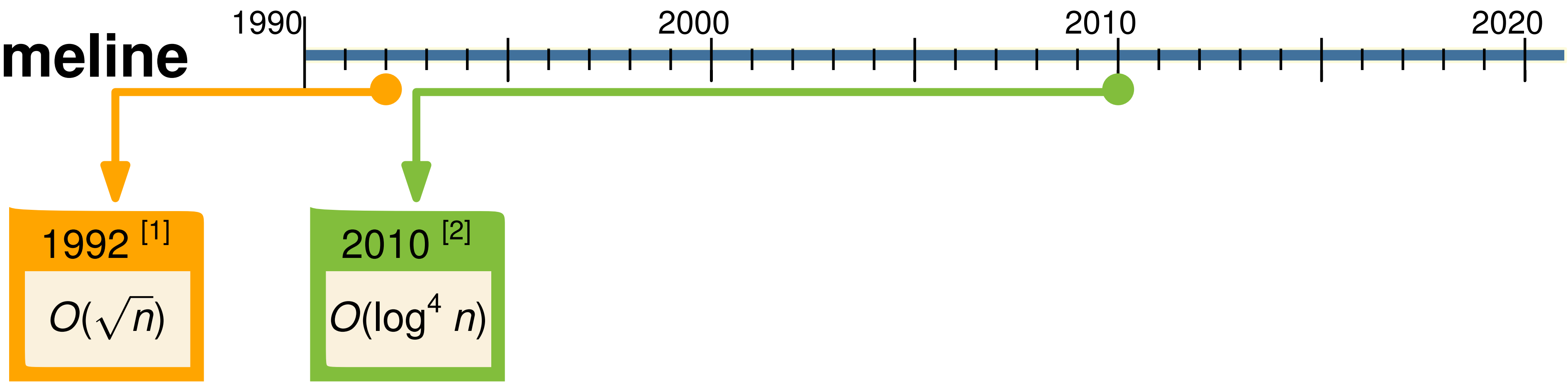
$O(\sqrt{n})$

Conjecture[1]**:**

Planar graphs have bounded queue number.

[1] Heath, Leighton, Rosenberg *Comparing queues and stacks asmechanisms for laying out graphs* (1992)

Timeline



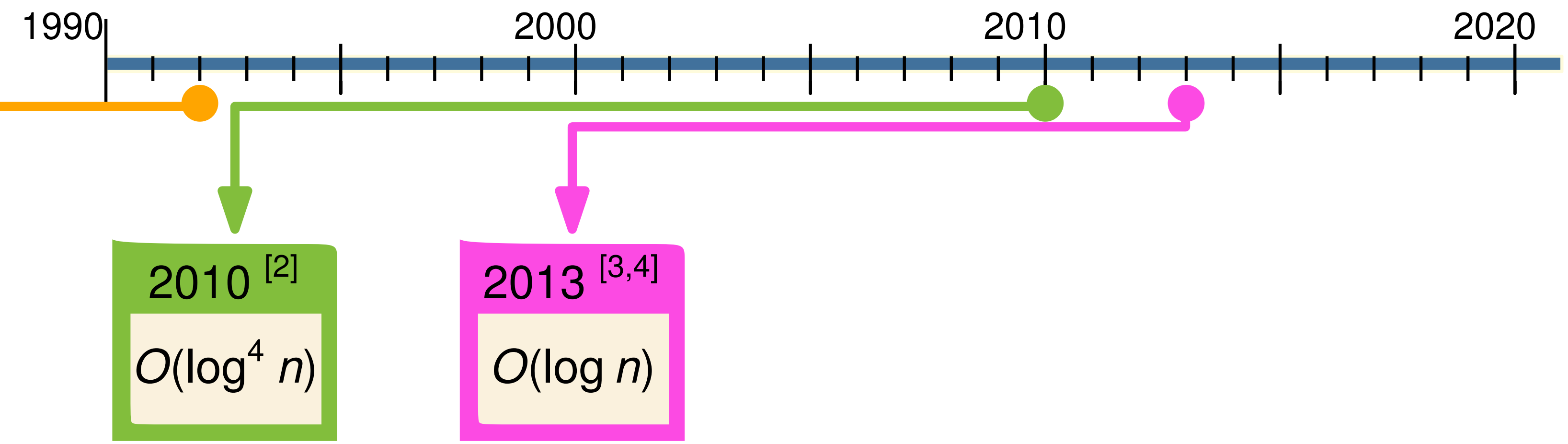
Conjecture[1]:

Planar graphs have bounded queue number.

[1] Heath, Leighton, Rosenberg *Comparing queues and stacks asmechanisms for laying out graphs* (1992)

[2] Di Battista, Frati, Pach *On the queue number of planar graphs.* (2010)

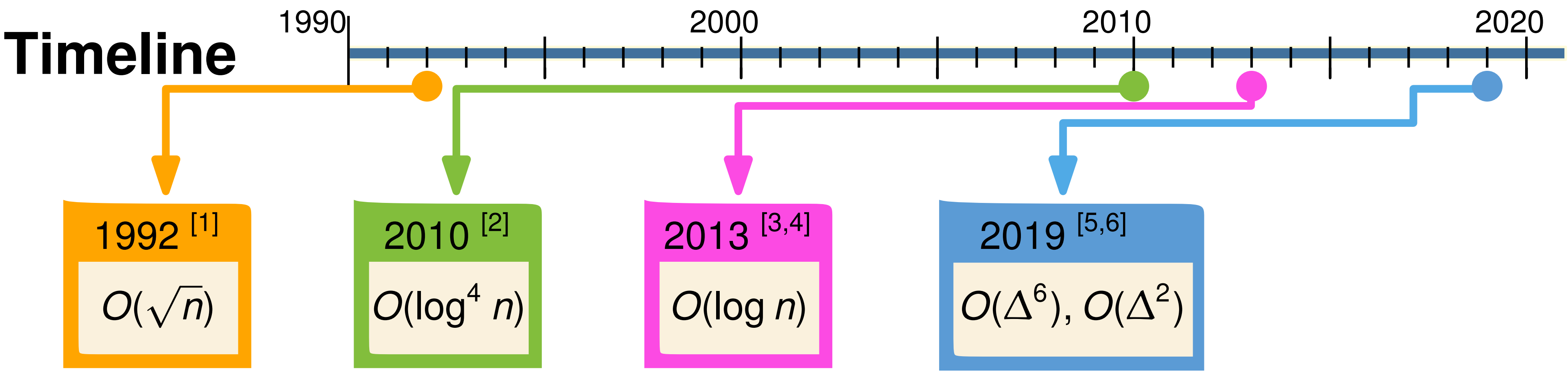
Timeline



Conjecture[1]:

Planar graphs have bounded queue number.

[1] Heath, Leighton, Rosenberg *Comparing queues and stacks asmechanisms for laying out graphs* (1992)
[2] Di Battista, Frati, Pach *On the queue number of planar graphs.* (2010)
[3] Dujmović *Graph layouts via layered separators* (2013)
[4] Dujmović, Morin, Wood *Layered separators in minor-closed graph classes with applications* (2013)



Conjecture[1]:

Planar graphs have bounded queue number.

[1] Heath, Leighton, Rosenberg *Comparing queues and stacks asmechanisms for laying out graphs* (1992)

[2] Di Battista, Frati, Pach *On the queue number of planar graphs.* (2010)

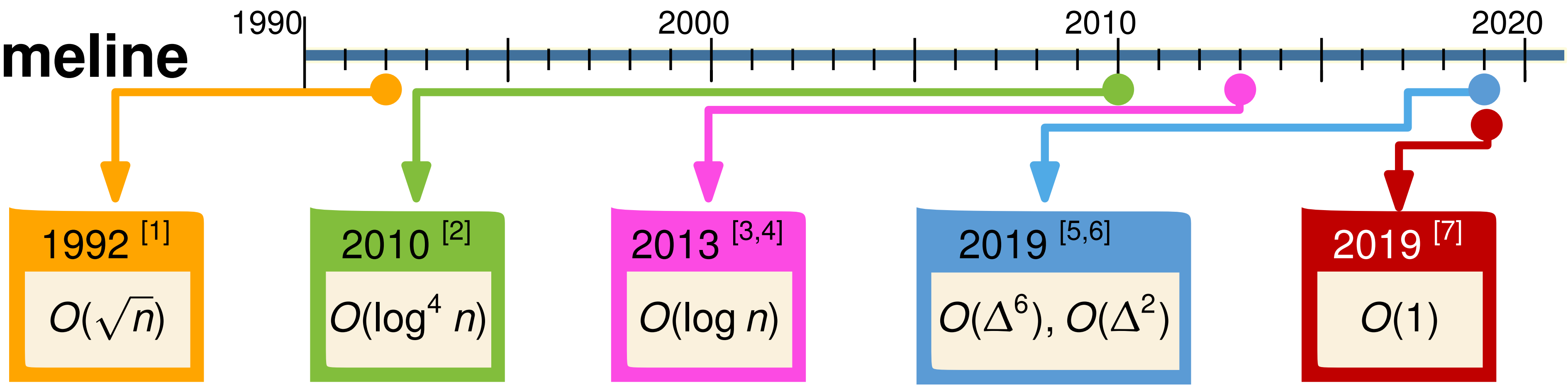
[3] Dujmović *Graph layouts via layered separators* (2013)

[4] Dujmović, Morin, Wood *Layered separators in minor-closed graph classes with applications* (2013)

[5] Bekos, Förster, Gronemann, Mchedlidze, Montecchiani, Raftopoulou, Ueckerdt:
Planar graphs of bounded degree have boundedqueue number (2019)

[6] Dujmović, Morin, Wood *Queue layouts of graphs with bounded degree and bounded genus* (2019)

Timeline



Conjecture[1]:

Planar graphs have bounded queue number.

[1] Heath, Leighton, Rosenberg *Comparing queues and stacks asmechanisms for laying out graphs* (1992)

[2] Di Battista, Frati, Pach *On the queue number of planar graphs.* (2010)

[3] Dujmović *Graph layouts via layered separators* (2013)

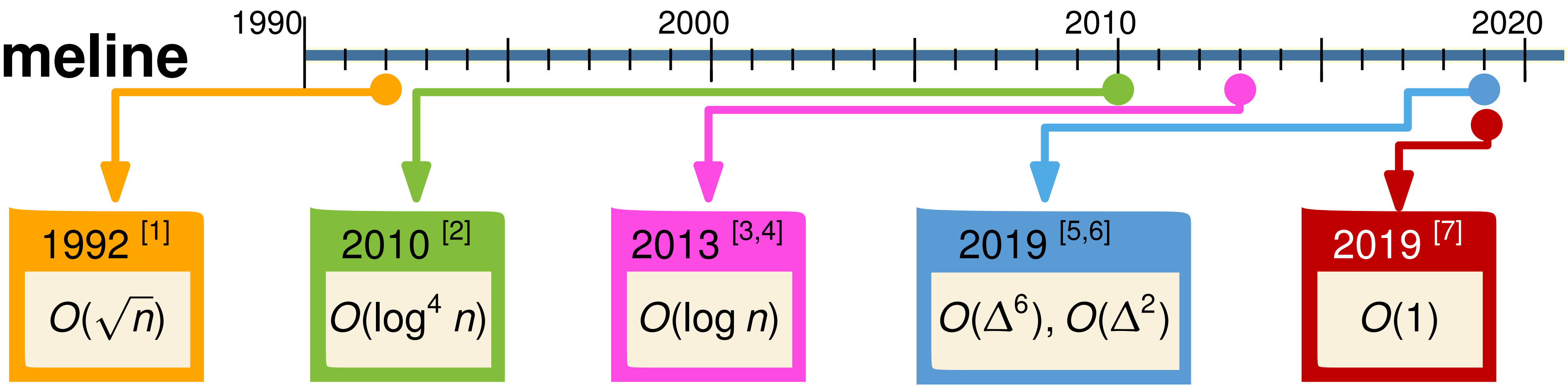
[4] Dujmović, Morin, Wood *Layered separators in minor-closed graph classes with applications* (2013)

[5] Bekos, Förster, Gronemann, Mchedlidze, Montecchiani, Raftopoulou, Ueckerdt: *Planar graphs of bounded degree have boundedqueue number* (2019)

[6] Dujmović, Morin, Wood *Queue layouts of graphs with bounded degree and bounded genus* (2019)

[7] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood *Planar graphs have bounded queue-number* (2019)

Timeline



Conjecture[1]:

Planar graphs have bounded queue number.

Theorem[7]:

Planar graphs have queue number at most 49.

[1] Heath, Leighton, Rosenberg *Comparing queues and stacks asmechanisms for laying out graphs* (1992)
[2] Di Battista, Frati, Pach *On the queue number of planar graphs.* (2010)
[3] Dujmović *Graph layouts via layered separators* (2013)
[4] Dujmović, Morin, Wood *Layered separators in minor-closed graph classes with applications* (2013)
[5] Bekos, Förster, Gronemann, Mchedlidze, Montecchiani, Raftopoulou, Ueckerdt:
Planar graphs of bounded degree have boundedqueue number (2019)
[6] Dujmović, Morin, Wood *Queue layouts of graphs with bounded degree and bounded genus* (2019)
[7] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood *Planar graphs have bounded queue-number* (2019)

Strong Product

Definition: $G \boxtimes H$

strong product of G and H

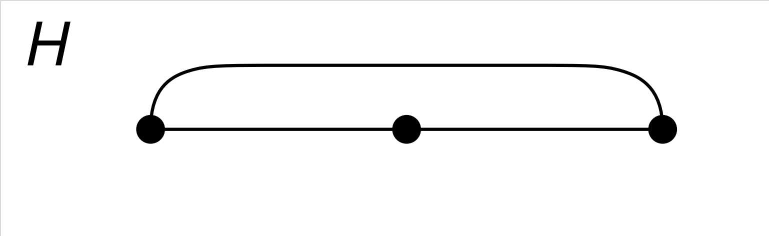
(assume that $G = P_k$)

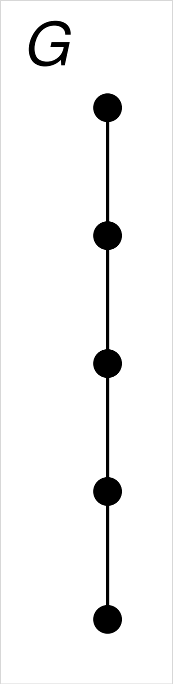
Strong Product

Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)





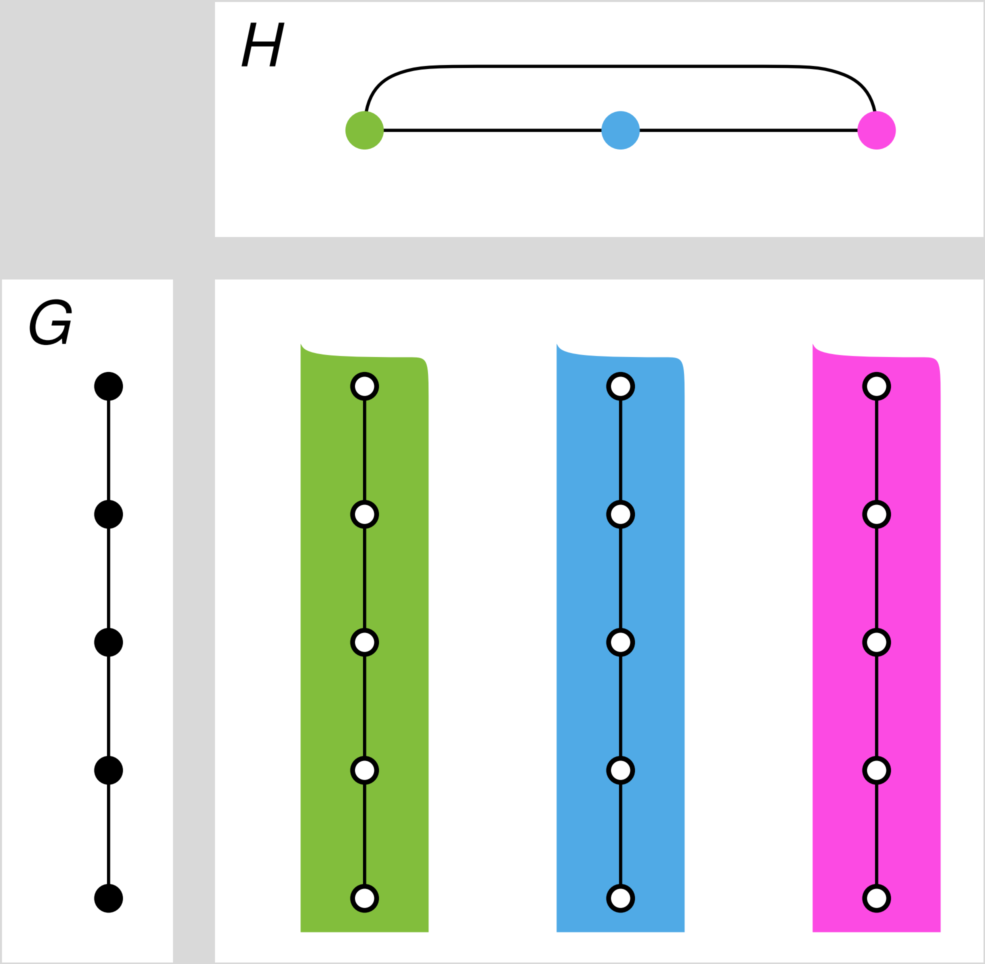
Strong Product

Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)

- vertices of H become copies of G



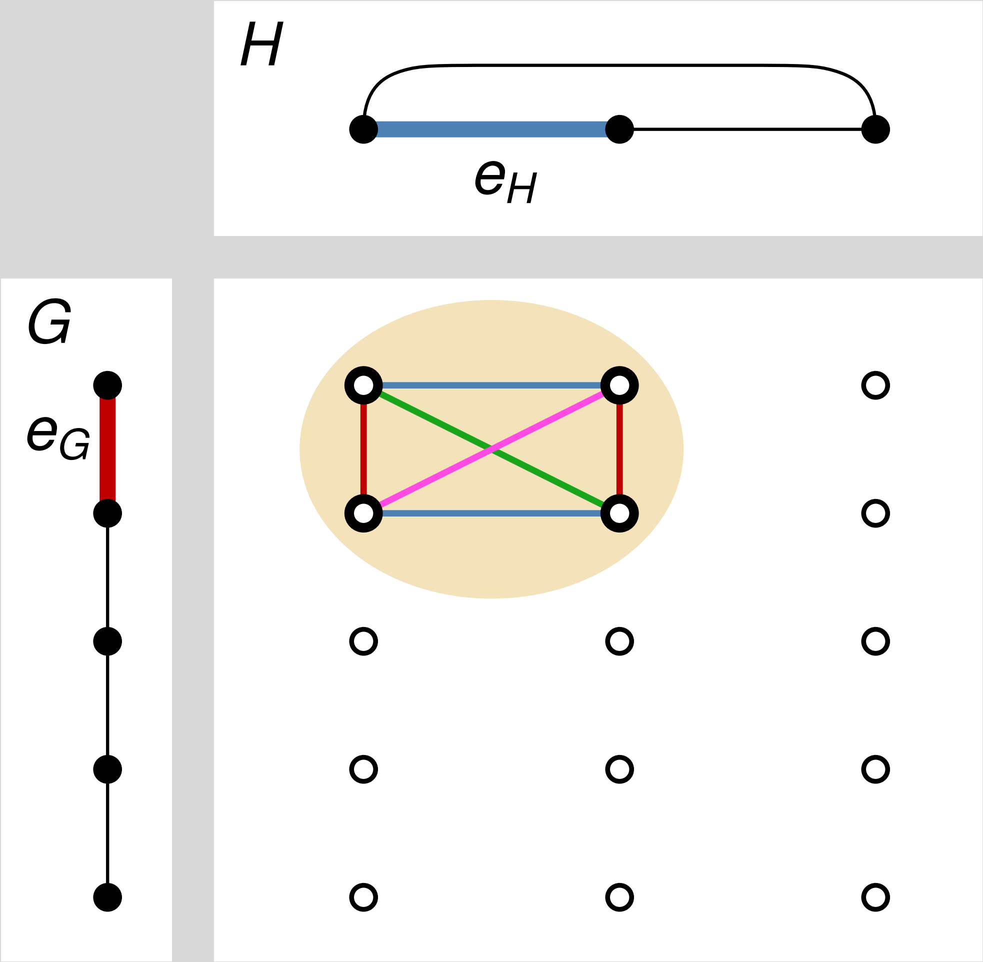
Strong Product

Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)

- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4



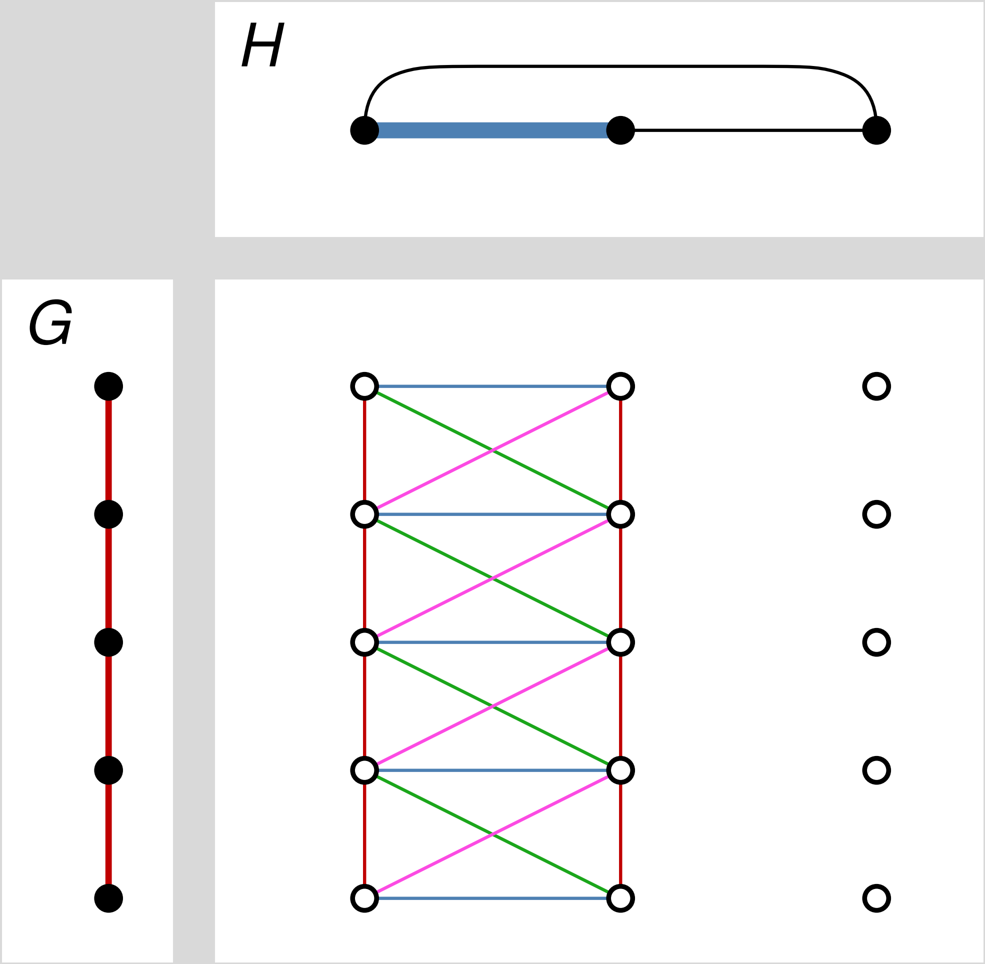
Strong Product

Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)

- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4



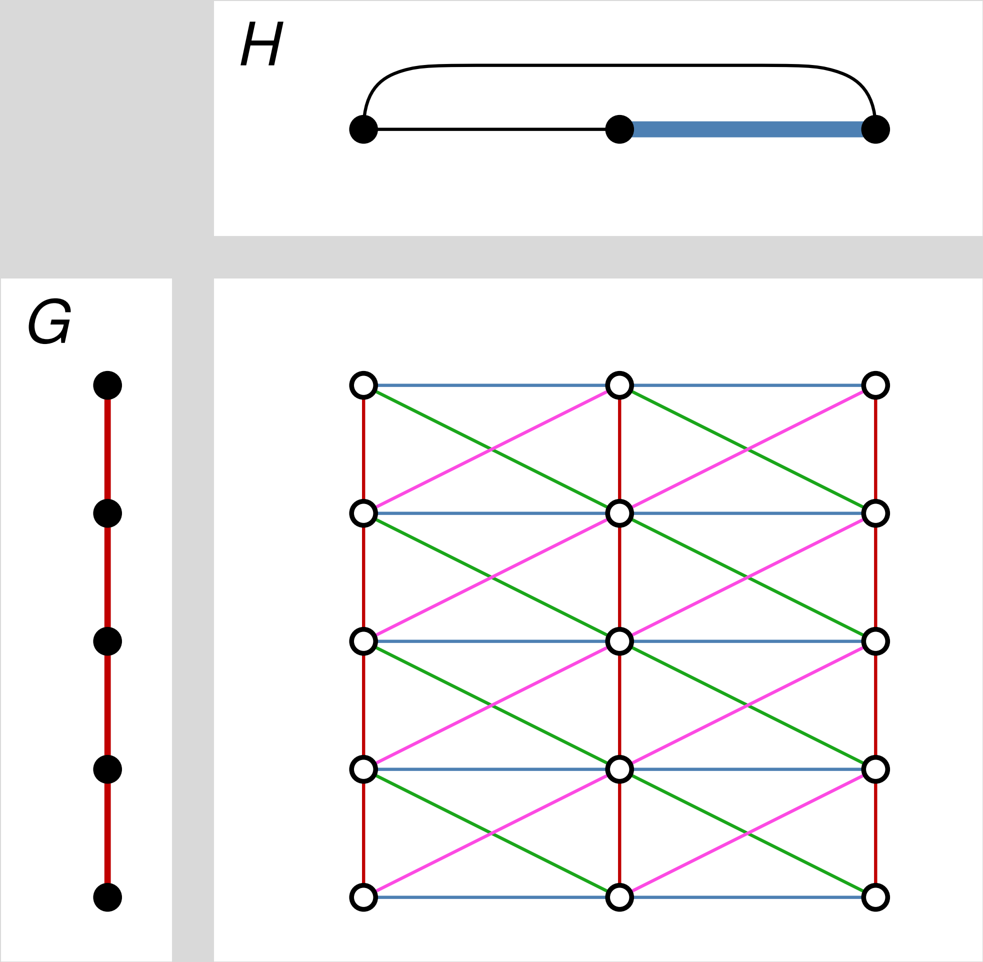
Strong Product

Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)

- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4



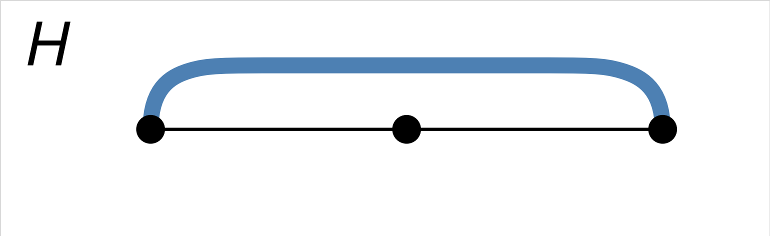
Strong Product

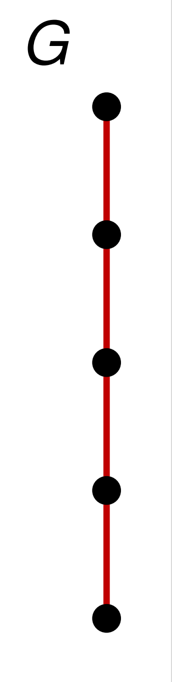
Definition: $G \boxtimes H$

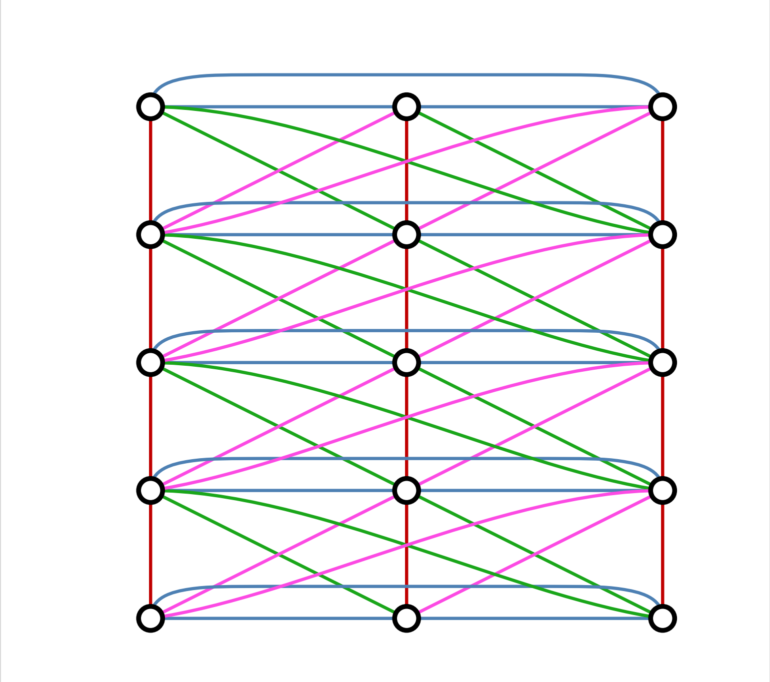
strong product of G and H

(assume that $G = P_k$)

- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4







Strong Product

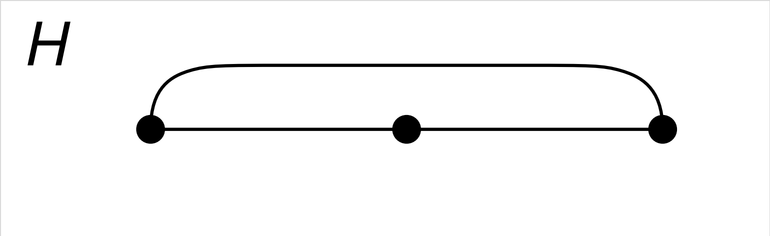
Definition: $G \boxtimes H$

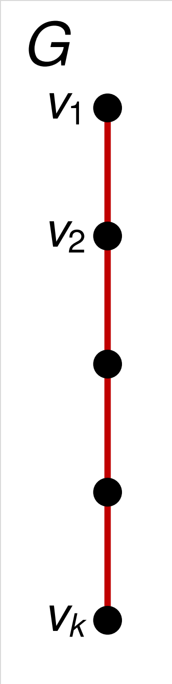
strong product of G and H

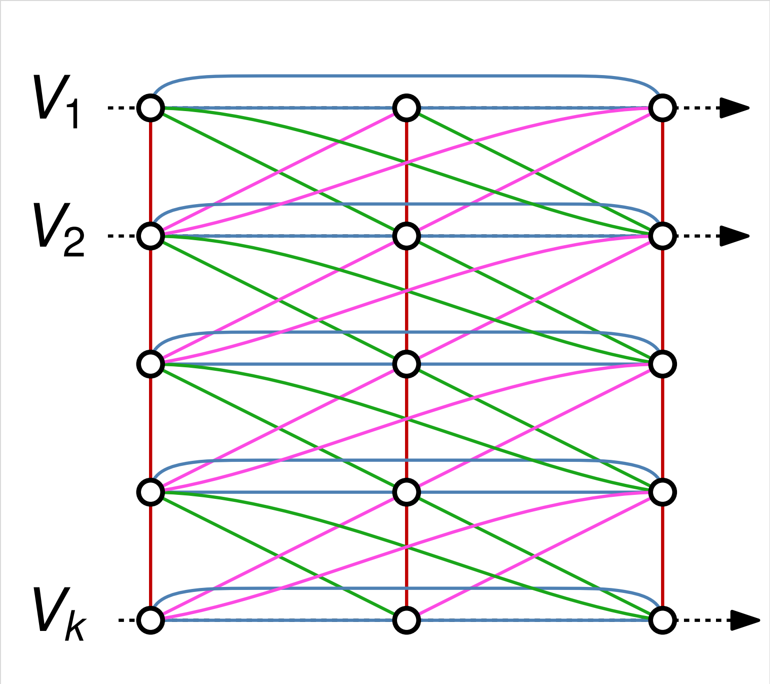
(assume that $G = P_k$)

- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4

P_k implies a leveling $\{V_1, V_2, \ldots, V_k\}$







Strong Product

Definition: $G \boxtimes H$

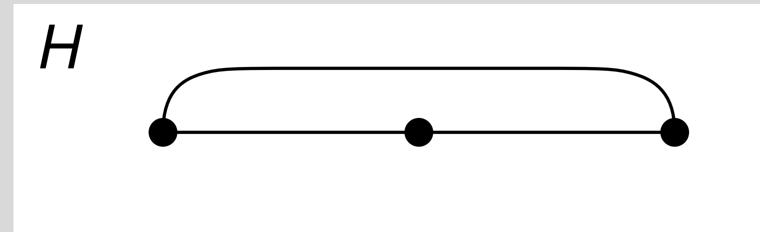
strong product of G and H

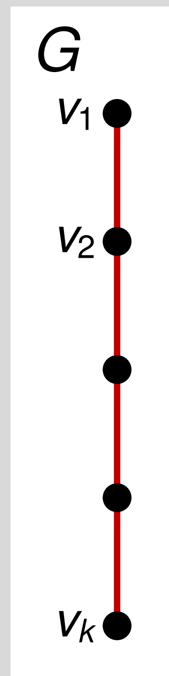
(assume that $G = P_k$)

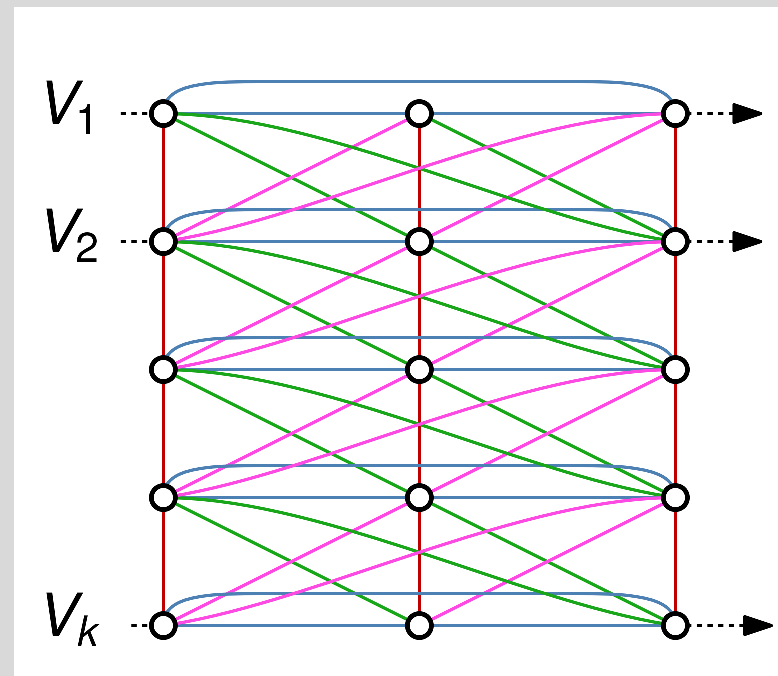
- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4

P_k implies a leveling $\{V_1, V_2, \ldots, V_k\}$

Types of edges







Strong Product

Definition: $G \boxtimes H$

strong product of G and H

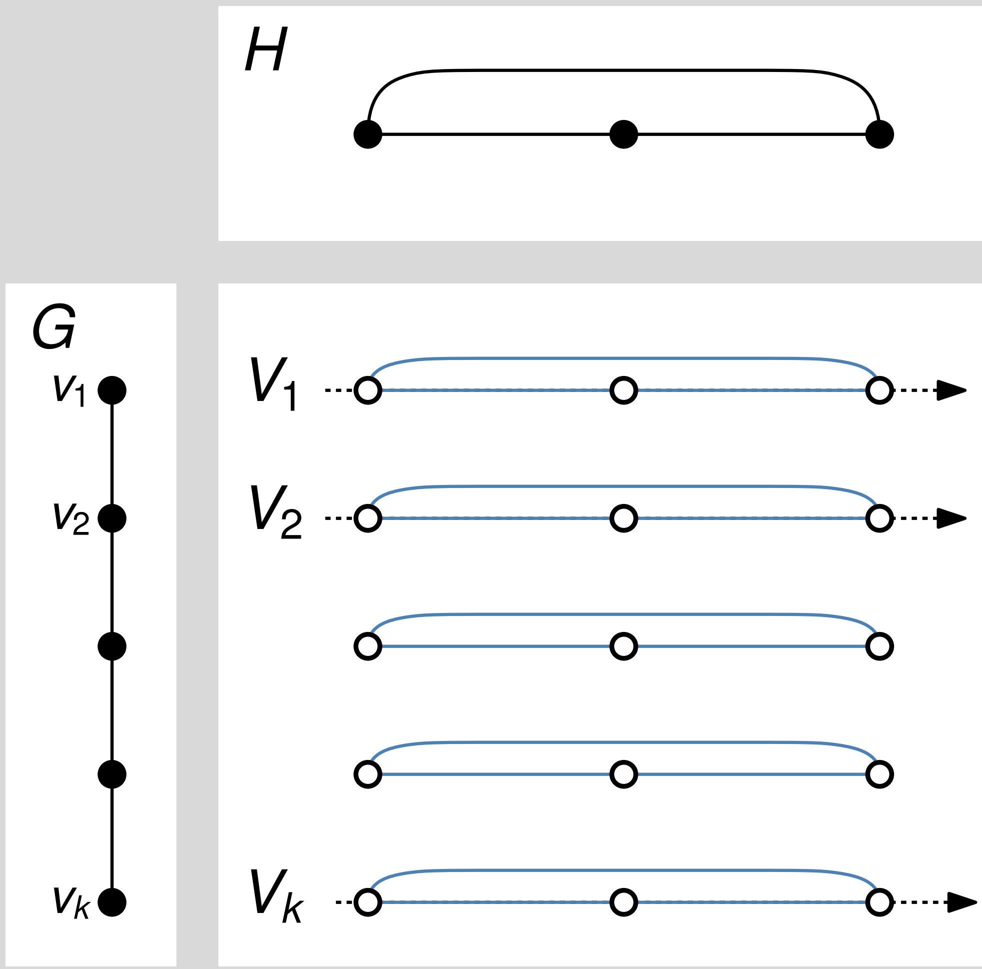
(assume that $G = P_k$)

- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4

P_k implies a leveling $\{V_1, V_2, \ldots, V_k\}$

Types of edges

level



Strong Product

Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)

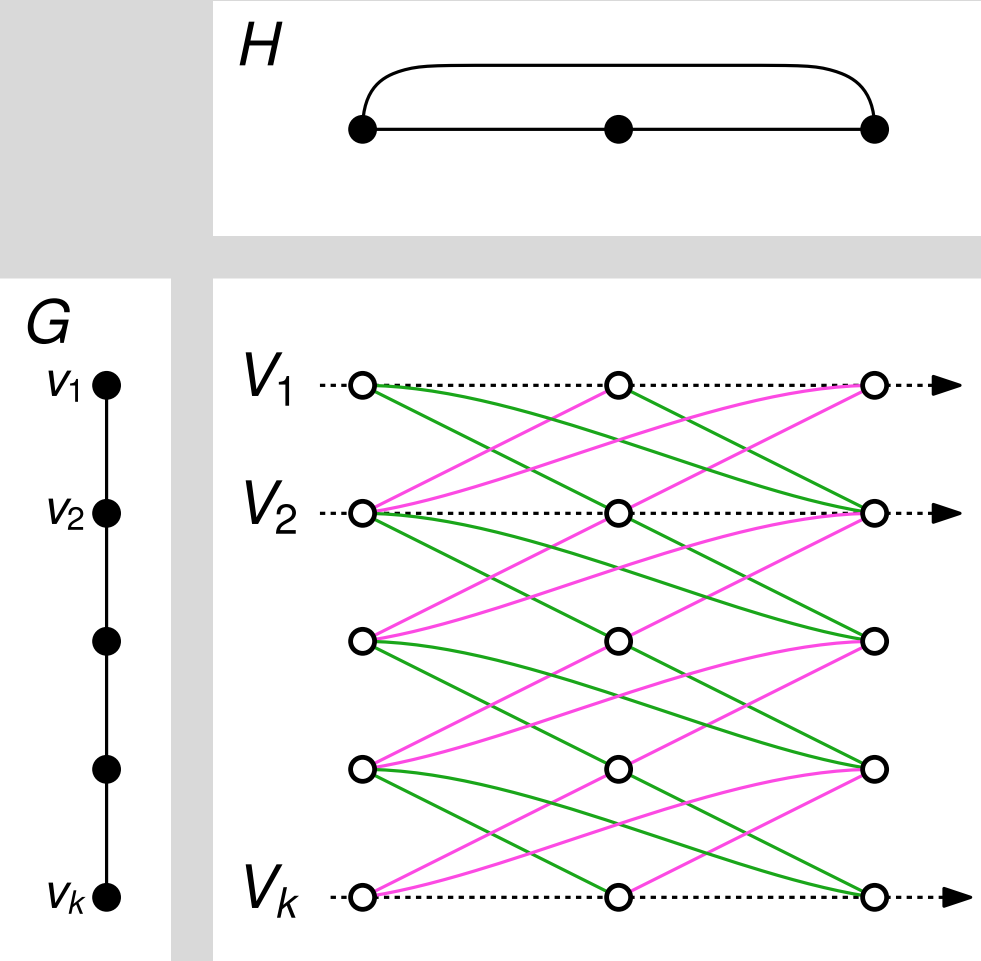
- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4

P_k implies a leveling $\{V_1, V_2, \ldots, V_k\}$

Types of edges

level

binding



Strong Product

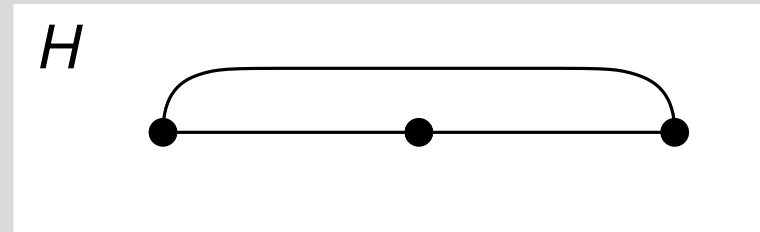
Definition: $G \boxtimes H$

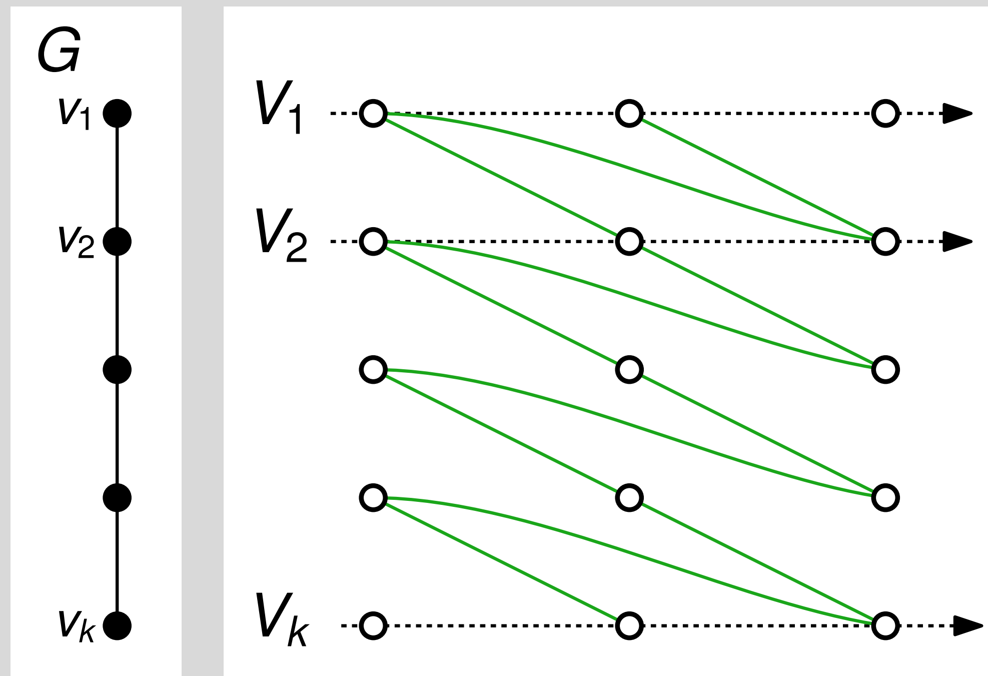
strong product of G and H

(assume that $G = P_k$)

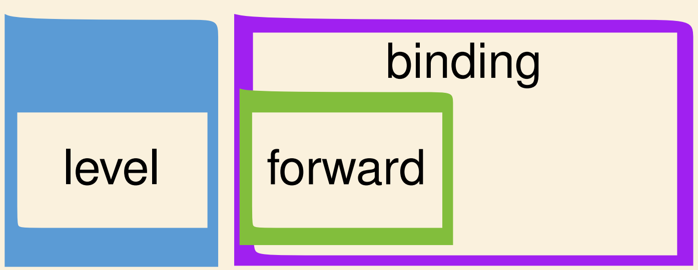
- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4

P_k implies a leveling $\{V_1, V_2, \ldots, V_k\}$





Types of edges



Strong Product

Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)

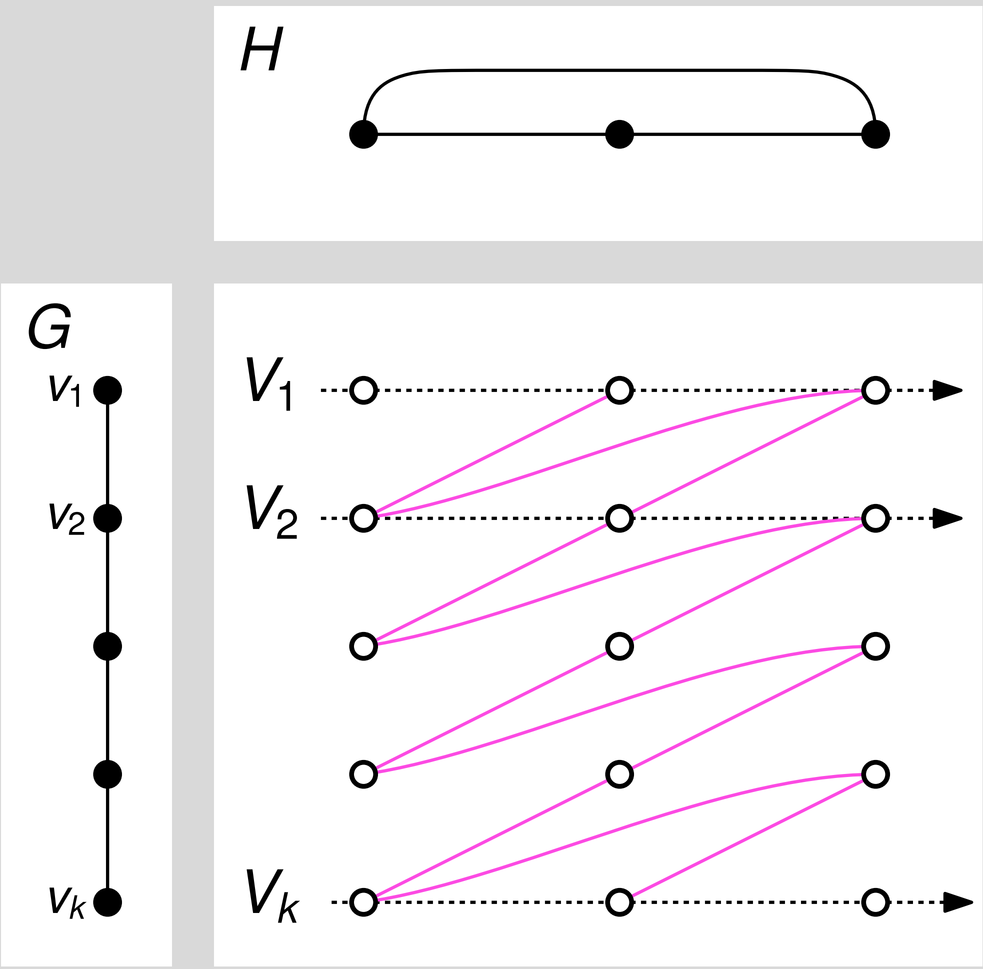
- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4

P_k implies a leveling $\{V_1, V_2, \ldots, V_k\}$

Types of edges

level

binding: forward, backward



Strong Product

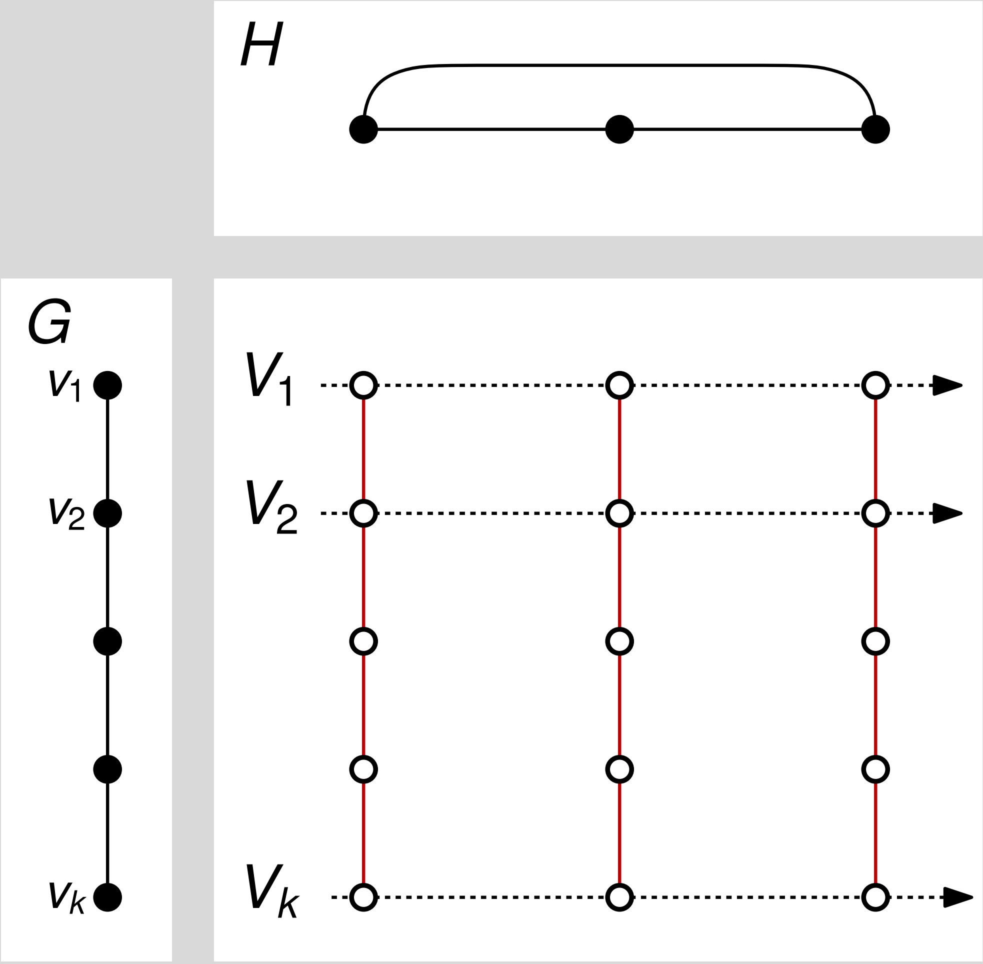
Definition: $G \boxtimes H$

strong product of G and H

(assume that $G = P_k$)

- vertices of H become copies of G
- edges e_G and e_H give rise to a K_4

P_k implies a leveling $\{V_1, V_2, \ldots, V_k\}$



Types of edges

level

binding: forward, backward

vertex

Strong Product

Queue number[1]: $P \boxtimes H$

$$\mathrm{qn}(P \boxtimes H) \leq 3 \cdot \mathrm{qn}(H) + 1$$

[1] Wood, *Queue layouts of graph products and powers* (2005)

Strong Product

Queue number[1]: $P \boxtimes H$

$$\text{qn}(P \boxtimes H) \leq 3 \cdot \text{qn}(H) + 1$$

Theorem [2]:

$\text{qn}(G) = k$ if and only if there is a vertex order with no $(k+1)$-rainbow

[1] Wood, *Queue layouts of graph products and powers* (2005)
[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product

Queue number[1]: $P \boxtimes H$

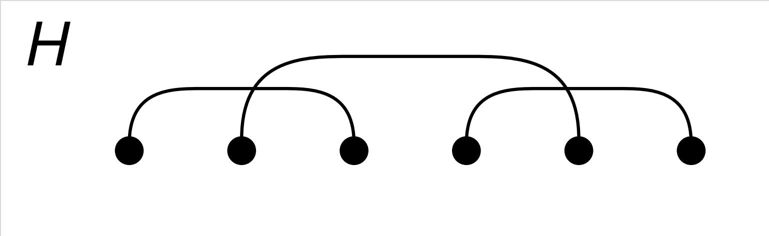
$\text{qn}(P \boxtimes H) \leq 3 \cdot \text{qn}(H) + 1$

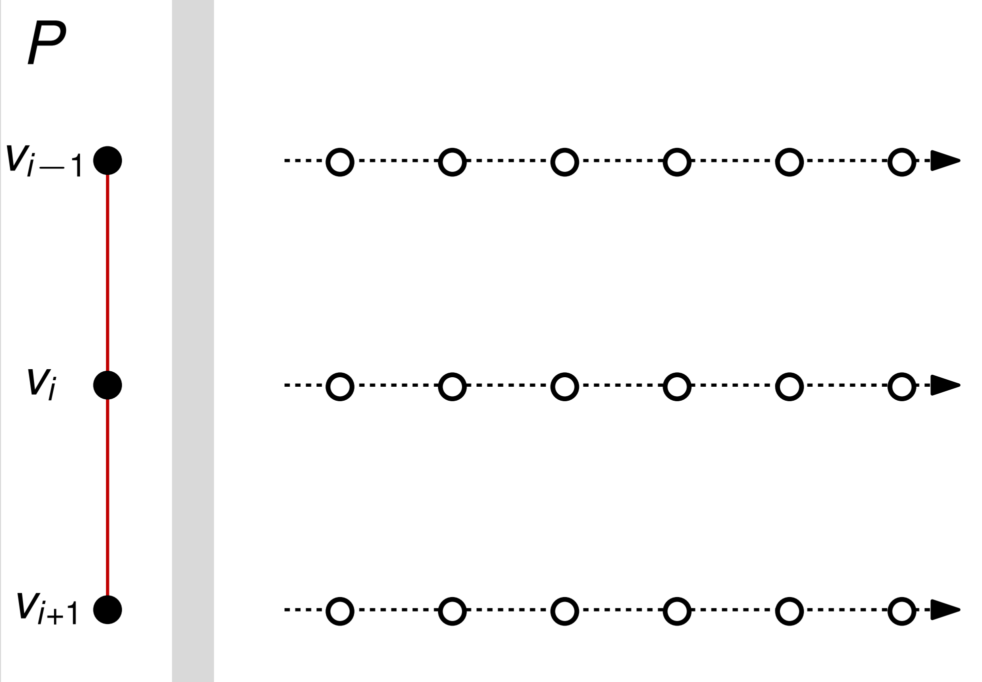
Theorem [2]:

$\text{qn}(G) = k$ if and only if there is a vertex order with no $(k+1)$-rainbow

- queue layout of H with $\text{qn}(H)$ pages

edges on one queue of H





[1] Wood, *Queue layouts of graph products and powers* (2005)
[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product

Queue number[1]: $P \boxtimes H$

$$\mathrm{qn}(P \boxtimes H) \leq 3 \cdot \mathrm{qn}(H) + 1$$

Theorem [2]:

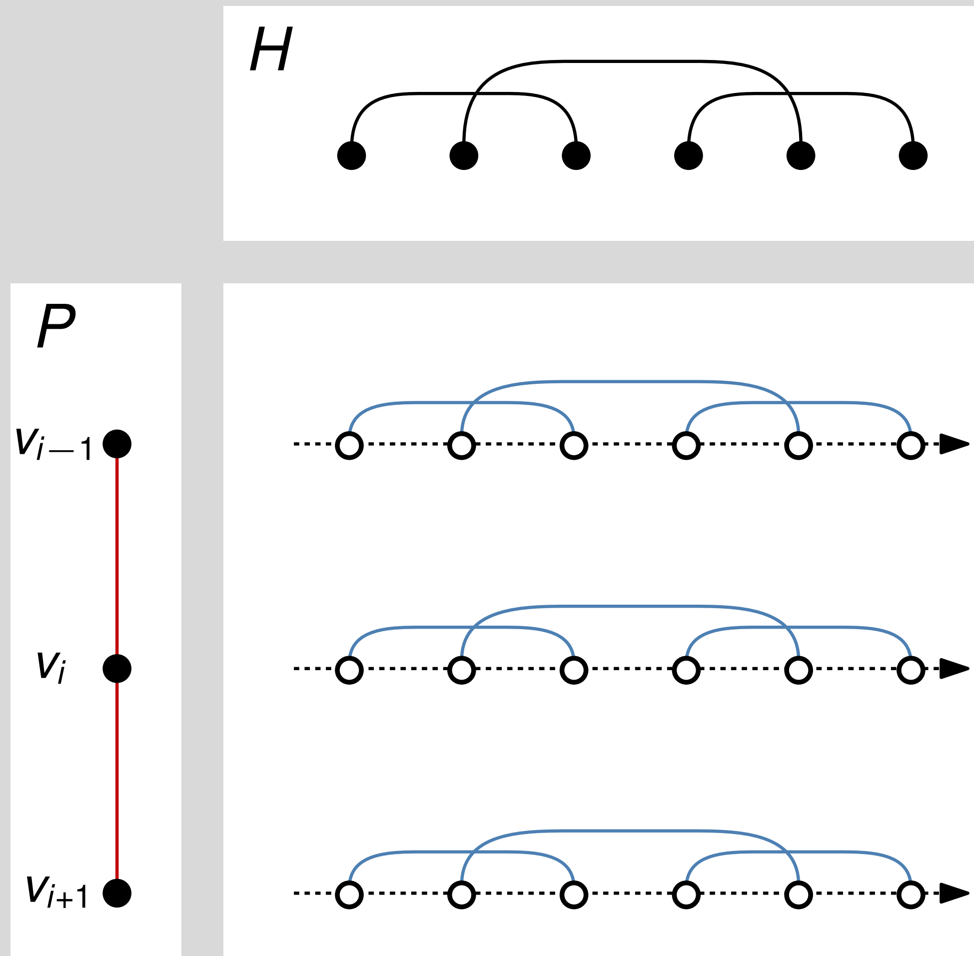
$\mathrm{qn}(G) = k$ if and only if there is a vertex order with no $(k+1)$-rainbow

- queue layout of H with $\mathrm{qn}(H)$ pages

edges on one queue of H

level

one



[1] Wood, *Queue layouts of graph products and powers* (2005)

[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product

Queue number[1]: $P \boxtimes H$

$$\mathrm{qn}(P \boxtimes H) \leq 3 \cdot \mathrm{qn}(H) + 1$$

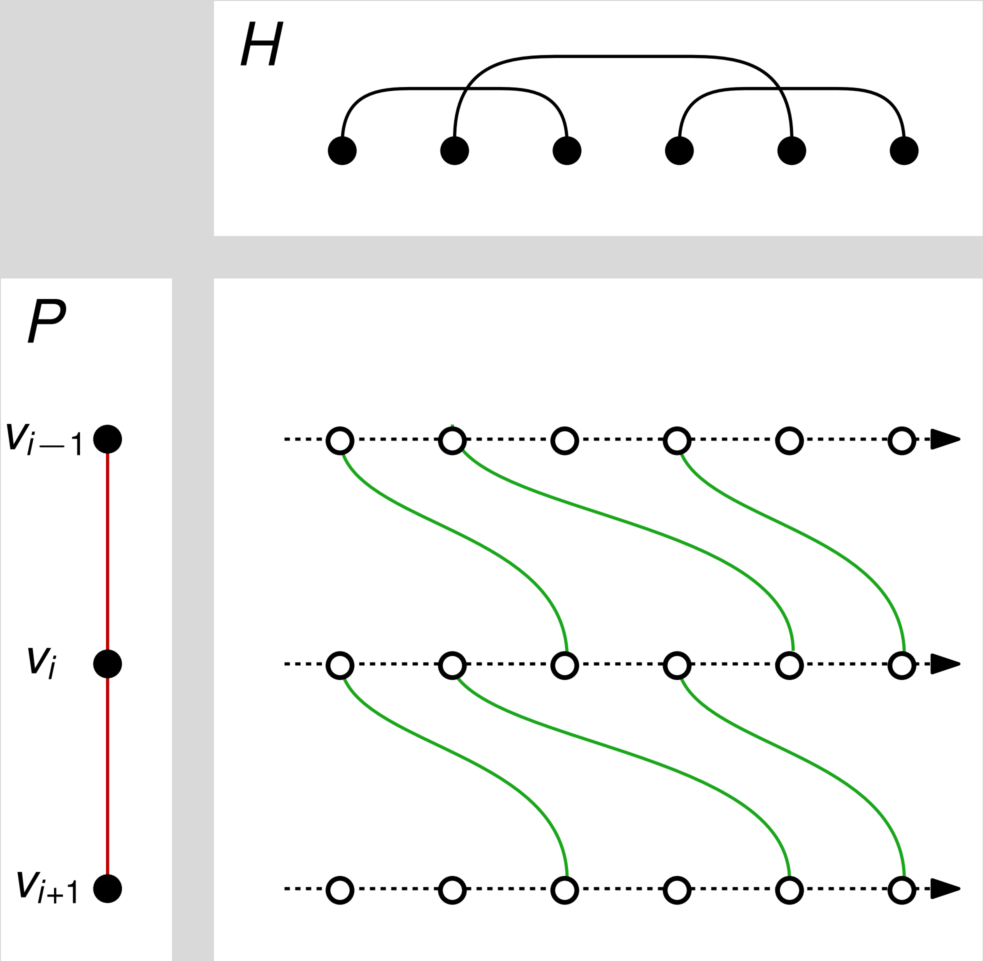
Theorem [2]:

$\mathrm{qn}(G) = k$ if and only if there is a vertex order with no $(k+1)$-rainbow

- queue layout of H with $\mathrm{qn}(H)$ pages

edges on one queue of H

level	forward
one	one



[1] Wood, *Queue layouts of graph products and powers* (2005)
[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product

Queue number[1]: $P \boxtimes H$

$$\mathrm{qn}(P \boxtimes H) \leq 3 \cdot \mathrm{qn}(H) + 1$$

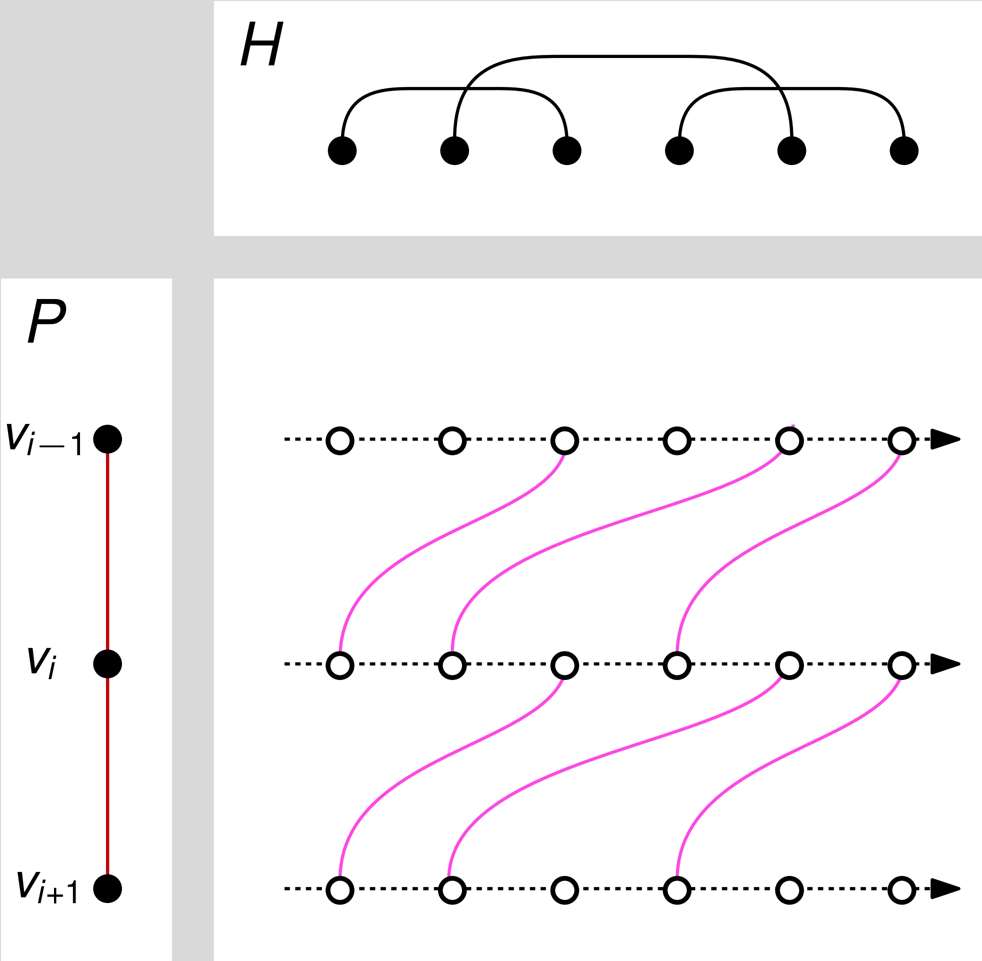
Theorem [2]:

$\mathrm{qn}(G) = k$ if and only if there is a vertex order with no $(k+1)$-rainbow

- queue layout of H with $\mathrm{qn}(H)$ pages

edges on one queue of H

level	forward	backward
one	one	one



[1] Wood, *Queue layouts of graph products and powers* (2005)
[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product

Queue number[1]: $P \boxtimes H$

$$\text{qn}(P \boxtimes H) \leq 3 \cdot \text{qn}(H) + 1$$

Theorem [2]:

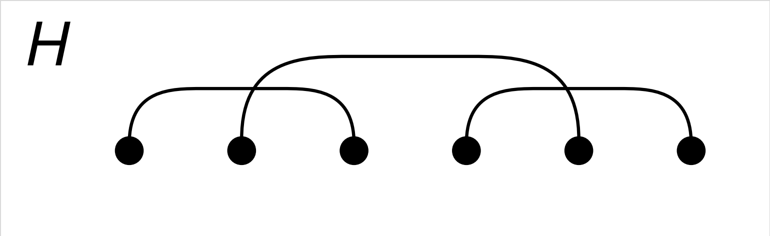
$\text{qn}(G) = k$ if and only if there is a vertex order with no $(k+1)$-rainbow

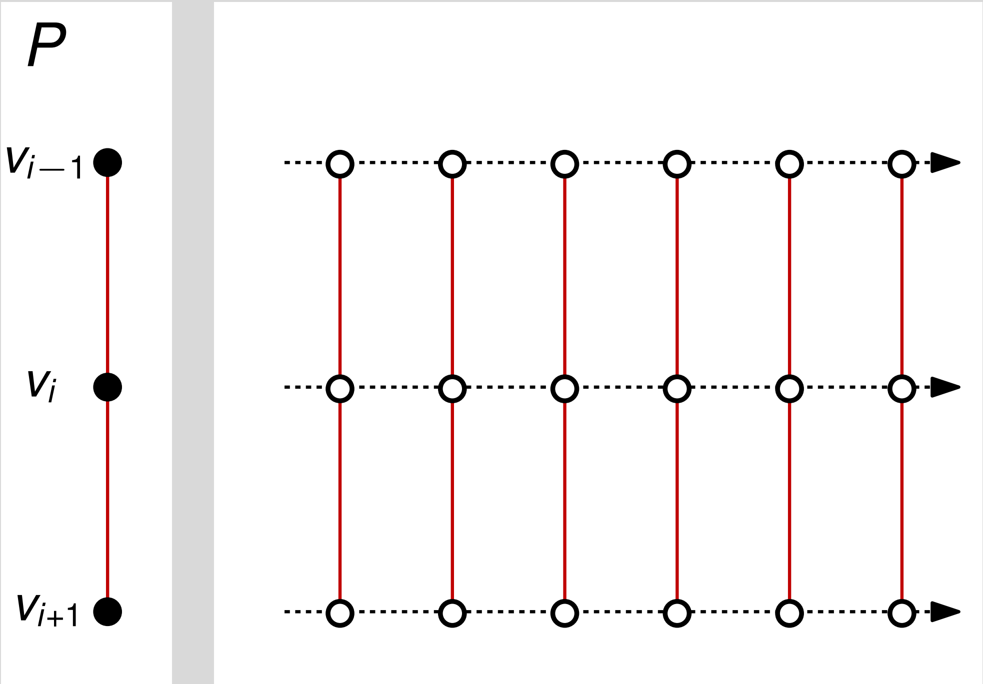
- queue layout of H with $\text{qn}(H)$ pages

edges on one queue of H

level	forward	backward
one	one	one

vertex
one





[1] Wood, *Queue layouts of graph products and powers* (2005)
[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product

Queue number[1]: $P \boxtimes H$

$$\mathrm{qn}(P \boxtimes H) \leq 3 \cdot \mathrm{qn}(H) + 1$$

Theorem [2]:

qn(G) = k if and only if there is a vertex order with no ($k + 1$)-rainbow

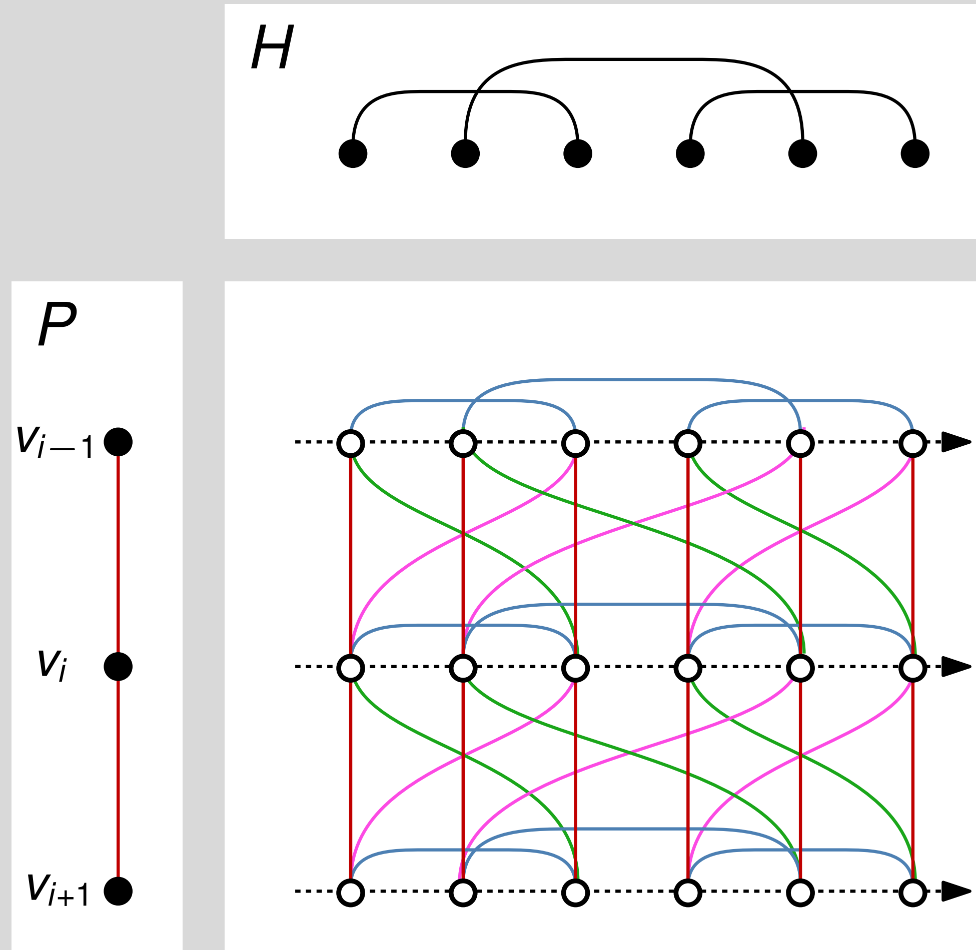
- queue layout of H with qn(H) pages

edges on one queue of H

level	forward	backward
one	one	one

vertex
one

$\boxed{3} \cdot \mathrm{qn}(H) + \boxed{1}$



[1] Wood, *Queue layouts of graph products and powers* (2005)

[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product

Queue number[1]: $P \boxtimes H$

$$\mathrm{qn}(P \boxtimes H) \leq 3 \cdot \mathrm{qn}(H) + 1$$

Theorem [2]:

$\mathrm{qn}(G) = k$ if and only if there is a vertex order with no $(k+1)$-rainbow

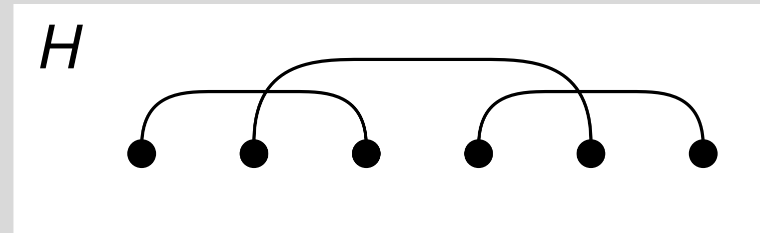
- queue layout of H with $\mathrm{qn}(H)$ pages

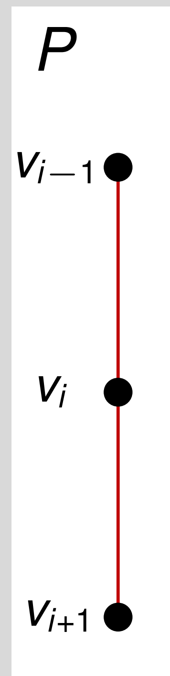
edges on one queue of H

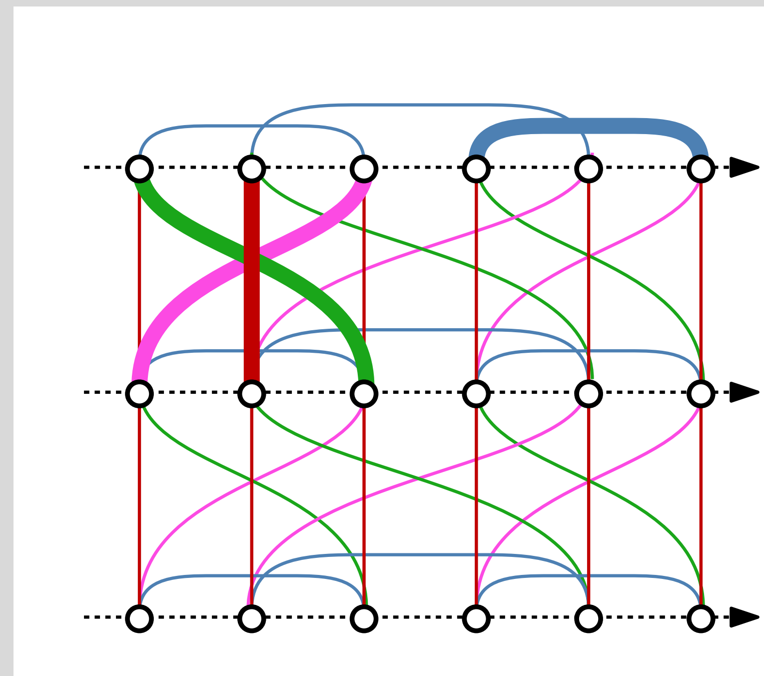
level	forward	backward
one	one	one

vertex
one

$$3 \cdot \mathrm{qn}(H) + 1$$







[1] Wood, *Queue layouts of graph products and powers* (2005)
[2] Heath, Rosenberg: *Laying out graphs using queues* (1992)

Strong Product of planar graphs

Theorem[1]:

G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

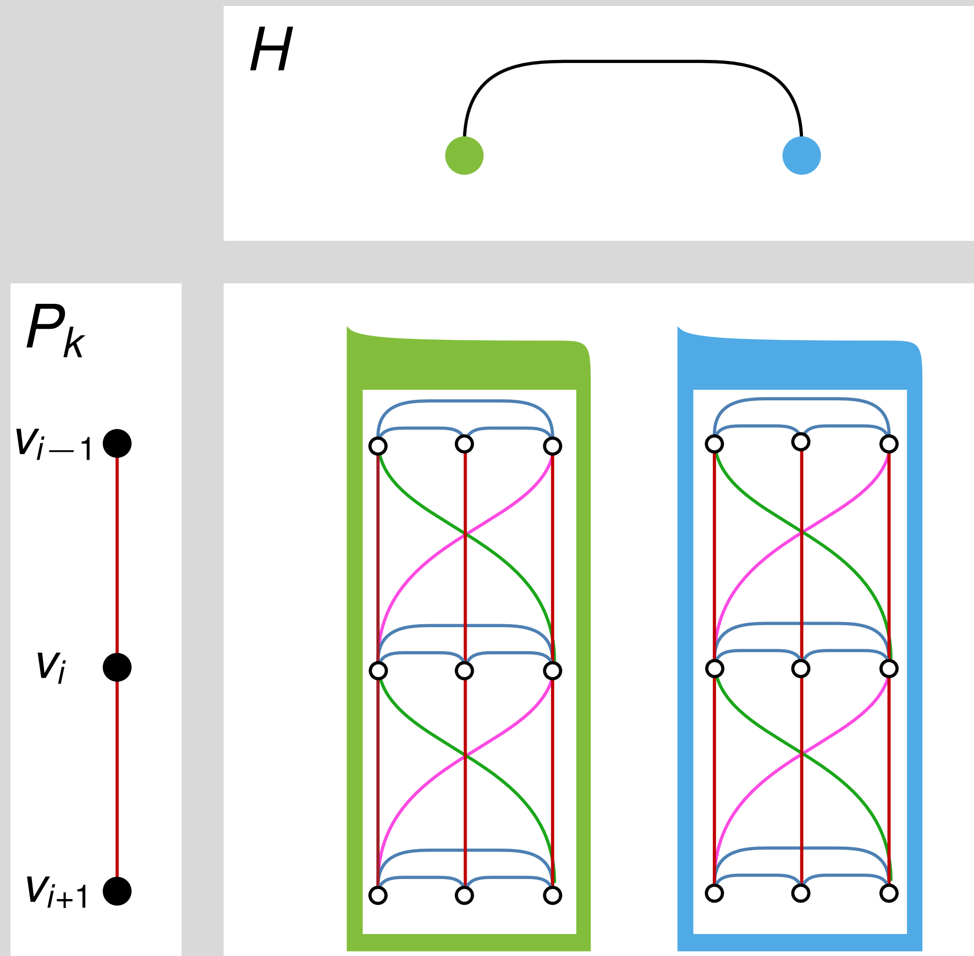
[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

Strong Product of planar graphs

Theorem[1]:

G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

- vertex of H becomes $P_k \boxtimes K_3$



[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

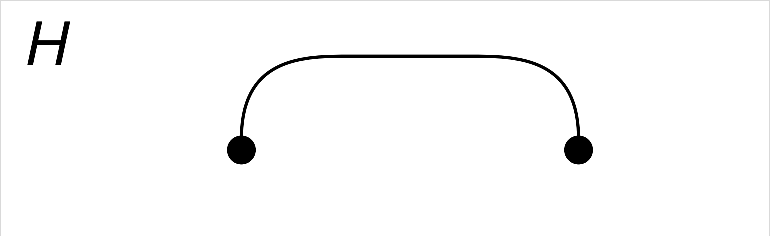
Strong Product of planar graphs

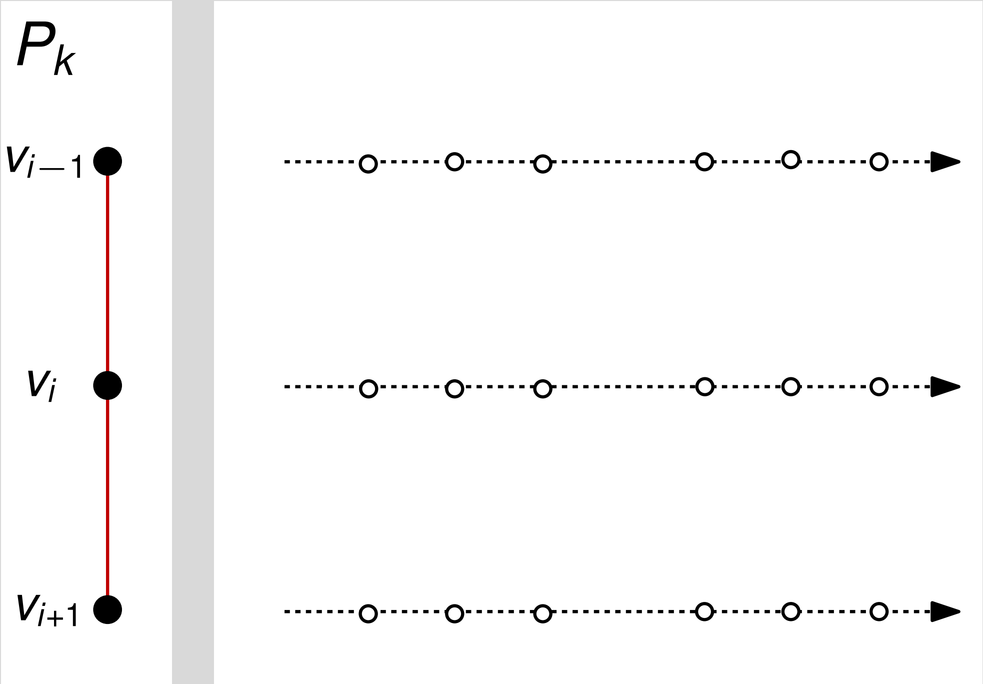
Theorem[1]:

G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

- vertex of H becomes $P_k \boxtimes K_3$
- edge becomes a $K_{3,3}$

edges on one queue of H





[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

Strong Product of planar graphs

Theorem[1]:

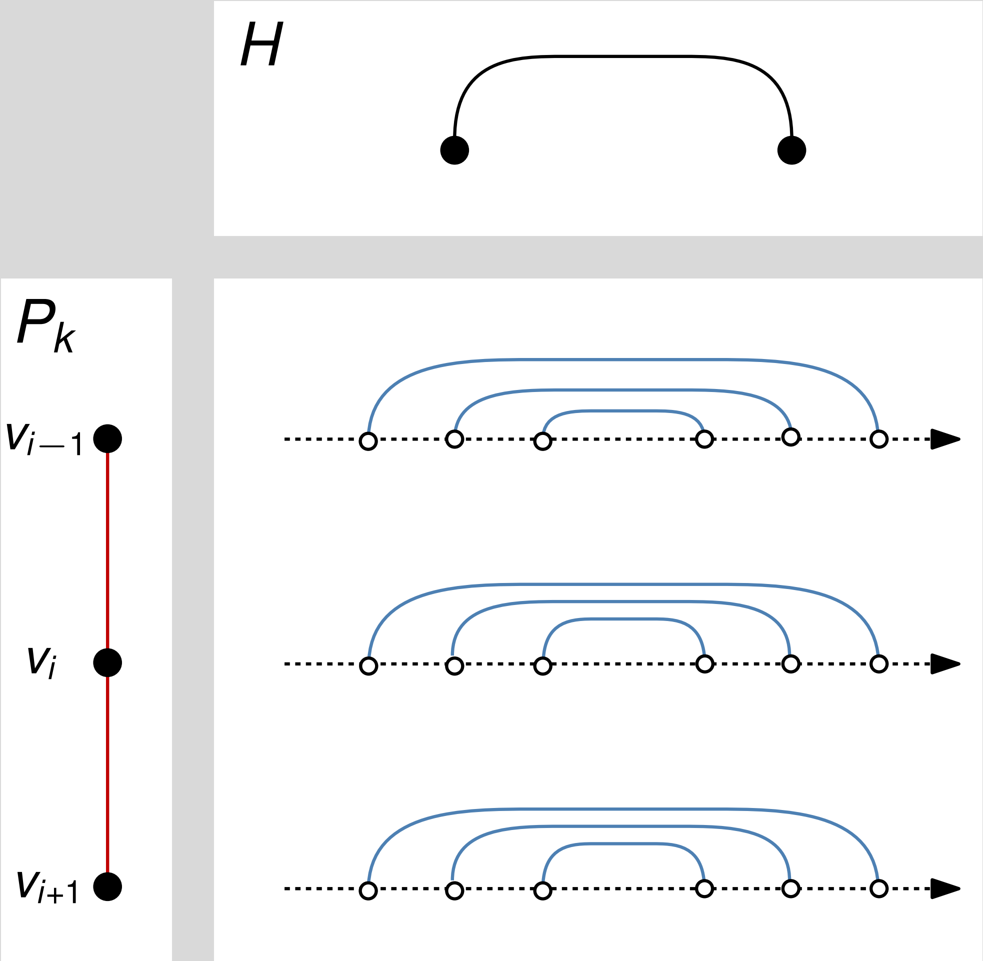
G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

- vertex of H becomes $P_k \boxtimes K_3$
- edge becomes a $K_{3,3}$

edges on one queue of H

level

three



[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

Strong Product of planar graphs

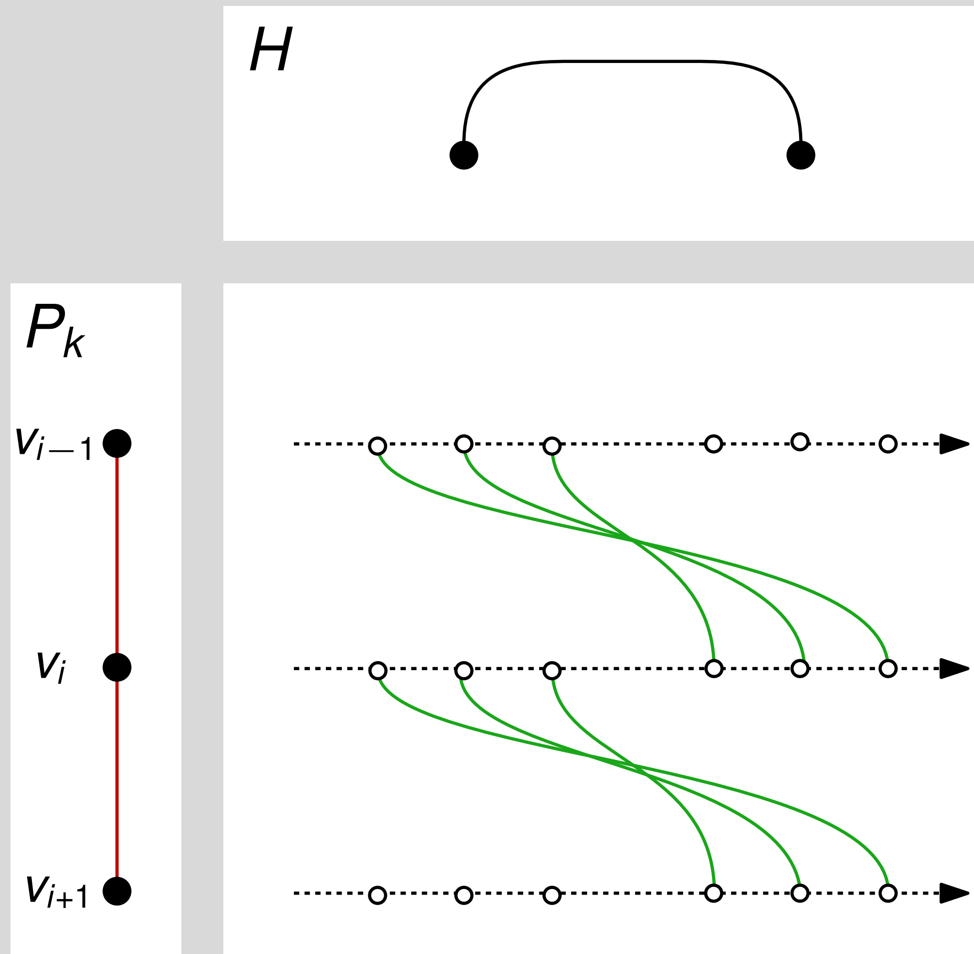
Theorem[1]:

G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

- vertex of H becomes $P_k \boxtimes K_3$
- edge becomes a $K_{3,3}$

edges on one queue of H

level	forward
three	three



[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

Strong Product of planar graphs

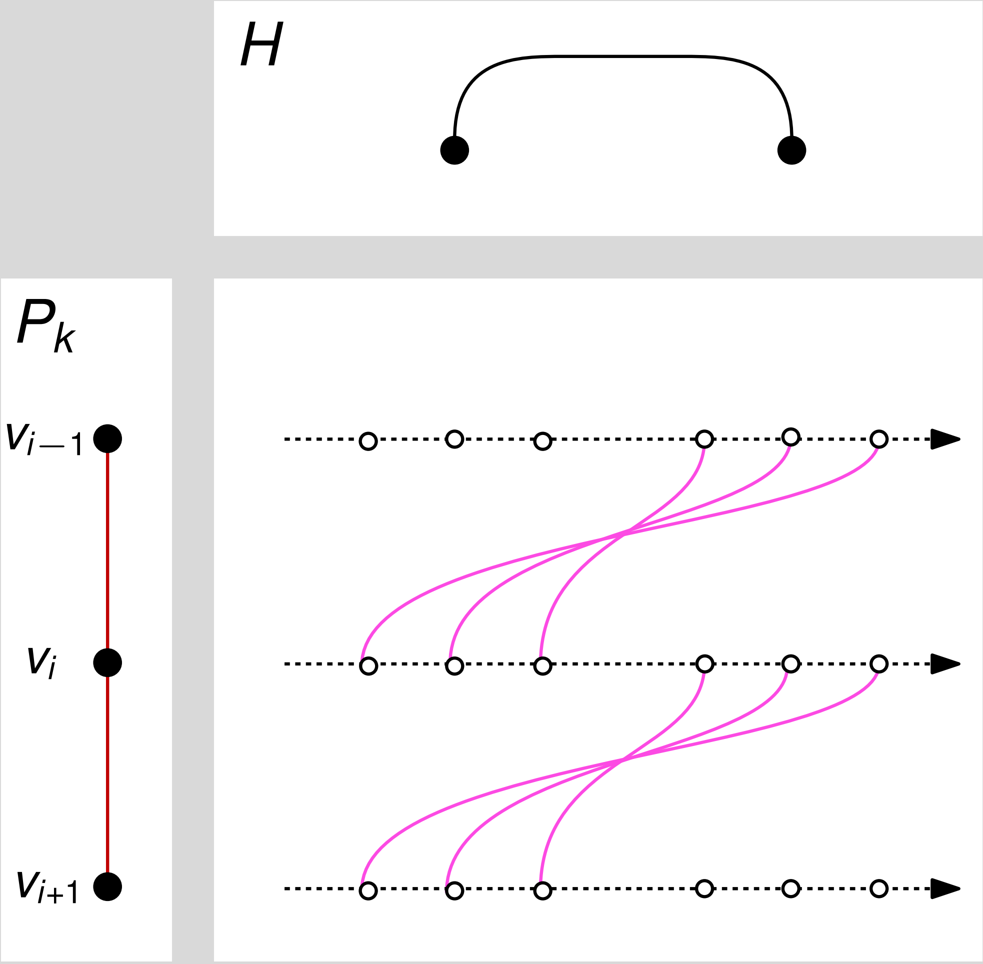
Theorem[1]:

G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

- vertex of H becomes $P_k \boxtimes K_3$
- edge becomes a $K_{3,3}$

edges on one queue of H

level	forward	backward
three	three	three



[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

Strong Product of planar graphs

Theorem[1]:

G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

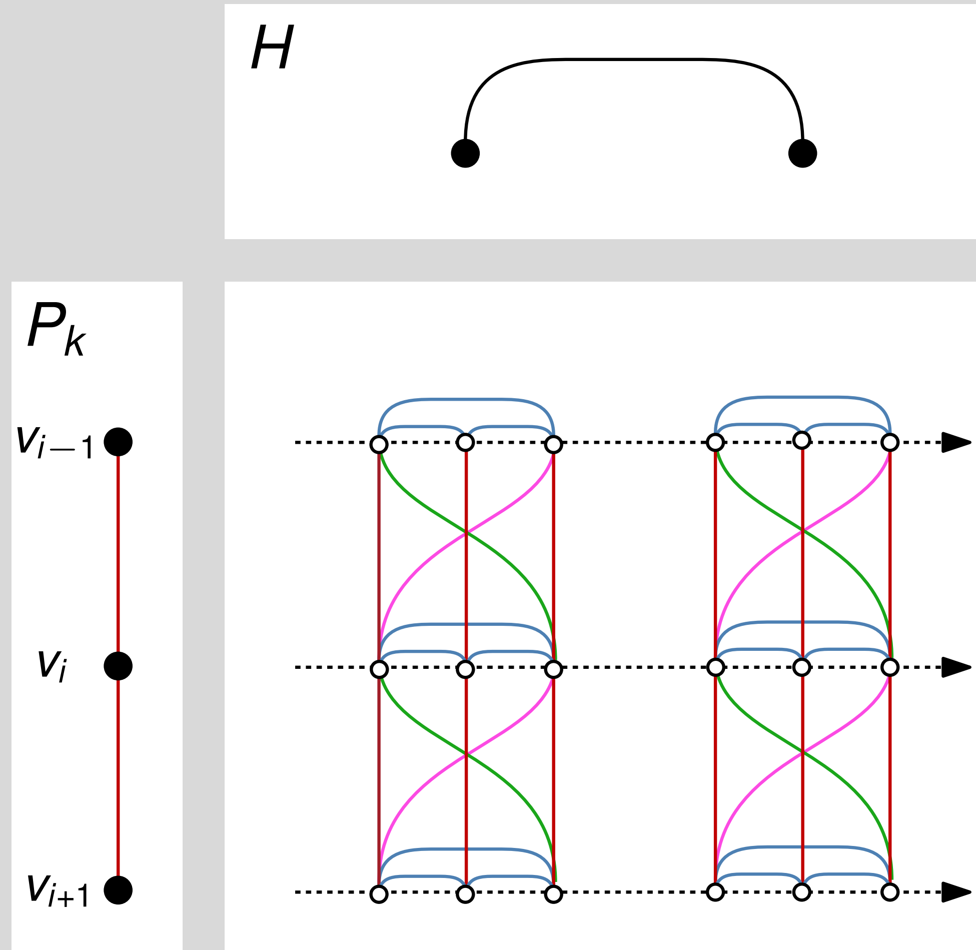
- vertex of H becomes $P_k \boxtimes K_3$
- edge becomes a $K_{3,3}$

edges on one queue of H

level	forward	backward
three	three	three

vertex

$3qn(K_3) + 1$



[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

Strong Product of planar graphs

Theorem[1]:

G planar subgraph of $P_k \boxtimes H \boxtimes K_3$

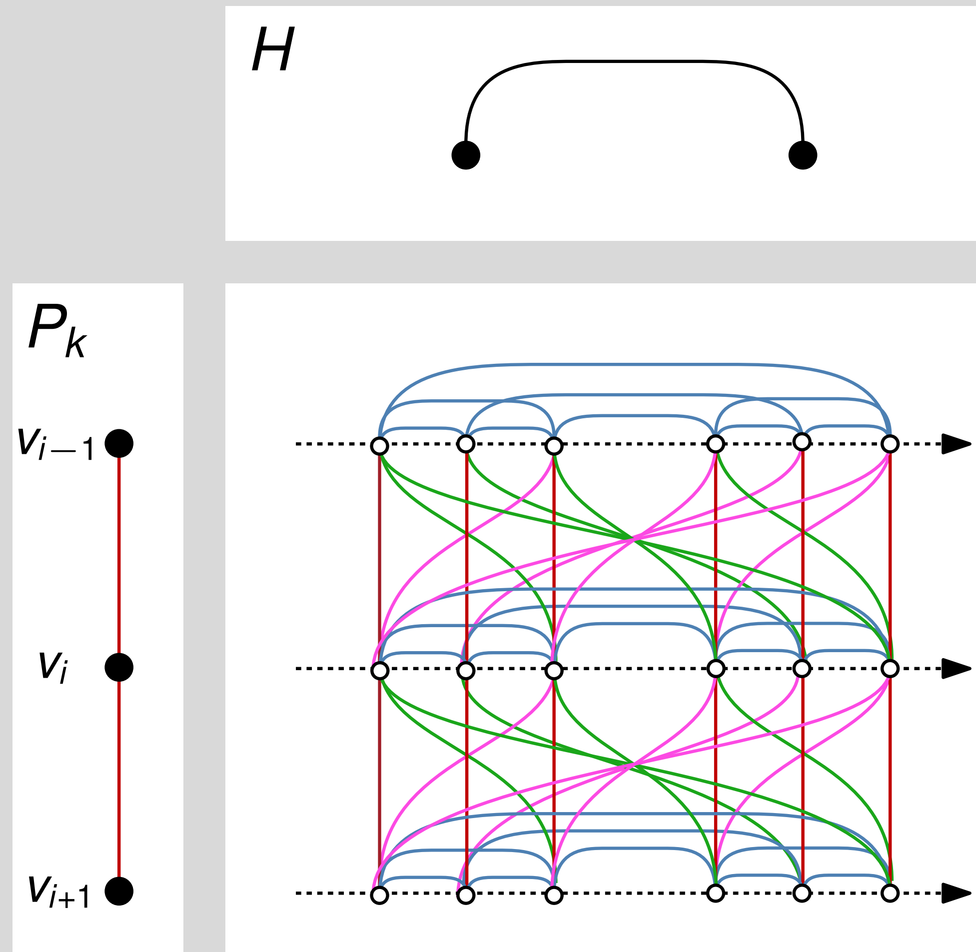
- vertex of H becomes $P_k \boxtimes K_3$
- edge becomes a $K_{3,3}$

edges on one queue of H

level	forward	backward
three	three	three

vertex: $3\text{qn}(K_3) + 1$

$3 \cdot 3 \cdot \text{qn}(H) + 4$



[1] Dujmović, Joret, Micek, Morin, Ueckerdt, Wood
Planar graphs have bounded queue-number (2020)

Step 1: compute graph H

Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49 \qquad (\mathrm{qn}(H) \leq 5)$$

Step 1: compute graph H

Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$$\text{qn}(G) \leq 3 \cdot 3 \cdot \text{qn}(H) + 4 \leq 49 \qquad (\text{qn}(H) \leq 5)$$

- BFS-layering

Step 1: compute graph H

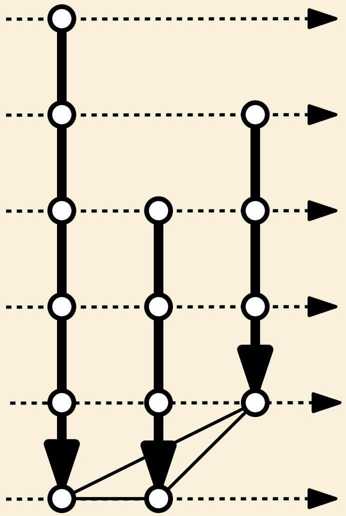
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$$\text{qn}(G) \leq 3 \cdot 3 \cdot \text{qn}(H) + 4 \leq 49 \qquad (\text{qn}(H) \leq 5)$$

- BFS-layering

Definition: Tripod



subgraph of $P \boxtimes K_3$

Step 1: compute graph H

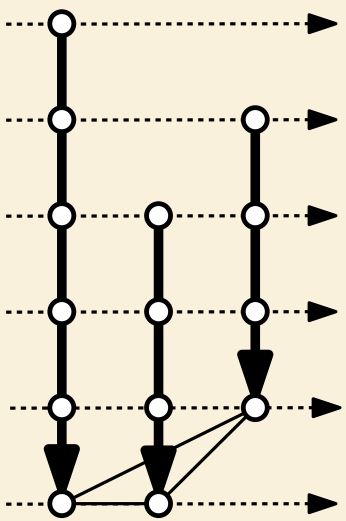
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49 \qquad (\mathrm{qn}(H) \leq 5)$$

- BFS-layering

Definition: Tripod



subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods

Step 1: compute graph H

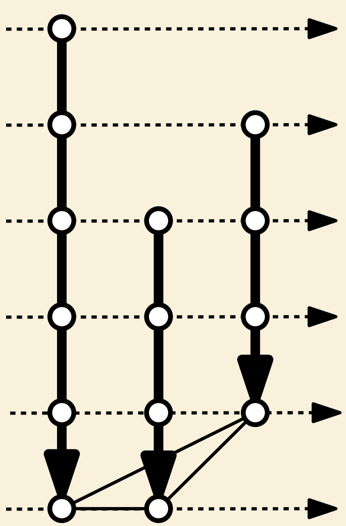
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49$ $\quad(\mathrm{qn}(H) \leq 5)$

- BFS-layering

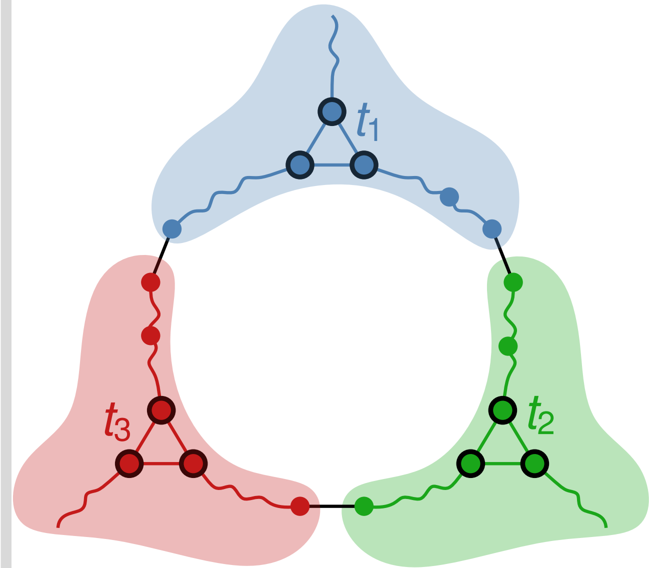
Definition: Tripod



subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods
- outerface bounded by t_1, t_2, t_3



Step 1: compute graph H

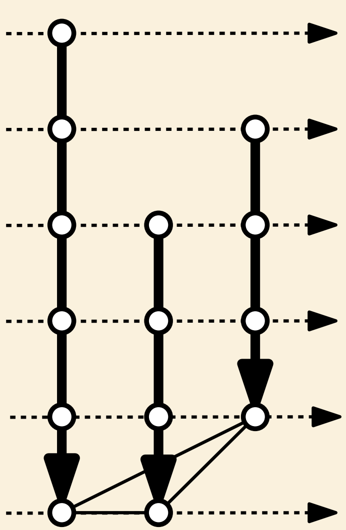
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49$ $\quad (\mathrm{qn}(H) \leq 5)$

- BFS-layering

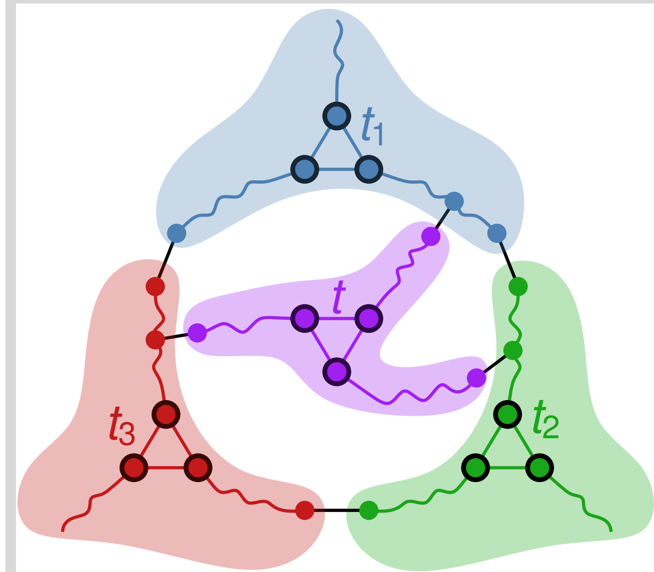
Definition: Tripod



subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods
- outerface bounded by t_1, t_2, t_3
- t attached to t_1, t_2, t_3



Step 1: compute graph H

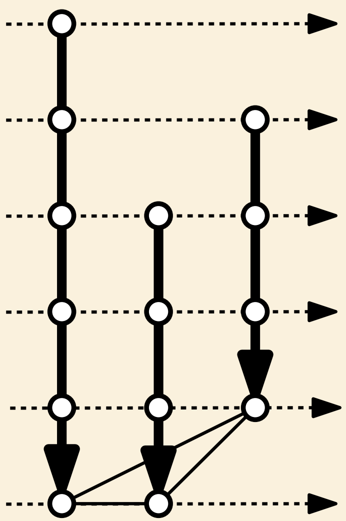
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49$ $(\mathrm{qn}(H) \leq 5)$

- BFS-layering

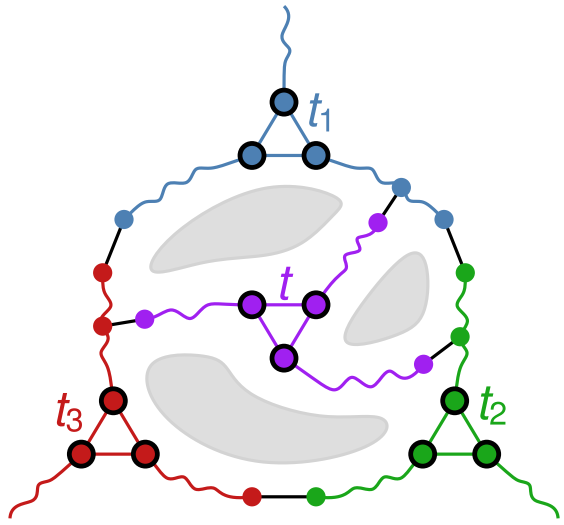
Definition: Tripod



subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods
- outerface bounded by t_1, t_2, t_3
- t attached to t_1, t_2, t_3
- recurse in regions



Step 1: compute graph H

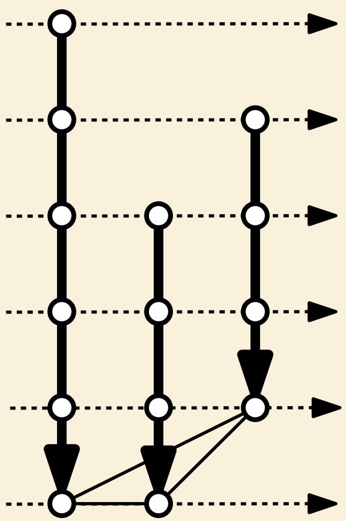
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49$ $(\mathrm{qn}(H) \leq 5)$

- BFS-layering

Definition: Tripod

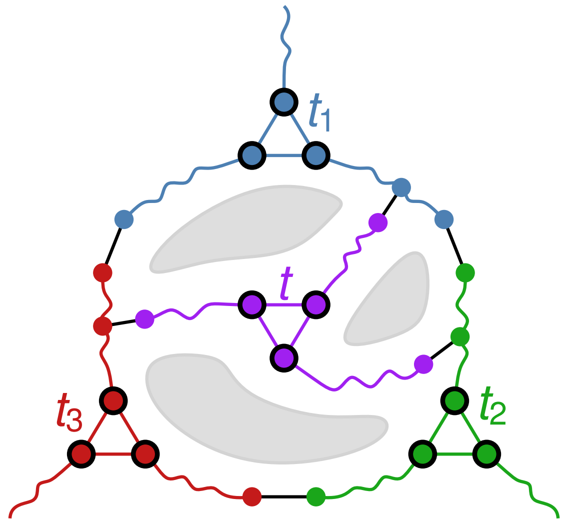


subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods
- outerface bounded by t_1, t_2, t_3
- t attached to t_1, t_2, t_3
- recurse in regions

- t_1, t_2, t_3 parents of t



Step 1: compute graph H

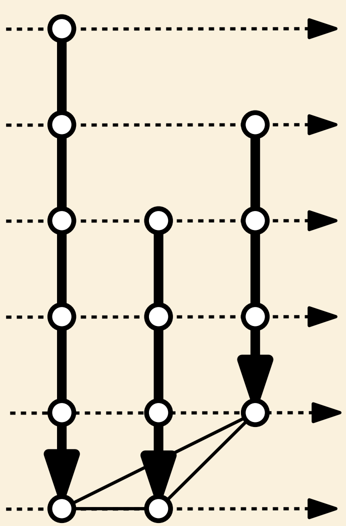
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49$ $(\mathrm{qn}(H) \leq 5)$

- BFS-layering

Definition: Tripod

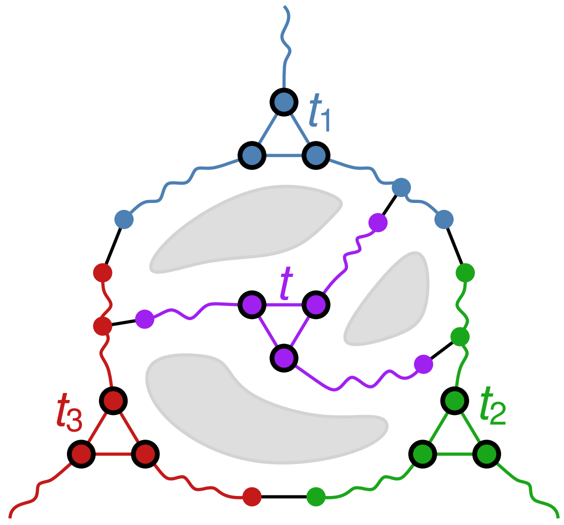


subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods
- outerface bounded by t_1, t_2, t_3
- t attached to t_1, t_2, t_3
- recurse in regions

- t_1, t_2, t_3 parents of t



Graph H

Step 1: compute graph H

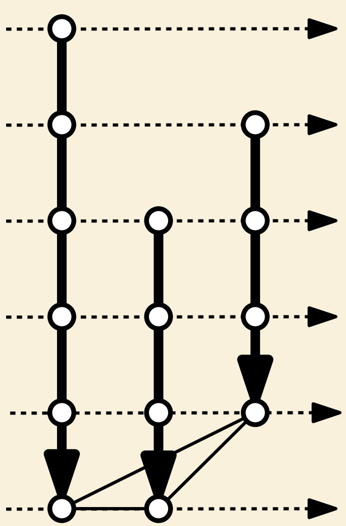
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49$ $(\mathrm{qn}(H) \leq 5)$

- BFS-layering

Definition: Tripod

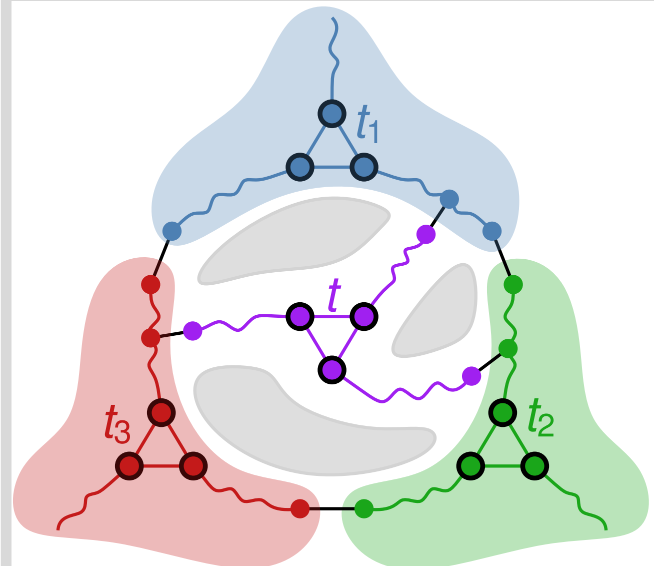


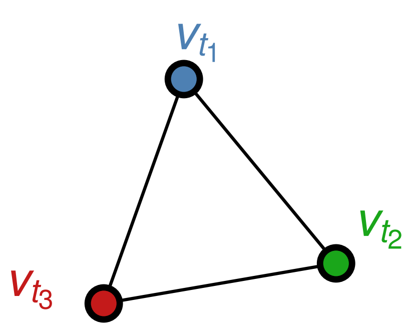
subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods
- outerface bounded by t_1, t_2, t_3
- t attached to t_1, t_2, t_3
- recurse in regions

- t_1, t_2, t_3 parents of t





Graph H

- vertex v_t corresponds to tripod t

Step 1: compute graph H

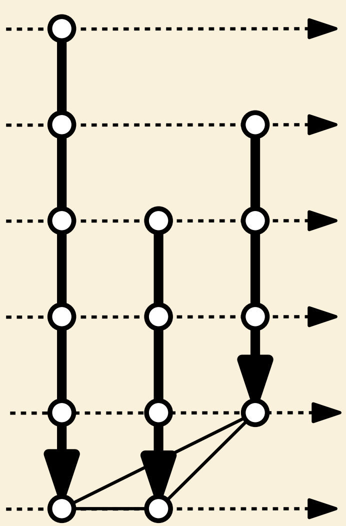
Theorem [Dujmović et al.]:

Planar graph G subgraph of $P \boxtimes H \boxtimes K_3$ where H is a planar 3-tree

$\mathrm{qn}(G) \leq 3 \cdot 3 \cdot \mathrm{qn}(H) + 4 \leq 49$ $\qquad$ $(\mathrm{qn}(H) \leq 5)$

- BFS-layering

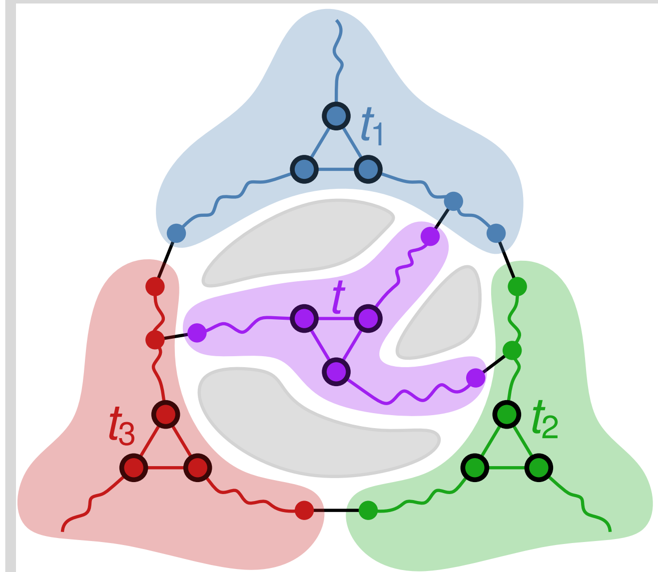
Definition: Tripod



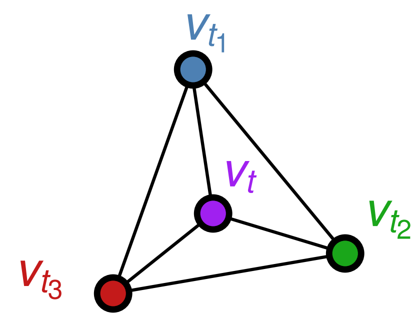
subgraph of $P \boxtimes K_3$

Tripod decomposition

- vertex partition in tripods
- outerface bounded by t_1, t_2, t_3
- t attached to t_1, t_2, t_3
- recurse in regions



- t_1, t_2, t_3 parents of t



Graph H

- vertex v_t corresponds to tripod t
- v_t adjacent with v_{t_1}, v_{t_2}, v_{t_3}

Step 2: 5-queue layout of H

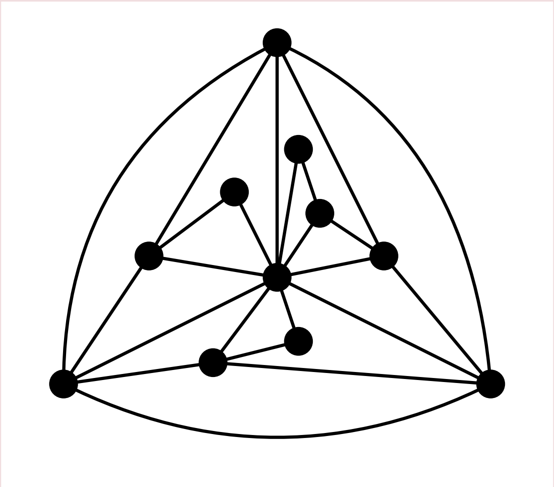
Theorem[1]:

A planar 3-tree H has $\text{qn}(H) \leq 5$

[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev:
Queue layouts of planar 3-trees (2020)

Step 2: 5-queue layout of H

Theorem[1]:
A planar 3-tree H has $\mathrm{qn}(H) \leq 5$



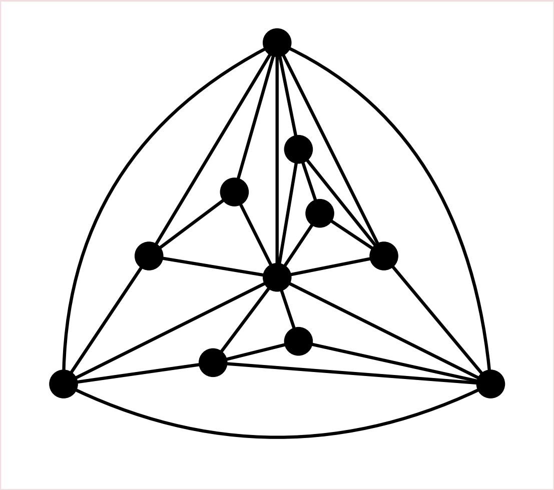
[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev:
Queue layouts of planar 3-trees (2020)

Step 2: 5-queue layout of H

Theorem[1]:

A planar 3-tree H has $\text{qn}(H) \leq 5$

- maximal planar 3-tree

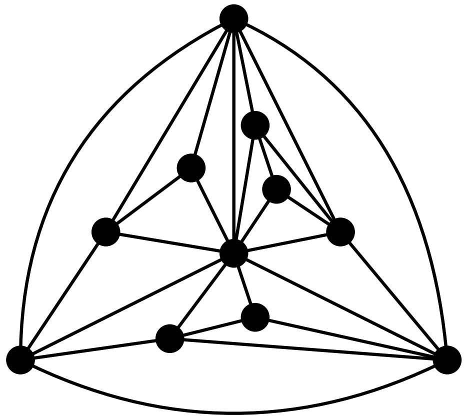


[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev: *Queue layouts of planar 3-trees* (2020)

Step 2: 5-queue layout of H

Theorem[1]:
A planar 3-tree H has $\text{qn}(H) \leq 5$

- maximal planar 3-tree
- leveling of vertices



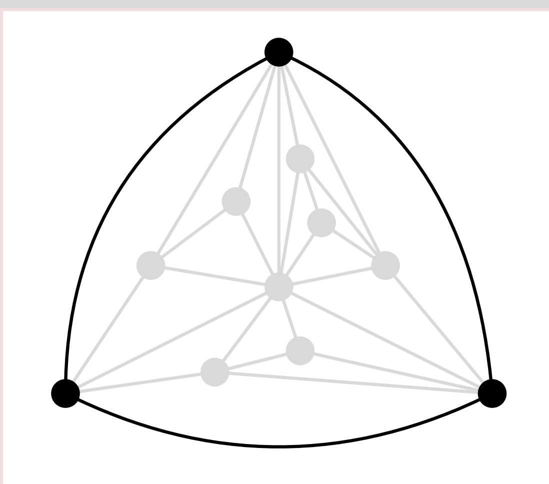
[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev:
Queue layouts of planar 3-trees (2020)

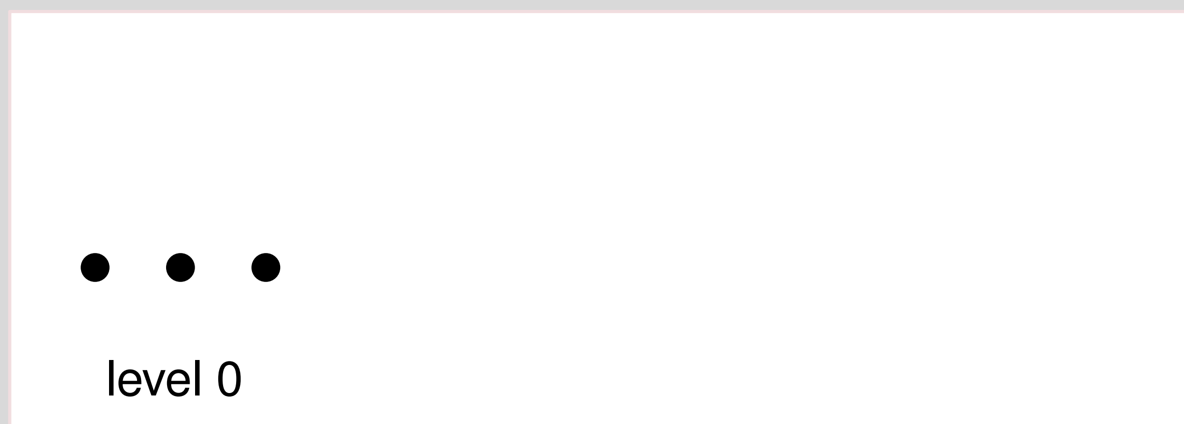
Step 2: 5-queue layout of H

Theorem[1]:

A planar 3-tree H has $\mathrm{qn}(H) \leq 5$

- maximal planar 3-tree
- leveling of vertices
- level-0 graph is triangle



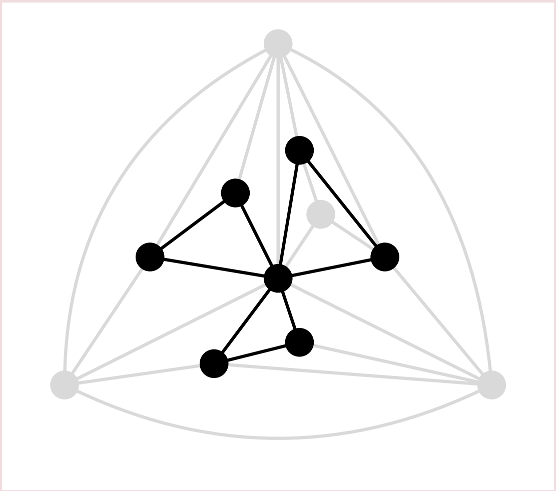


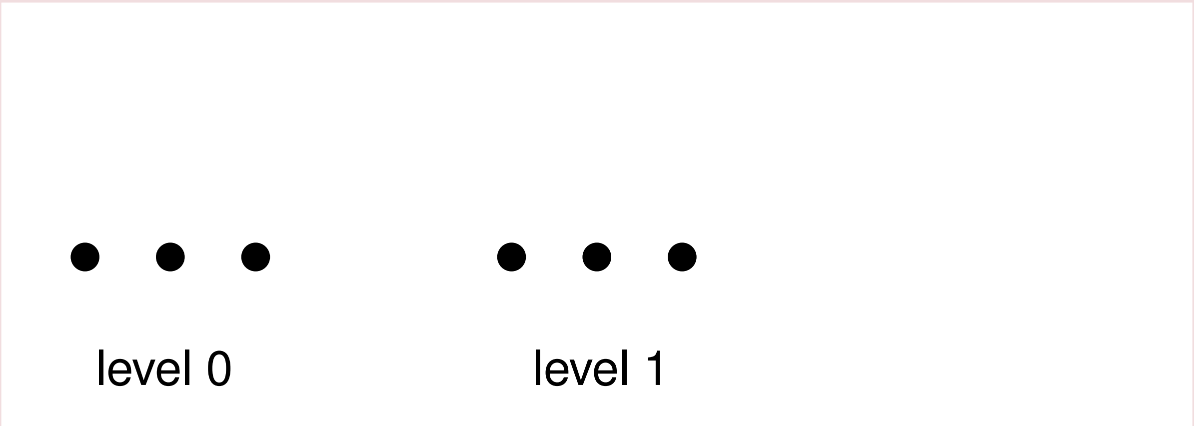
[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev:
Queue layouts of planar 3-trees (2020)

Step 2: 5-queue layout of H

Theorem[1]:
A planar 3-tree H has $\mathrm{qn}(H) \leq 5$

- maximal planar 3-tree
- leveling of vertices
- level-0 graph is triangle
- level 1 is outerplanar



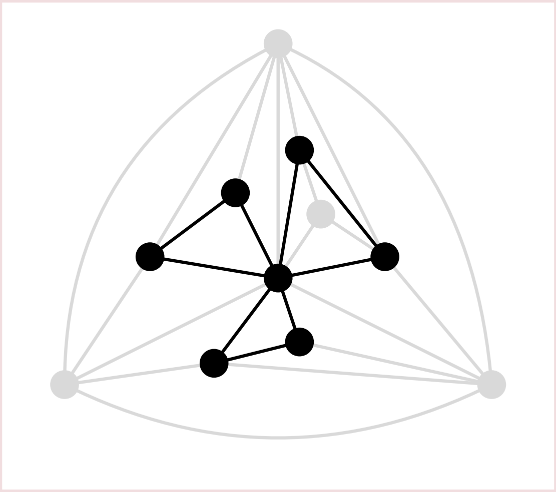


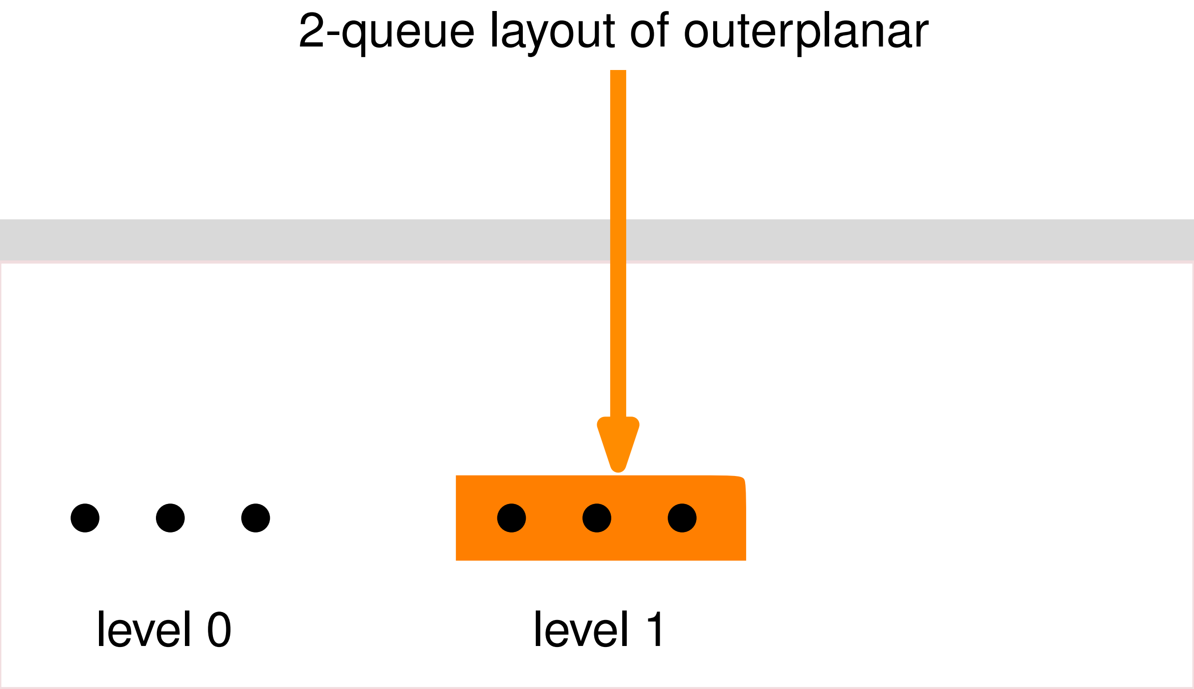
[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev: *Queue layouts of planar 3-trees* (2020)

Step 2: 5-queue layout of H

Theorem[1]:
A planar 3-tree H has $\mathrm{qn}(H) \leq 5$

- maximal planar 3-tree
- leveling of vertices
- level-0 graph is triangle
- level 1 is outerplanar





[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev:
Queue layouts of planar 3-trees (2020)

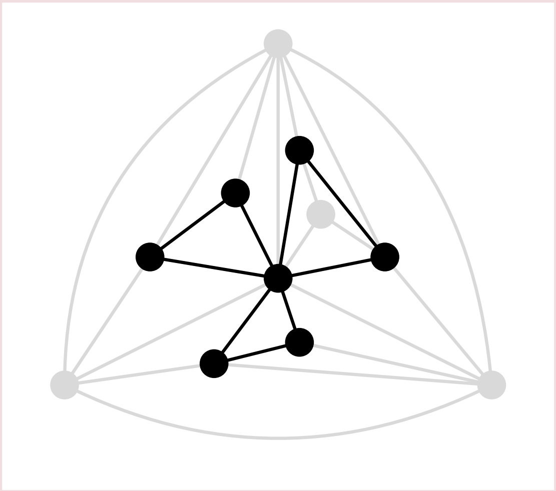
Step 2: 5-queue layout of H

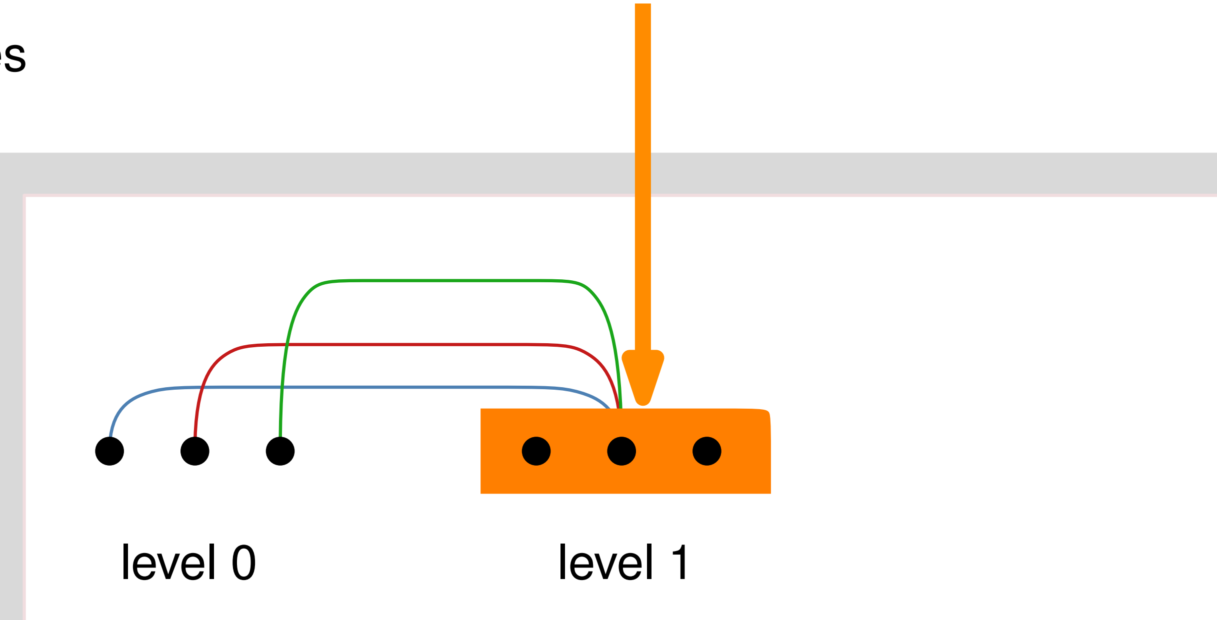
Theorem[1]:

A planar 3-tree H has $\mathrm{qn}(H) \leq 5$

- maximal planar 3-tree
- leveling of vertices
- level-0 graph is triangle
- level 1 is outerplanar
- 3 queues for binding edges

2-queue layout of outerplanar





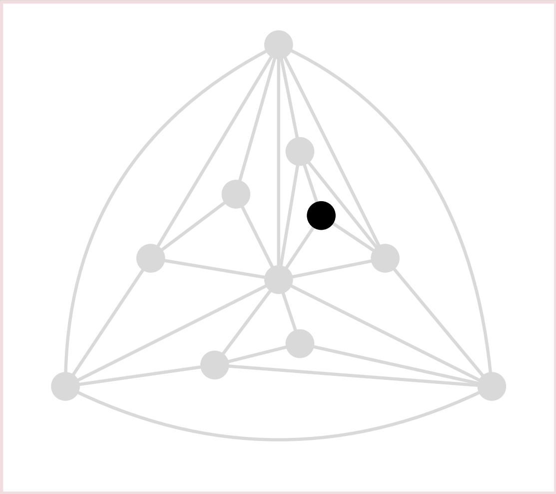
[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev:
Queue layouts of planar 3-trees (2020)

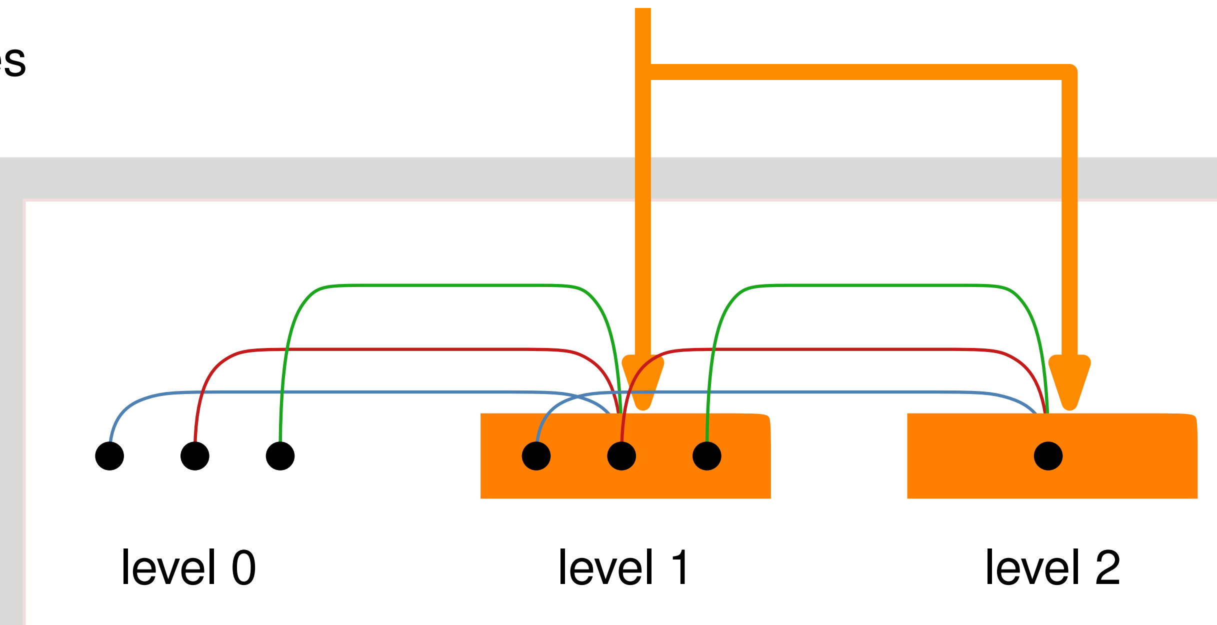
Step 2: 5-queue layout of H

Theorem[1]:
A planar 3-tree H has $\mathrm{qn}(H) \leq 5$

- maximal planar 3-tree
- leveling of vertices
- level-0 graph is triangle
- level 1 is outerplanar
- 3 queues for binding edges
- recurse in faces

2-queue layout of outerplanar





[1] Alam, Bekos, Gronemann, Kaufmann, Pupyrev: *Queue layouts of planar 3-trees* (2020)

Step 2: 2-queue layout of outerplanar

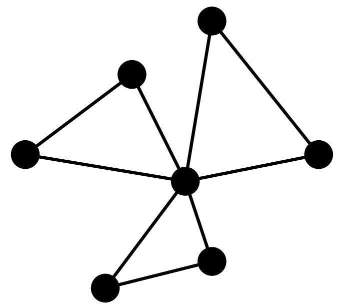
Theorem[1]:

Outerplanar graph G has $\mathrm{qn}(G) = 2$

[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\mathrm{qn}(G) = 2$

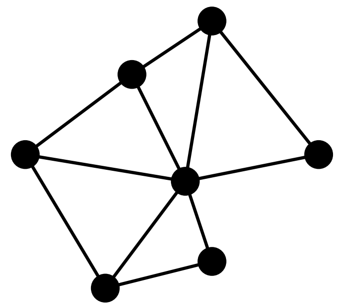


[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has qn(G) = 2

- maximal outerplanar



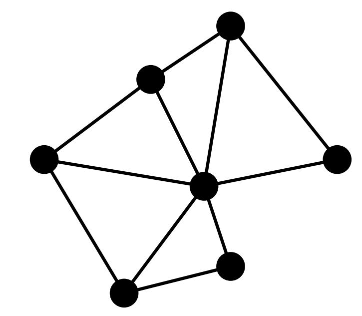
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:

Outerplanar graph G has $\mathrm{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels

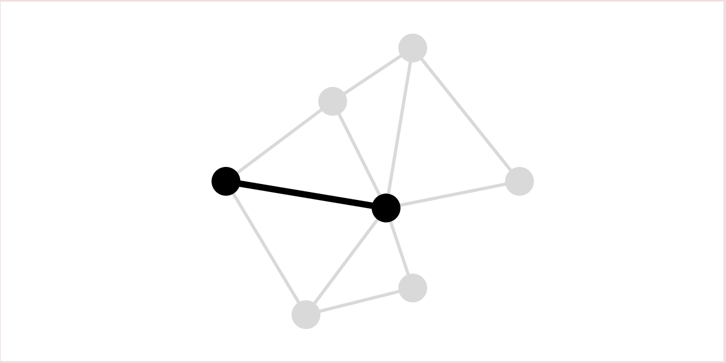


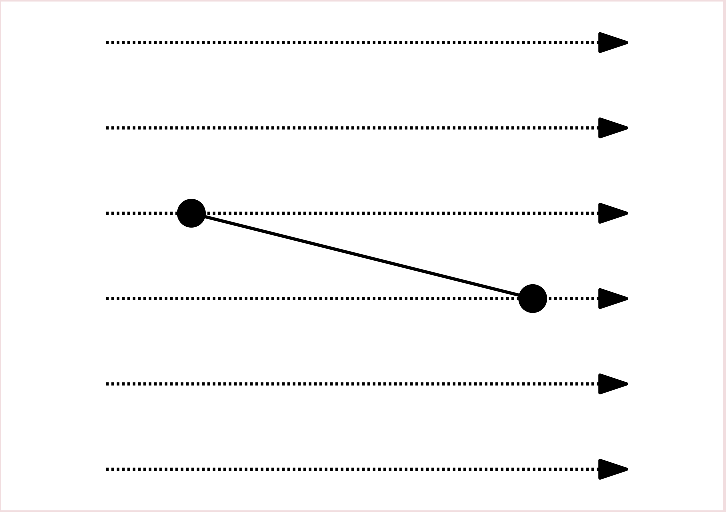
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\text{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge



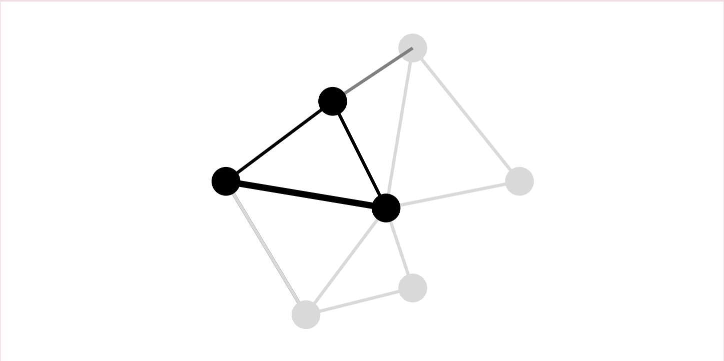


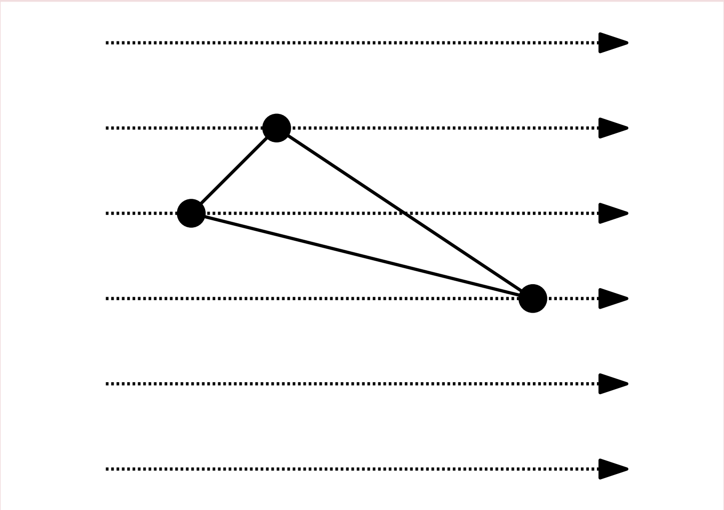
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\mathrm{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2



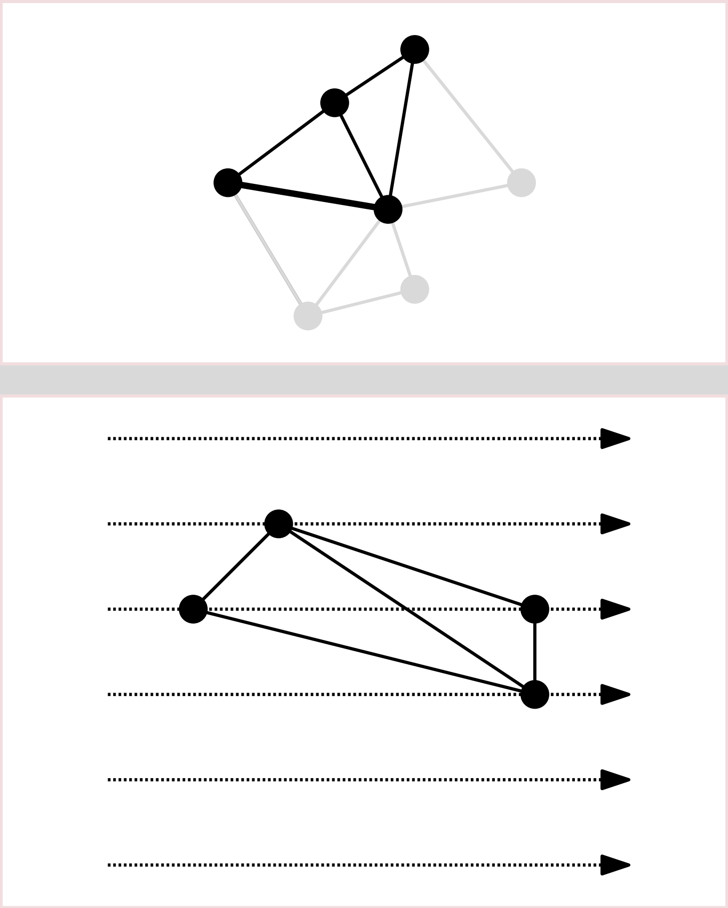


[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\mathrm{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2

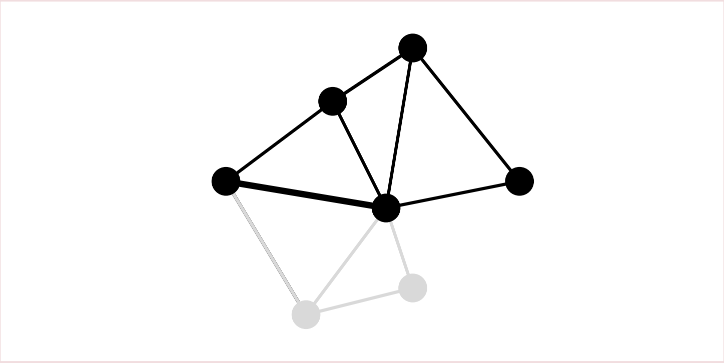


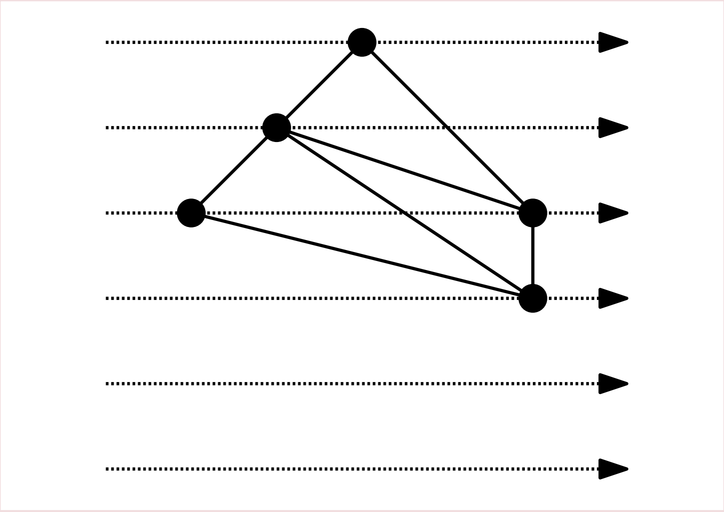
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\mathrm{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2



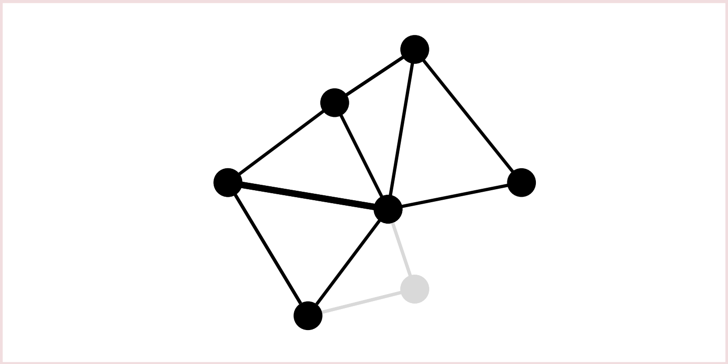


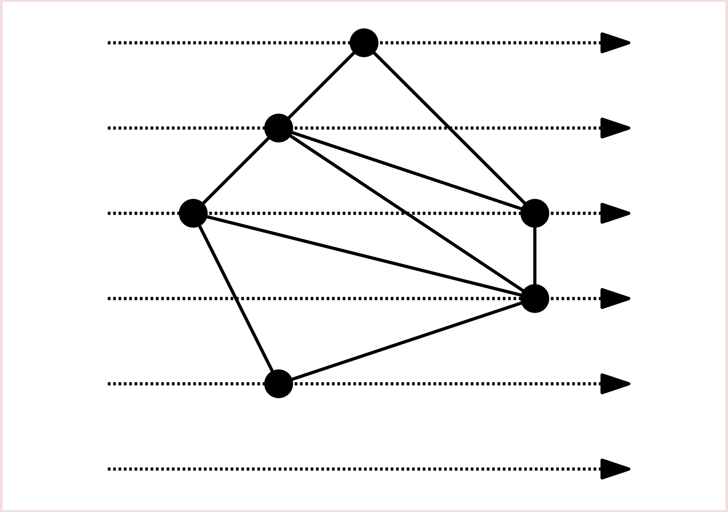
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\text{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2



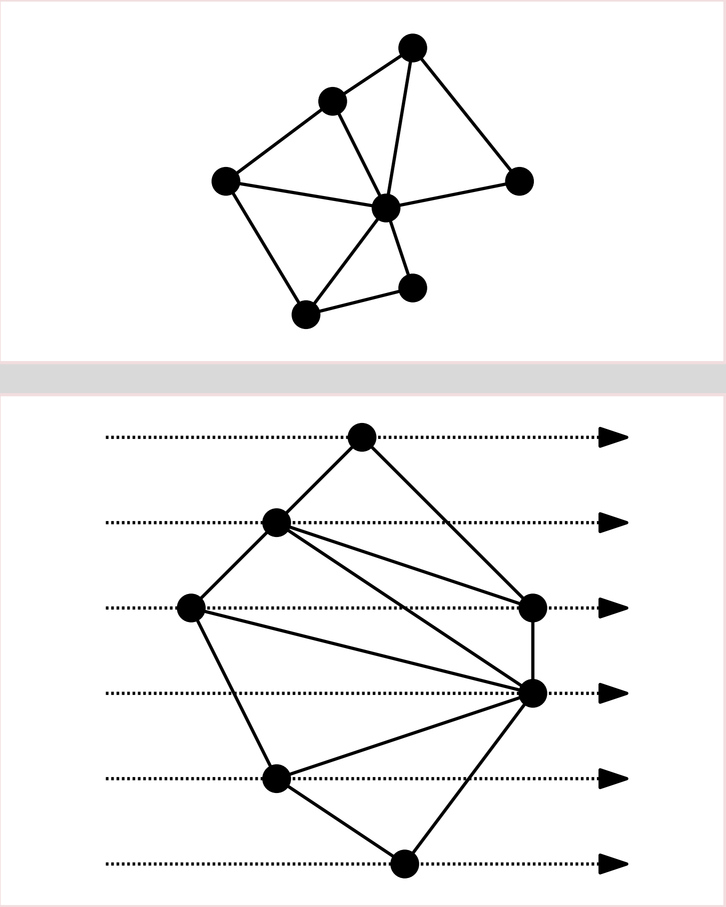


[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\mathrm{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2

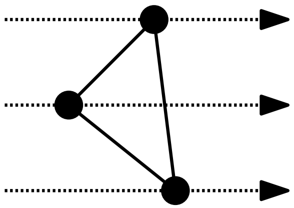


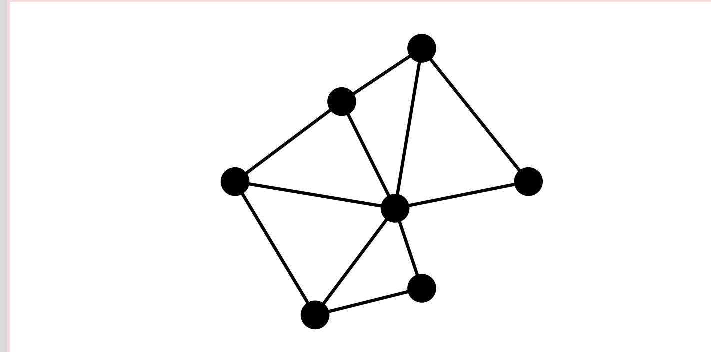
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

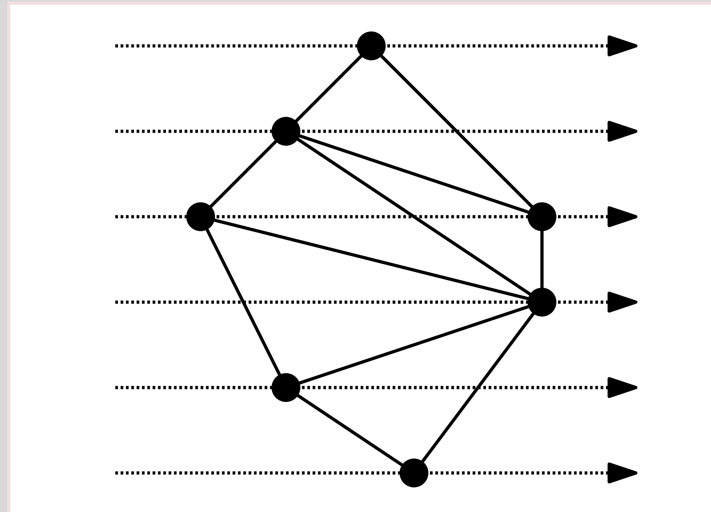
Step 2: 2-queue layout of outerplanar

Theorem[1]:
Outerplanar graph G has $\text{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2
- face has a top, middle and bottom vertex







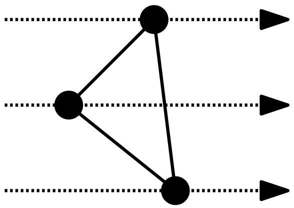
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

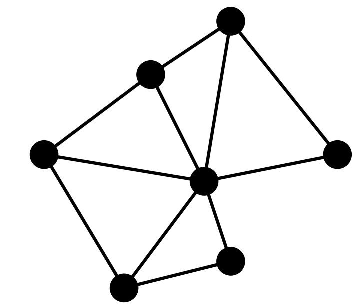
Step 2: 2-queue layout of outerplanar

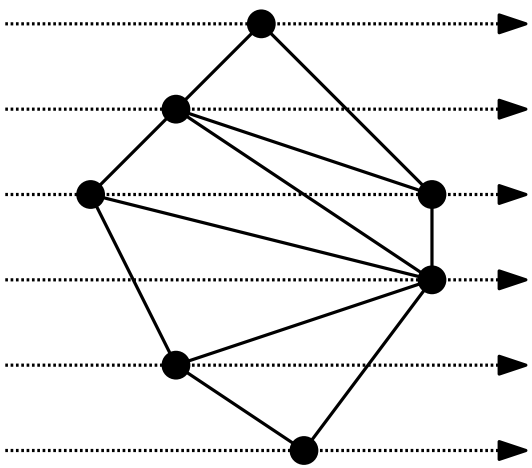
Theorem[1]:

Outerplanar graph G has $\text{qn}(G) = 2$

- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2
- face has a top, middle and bottom vertex







2-queue layout

- span 1 edges on one queue
- span 2 edges on one queue

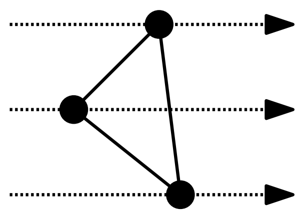
[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Step 2: 2-queue layout of outerplanar

Theorem[1]:

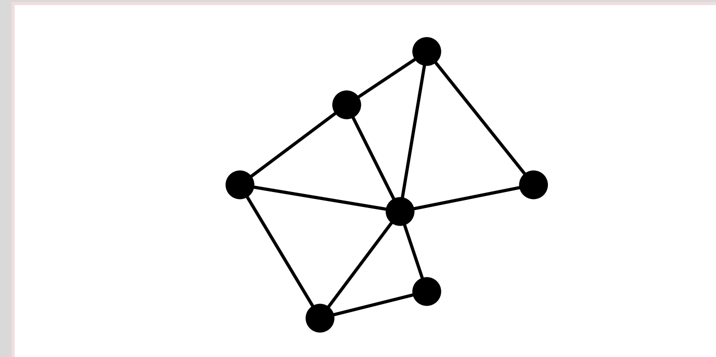
Outerplanar graph *G* has qn(*G*) = 2

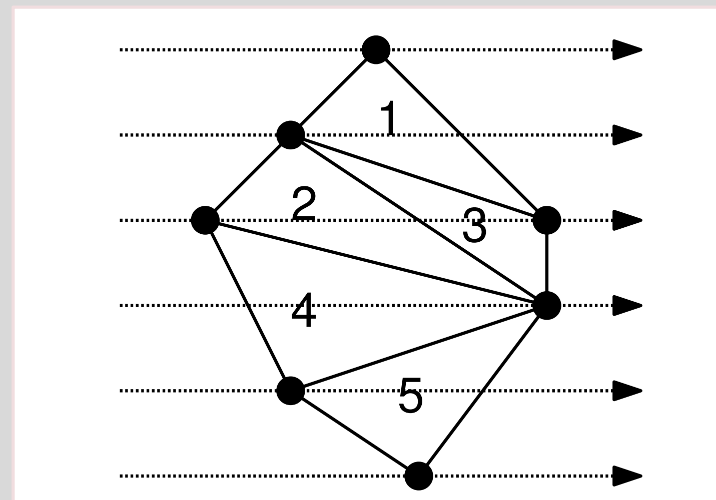
- maximal outerplanar
- vertices placed on levels
- start with an edge
- add degree-2 vertex
- edges have span 1 or 2
- face has a top, middle and bottom vertex



2-queue layout

- span 1 edges on one queue
- span 2 edges on one queue
- faces are ordered





[1] Dujmović, Pór, Wood: *Track layouts of graphs* (2004)

Overview

Theorem [Dujmović et al.]:

$$qn(G) \leq 3 \cdot 3 \cdot qn(H) + 4 \leq 49$$

Overview

Theorem [Dujmović et al.]:

$$qn(G) \leq 3 \cdot 3 \cdot qn(H) + 4 \leq 49$$

three types of edges

- level
- forward
- backward

Overview

Theorem [Dujmović et al.]:

$$qn(G) \leq 3 \cdot 3 \cdot qn(H) + 4 \leq 49$$

three types of edges

- level
- forward
- backward

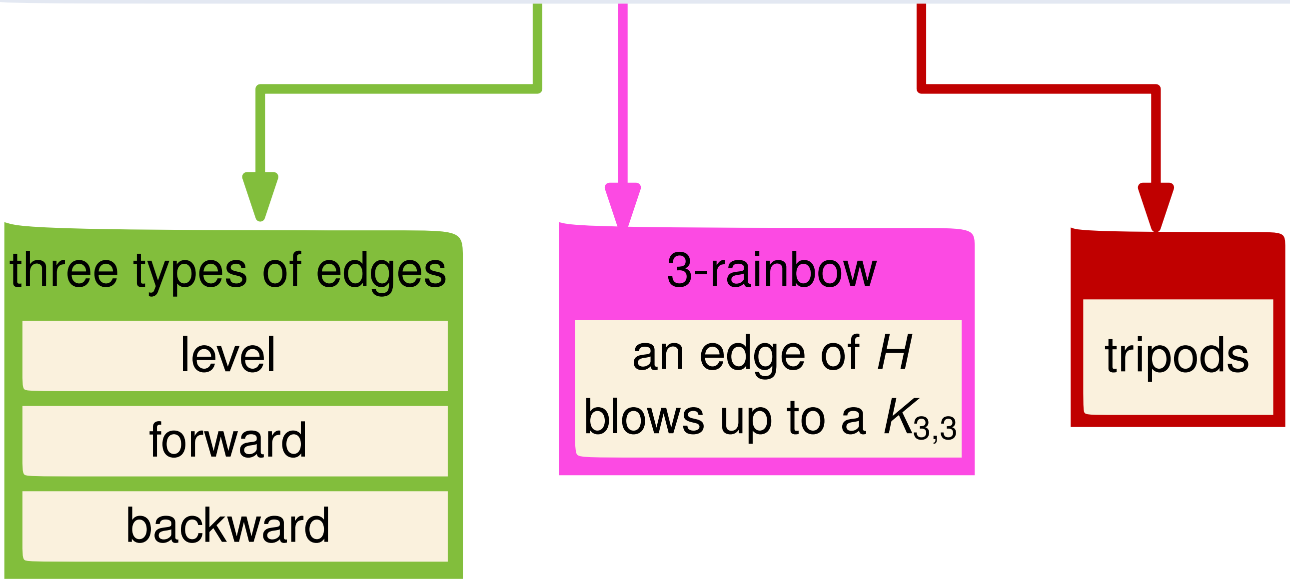
3-rainbow

an edge of H blows up to a $K_{3,3}$

Overview

Theorem [Dujmović et al.]:

$$qn(G) \leq 3 \cdot 3 \cdot qn(H) + 4 \leq 49$$



Overview

Theorem [Dujmović et al.]:

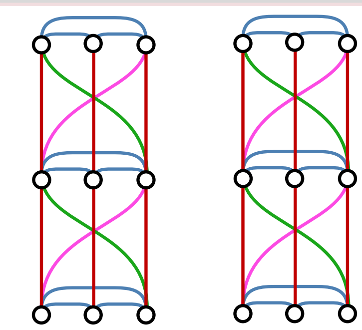
$$qn(G) \leq 3 \cdot 3 \cdot qn(H) + 4 \leq 49$$

three types of edges
- level
- forward
- backward

3-rainbow
- an edge of H blows up to a $K_{3,3}$

tripods

- 5-queue layout of H determines the order of the tripods



Overview

Theorem [Dujmović et al.]:

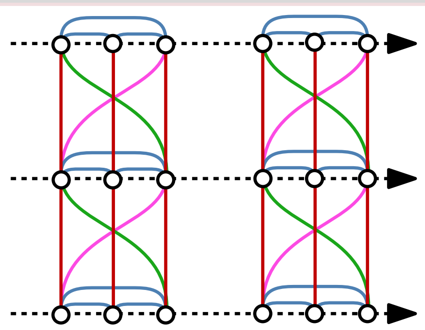
$$qn(G) \leq 3 \cdot 3 \cdot qn(H) + 4 \leq 49$$

three types of edges
- level
- forward
- backward

3-rainbow
- an edge of H blows up to a $K_{3,3}$

tripods

- 5-queue layout of H determines the order of the tripods
- BFS-leveling of G is a partial order



Overview

Theorem [Dujmović et al.]:

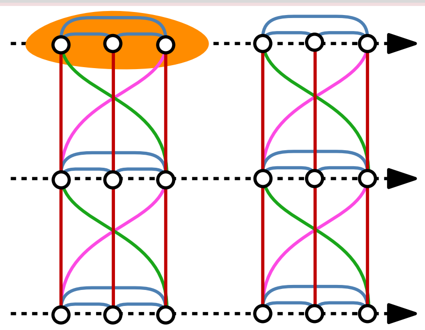
$$qn(G) \leq 3 \cdot 3 \cdot qn(H) + 4 \leq 49$$

three types of edges
- level
- forward
- backward

3-rainbow
- an edge of H blows up to a $K_{3,3}$

tripods

- 5-queue layout of H determines the order of the tripods
- BFS-leveling of G is a partial order
- each 3 vertices of the same tripod and on same BFS-level are unordered



Improvement

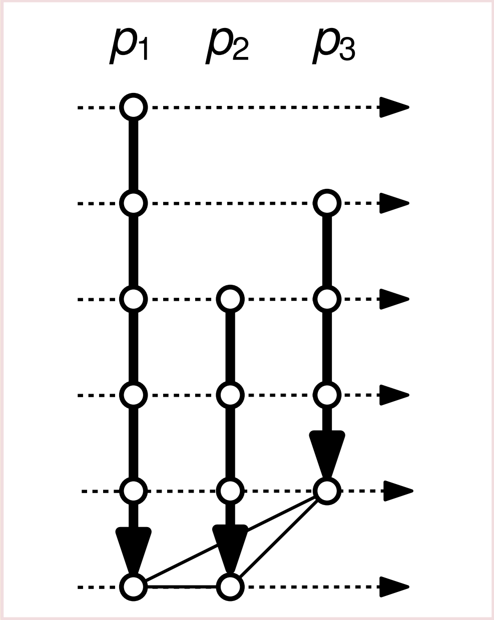
Main Idea:

order the paths of each tripod

Improvement

Main Idea:

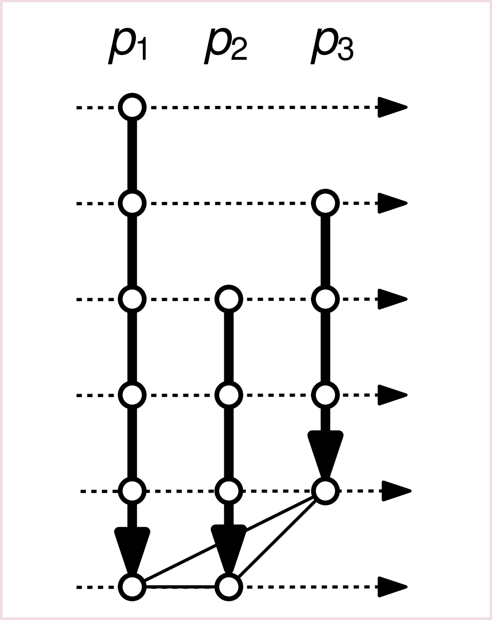
order the paths of each tripod



Improvement

Main Idea:

order the paths of each tripod

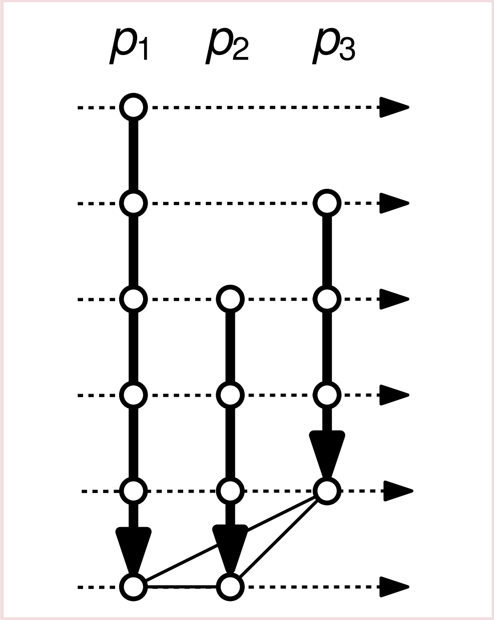


- level edges on one queue
  (level edges of $P_k \boxtimes K_3$)

Improvement

Main Idea:

order the paths of each tripod



- level edges on one queue
  (level edges of $P_k \boxtimes K_3$)

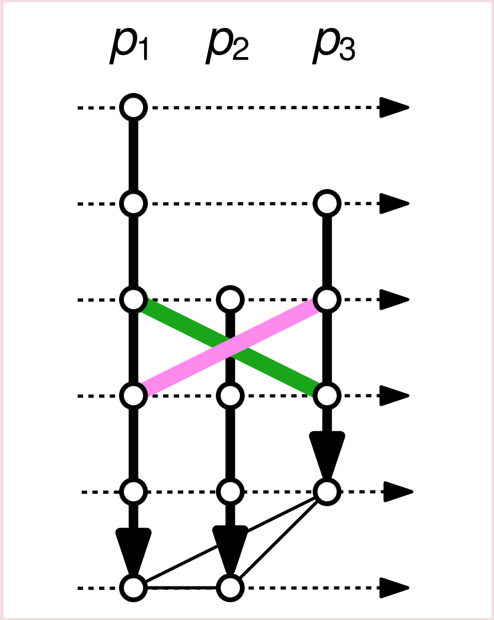
Claim

- binding (and vertex) edges
  create no 3-rainbow

Improvement

Main Idea:

order the paths of each tripod



- level edges on one queue
  (level edges of $P_k \boxtimes K_3$)

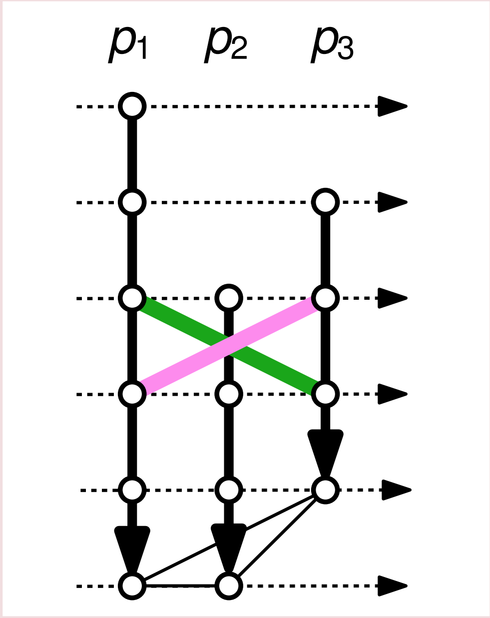
Claim

- binding (and vertex) edges
  create no 3-rainbow

Improvement

Main Idea:

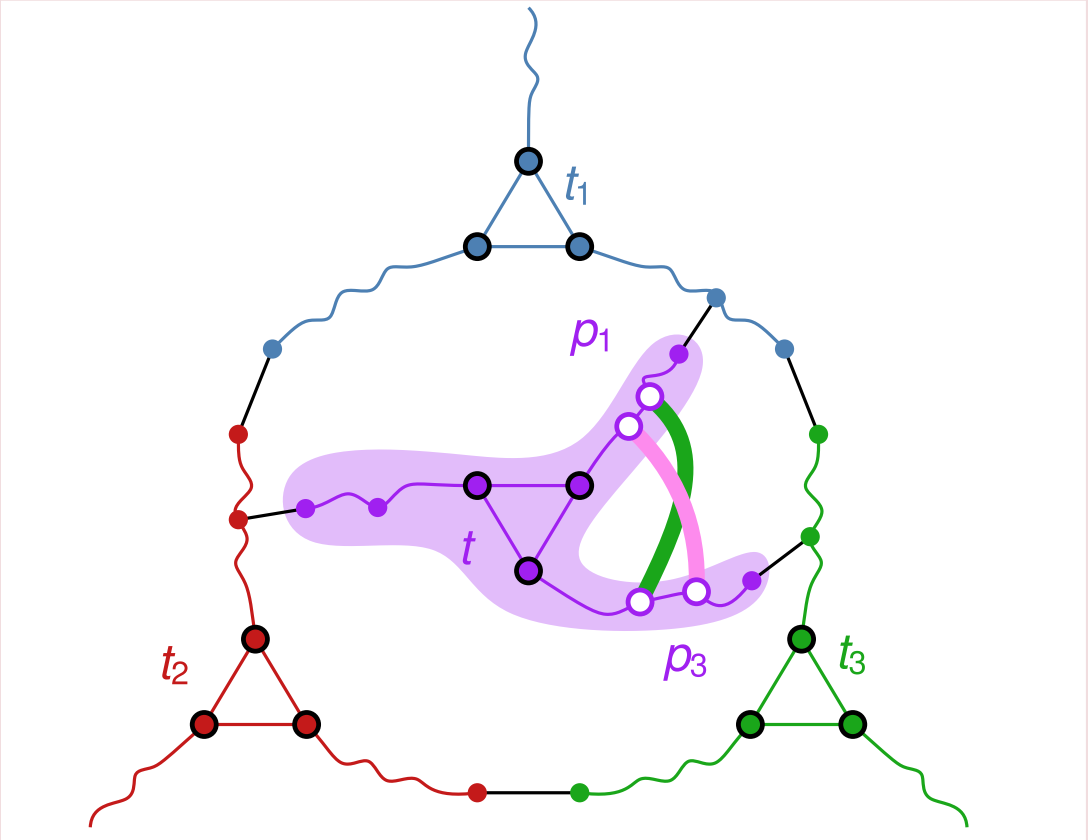
order the paths of each tripod



- level edges on one queue
  (level edges of $P_k \boxtimes K_3$)

Claim

- binding (and vertex) edges
  create no 3-rainbow

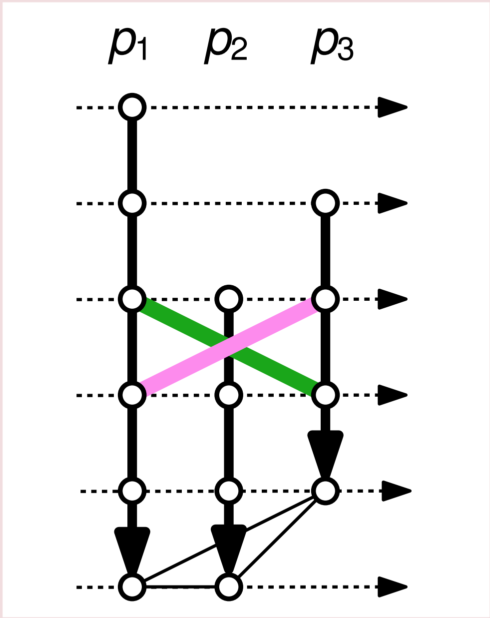


otherwise planarity is violated

Improvement

Main Idea:

order the paths of each tripod

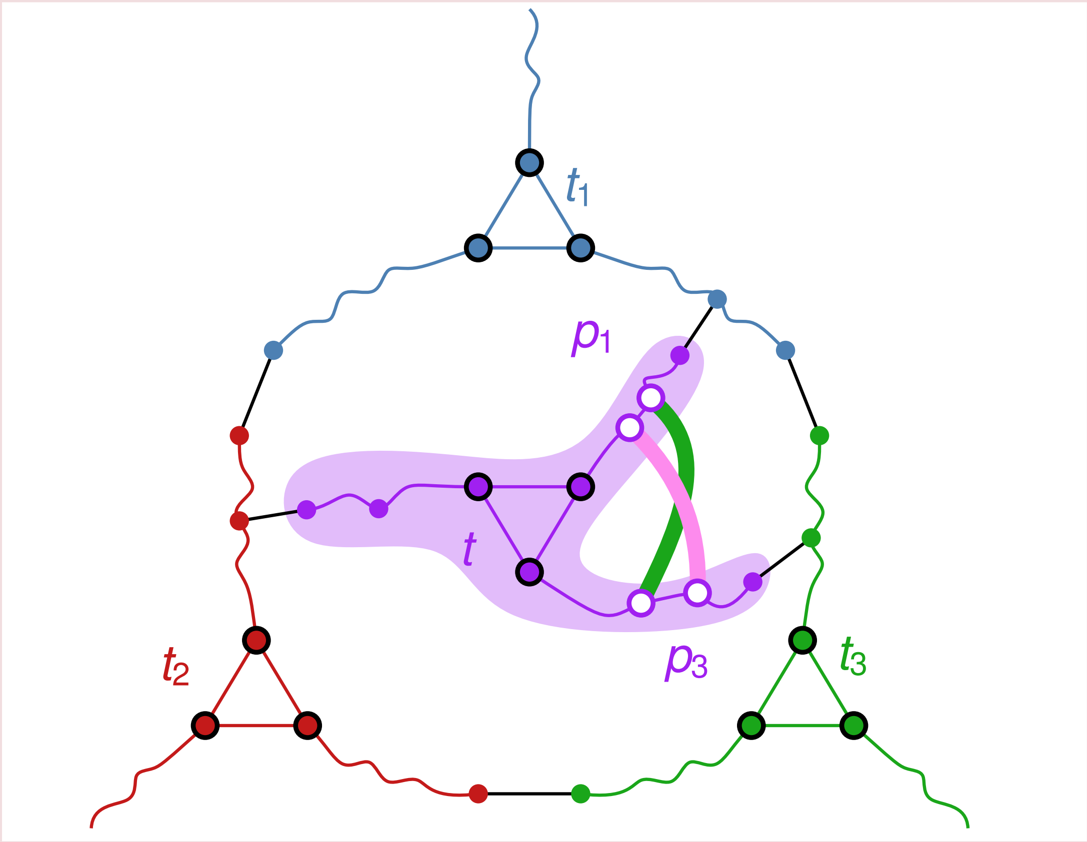


- level edges on one queue (level edges of $P_k \boxtimes K_3$)

Tripods need 3 queues upper bound reduced to 48

Claim

- binding (and vertex) edges create no 3-rainbow



otherwise planarity is violated

Improvement

Goal:

order the paths of each tripod to avoid 3-rainbows

Improvement

Goal:
order the paths of each tripod to avoid 3-rainbows

- focus on level edges on single level

Improvement

Goal:

order the paths of each tripod to avoid 3-rainbows

- focus on level edges on single level
- a 3-rainbow contains 3 vertices of the same tripod

Improvement

Goal:
order the paths of each tripod to avoid 3-rainbows

- focus on level edges on single level
- a 3-rainbow contains 3 vertices of the same tripod

Types of 3-rainbows

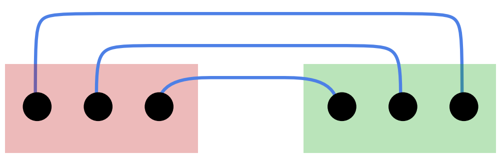
Improvement

Goal:
order the paths of each tripod to avoid 3-rainbows

- focus on level edges on single level
- a 3-rainbow contains 3 vertices of the same tripod

Types of 3-rainbows

one tripod



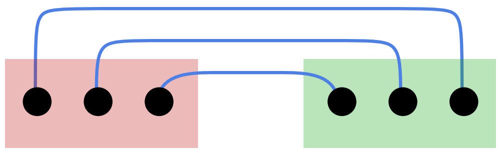
Improvement

Goal:
order the paths of each tripod to avoid 3-rainbows

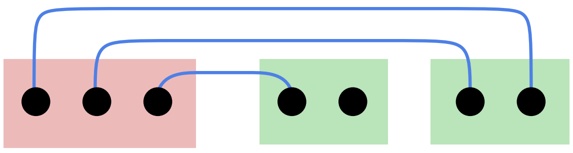
- focus on level edges on single level
- a 3-rainbow contains 3 vertices of the same tripod

Types of 3-rainbows

one tripod



two tripods



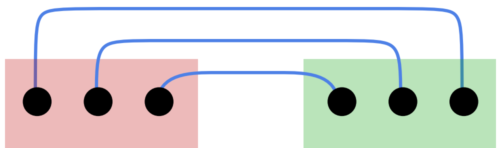
Improvement

Goal:
order the paths of each tripod to avoid 3-rainbows

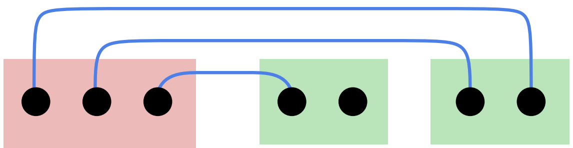
- focus on level edges on single level
- a 3-rainbow contains 3 vertices of the same tripod

Types of 3-rainbows

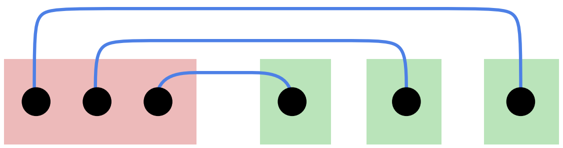
one tripod



two tripods



three tripod



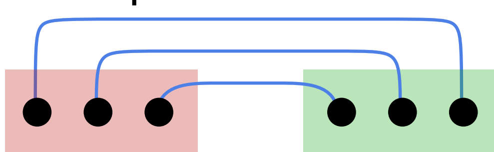
Improvement

Goal:
order the paths of each tripod to avoid 3-rainbows

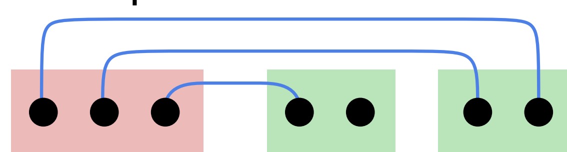
- focus on level edges on single level
- a 3-rainbow contains 3 vertices of the same tripod

Types of 3-rainbows

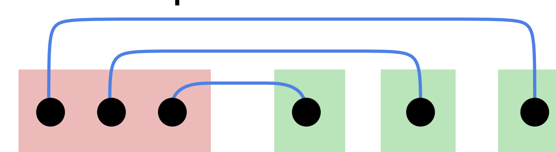
one tripod

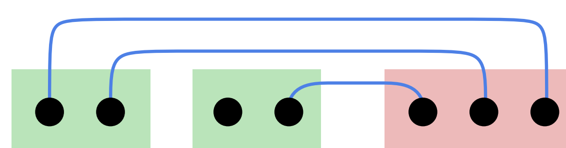


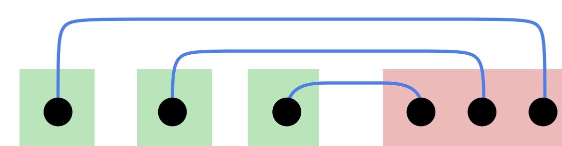
two tripods



three tripod

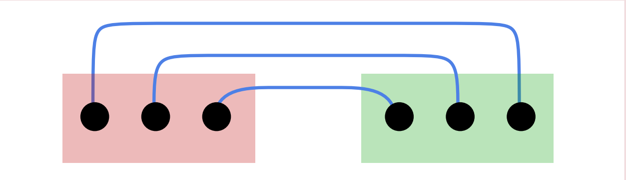


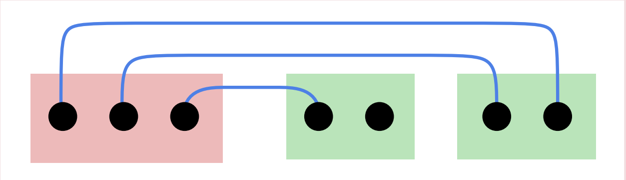


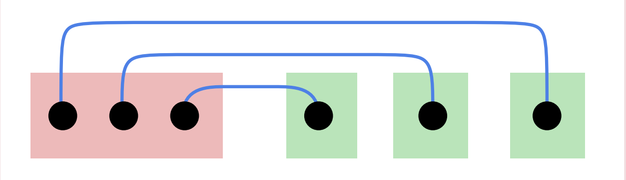


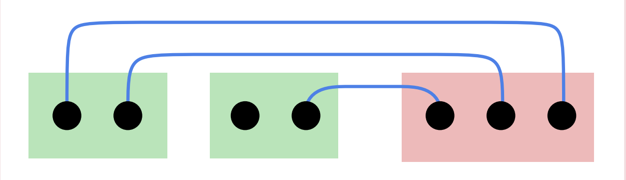
Improvement

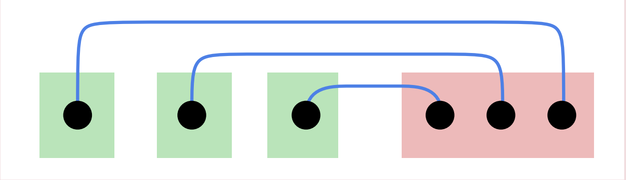
Types of 3-rainbows





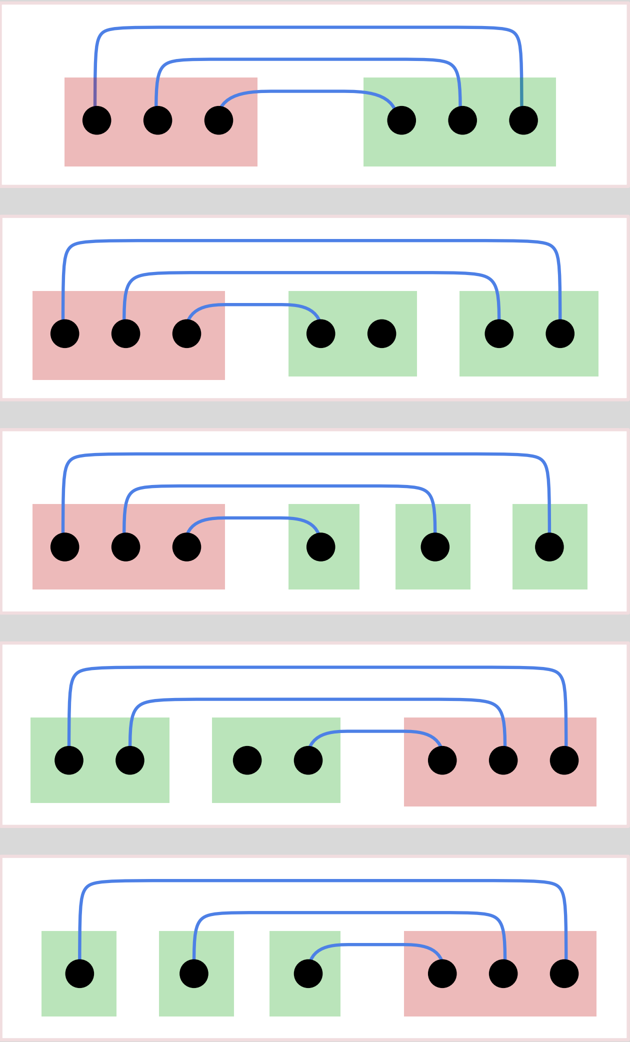






Improvement

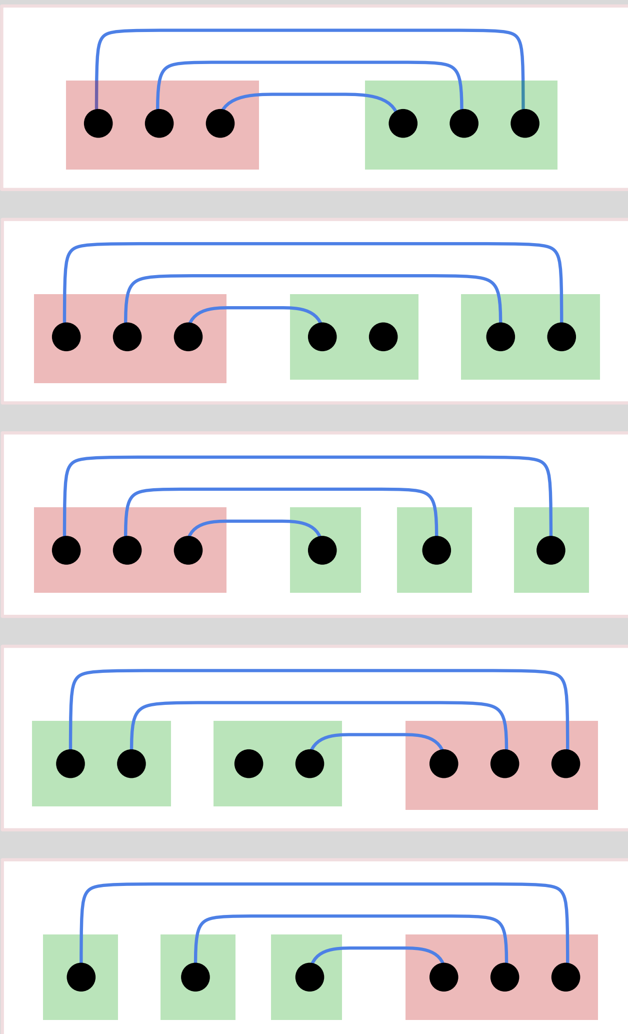
Types of 3-rainbows



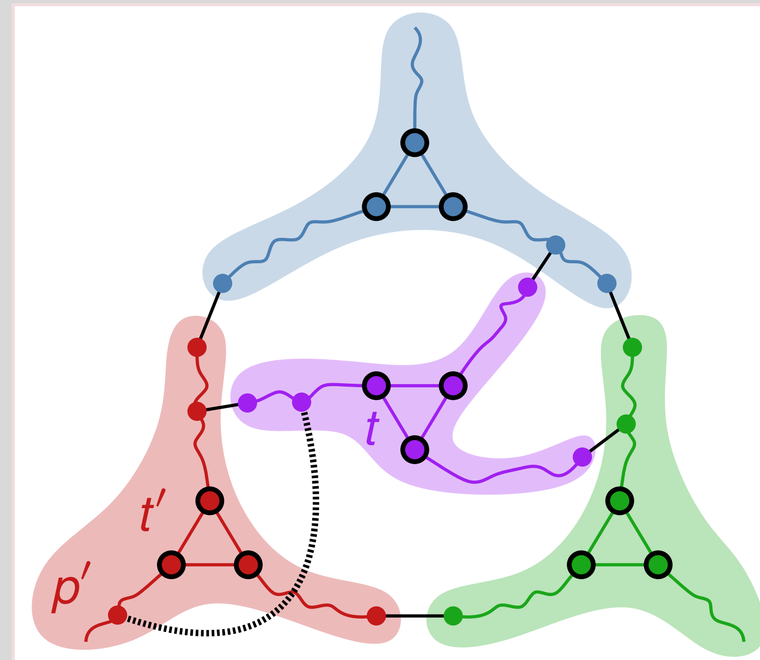
- for two adjacent tripods one is parent of the other

Improvement

Types of 3-rainbows



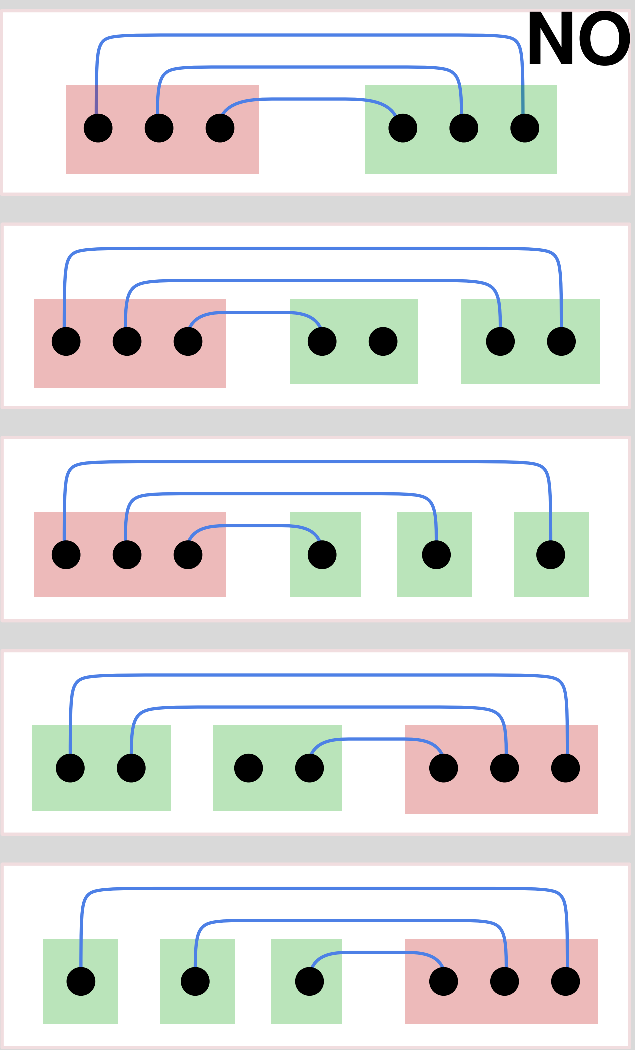
- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'



otherwise planarity is violated

Improvement

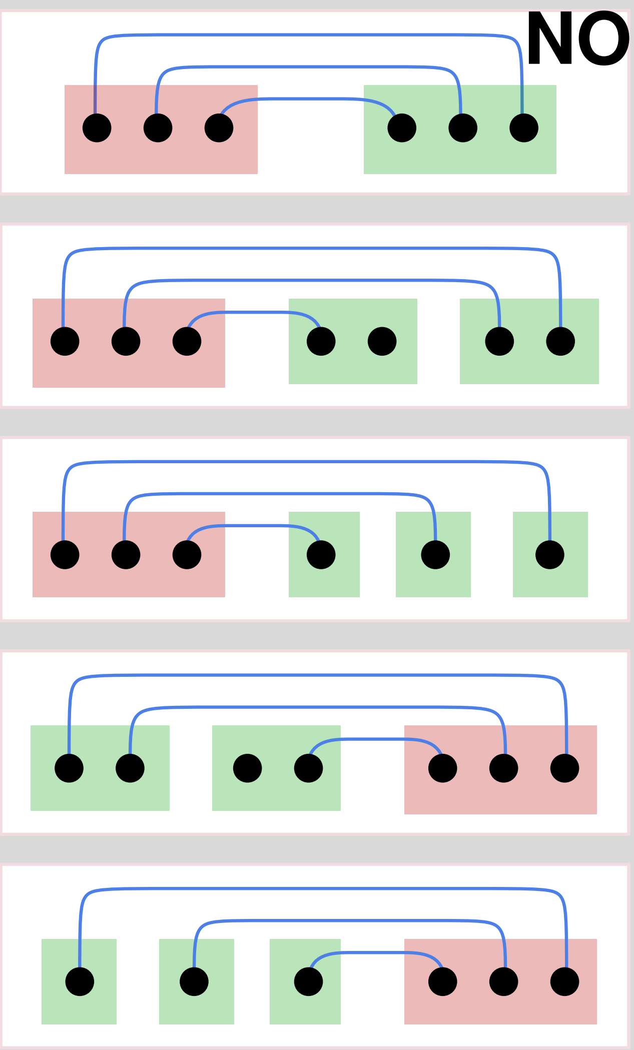
Types of 3-rainbows



- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'

Improvement

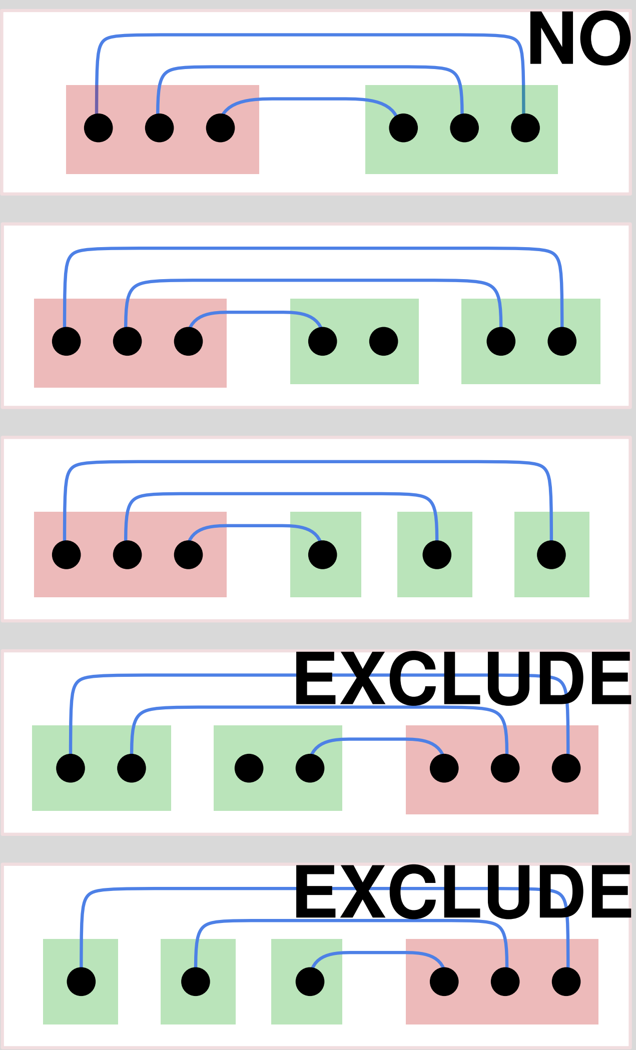
Types of 3-rainbows



- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'
- left and right might conflict

Improvement

Types of 3-rainbows

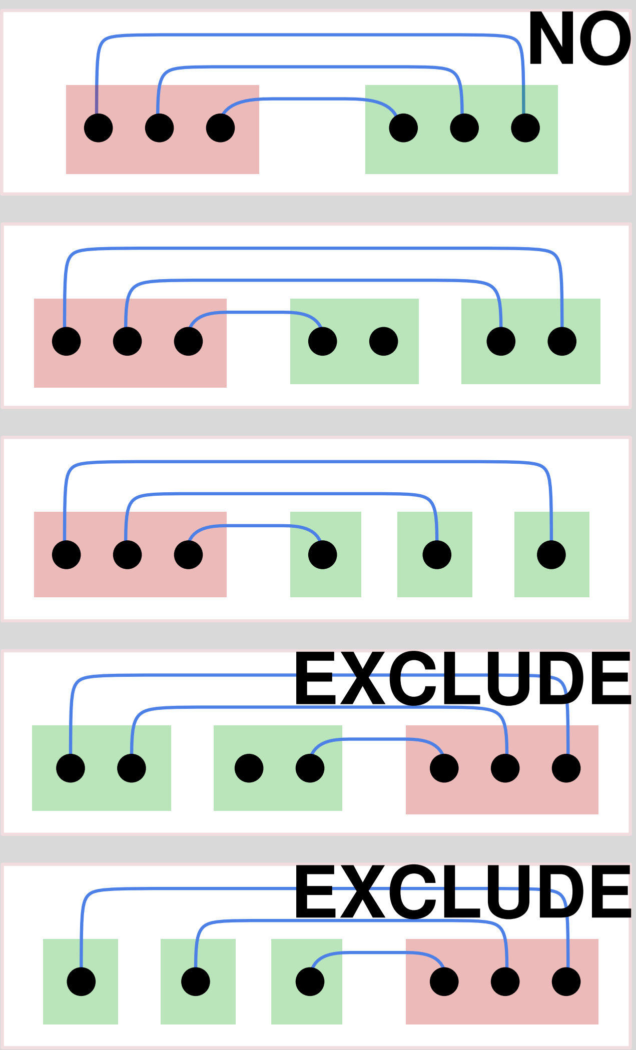


- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'
- left and right might conflict

$\rightarrow$ consider only following tripods

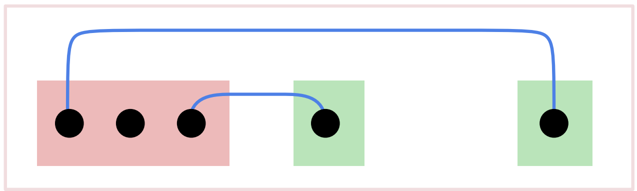
Improvement

Types of 3-rainbows



- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'
- left and right might conflict

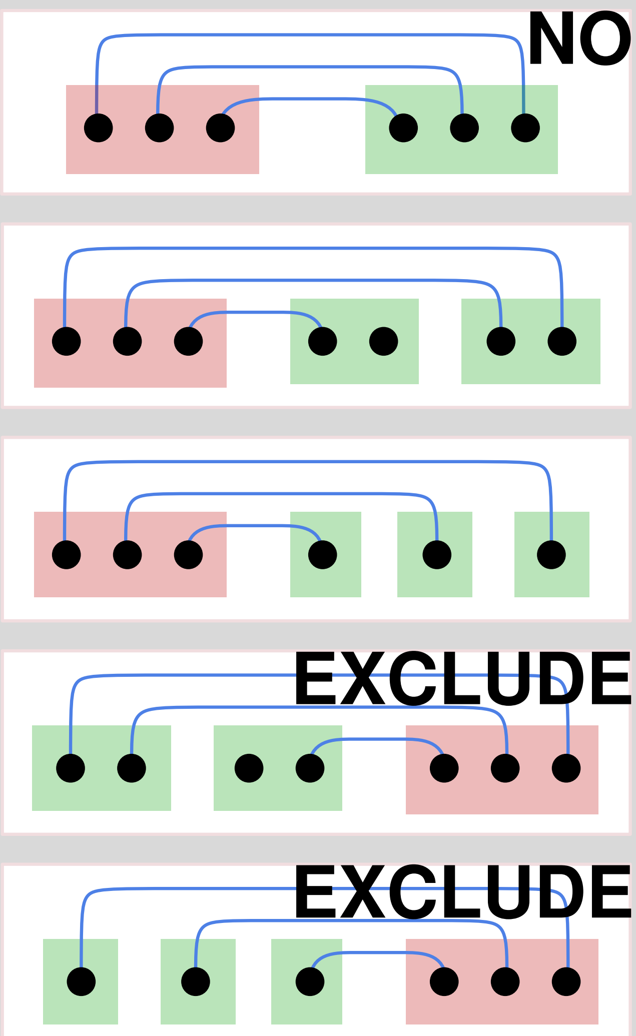
$\rightarrow$ consider only following tripods



- avoid one of the two edges

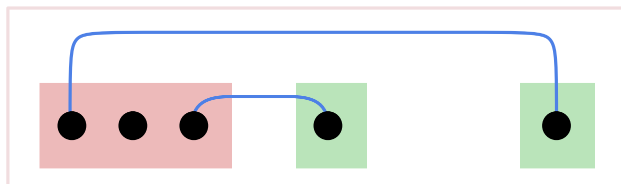
Improvement

Types of 3-rainbows



- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'
- left and right might conflict

$\rightarrow$ consider only following tripods

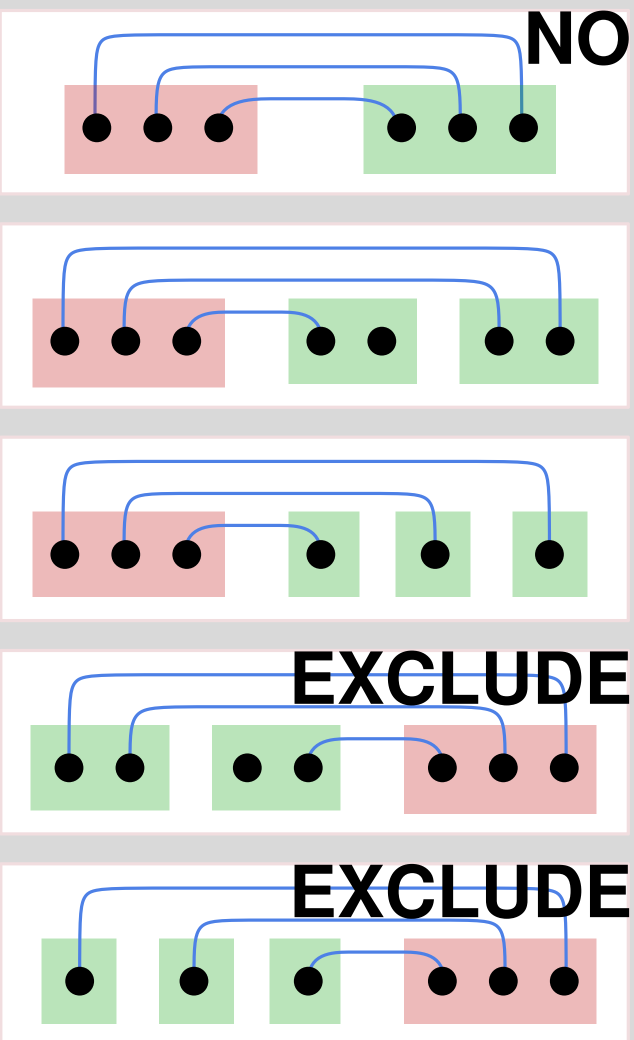


- avoid one of the two edges

$\rightarrow$ consider only children

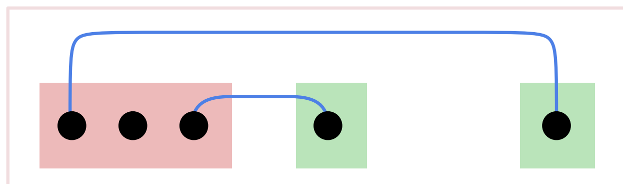
Improvement

Types of 3-rainbows



- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'
- left and right might conflict

$\rightarrow$ consider only following tripods



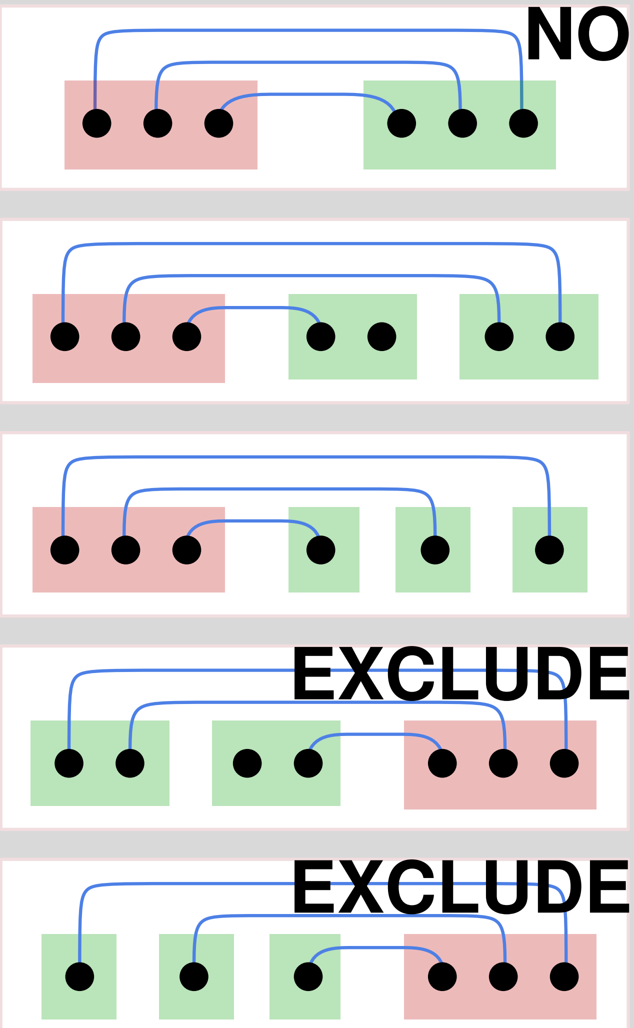
- avoid one of the two edges

$\rightarrow$ consider only children

- children are grouped based on which paths they don't see
- following tripods form a series of groups

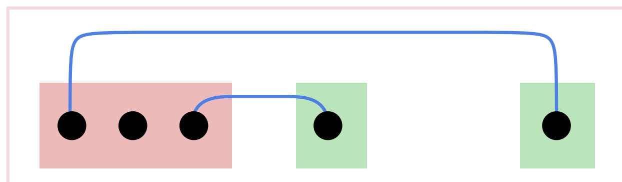
Improvement

Types of 3-rainbows



- for two adjacent tripods one is parent of the other
- if t' parent of t, t is not adjacent to a path p' of t'
- left and right might conflict

$\rightarrow$ consider only following tripods



- avoid one of the two edges

$\rightarrow$ consider only children

- children are grouped based on which paths they don't see
- following tripods form a series of groups

$\rightarrow$ require two groups

Restrictions

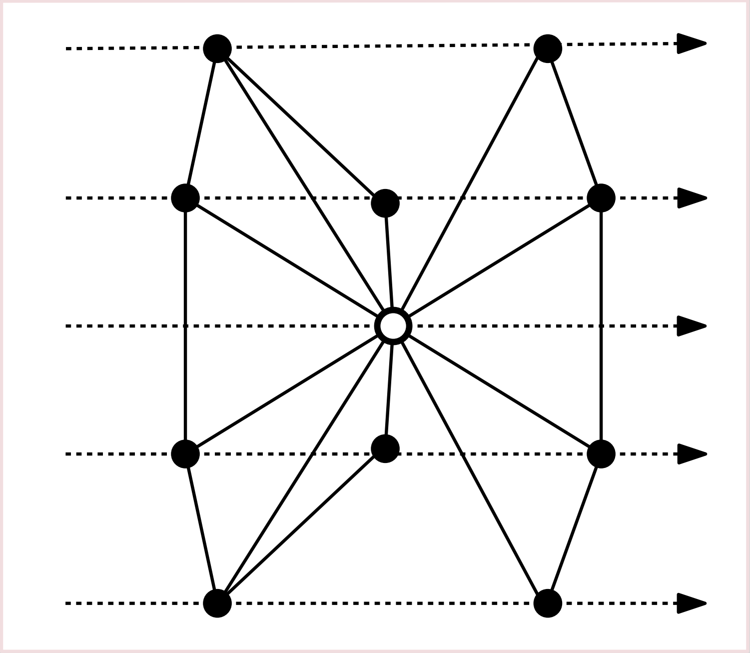
Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

Restrictions

Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

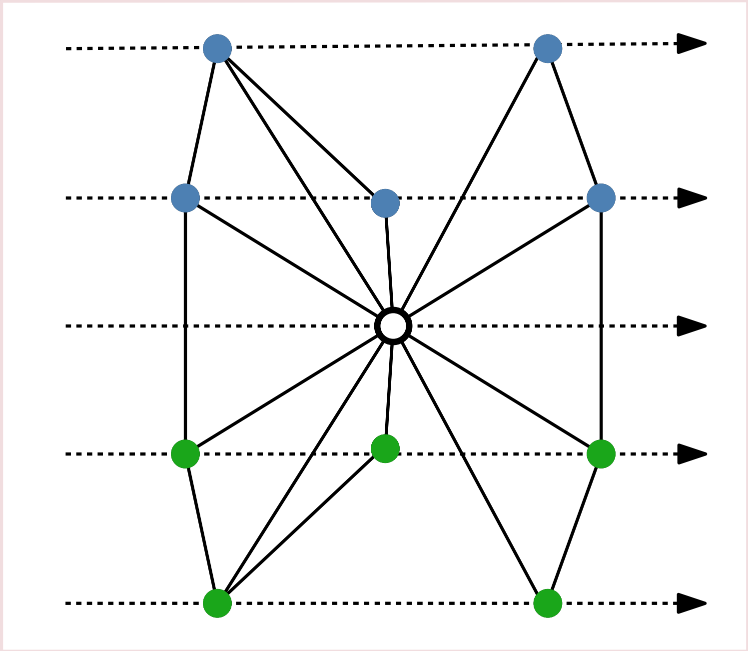


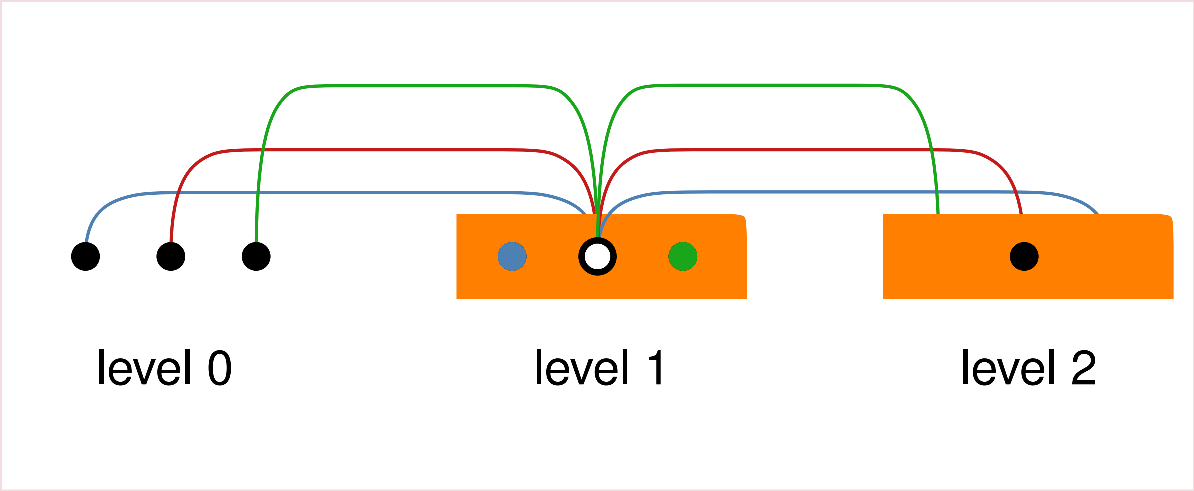
Restrictions

Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

- blue vertices precede and green follow
- exclude queues of the outerplanar subgraph
- other queues satisfy 1.



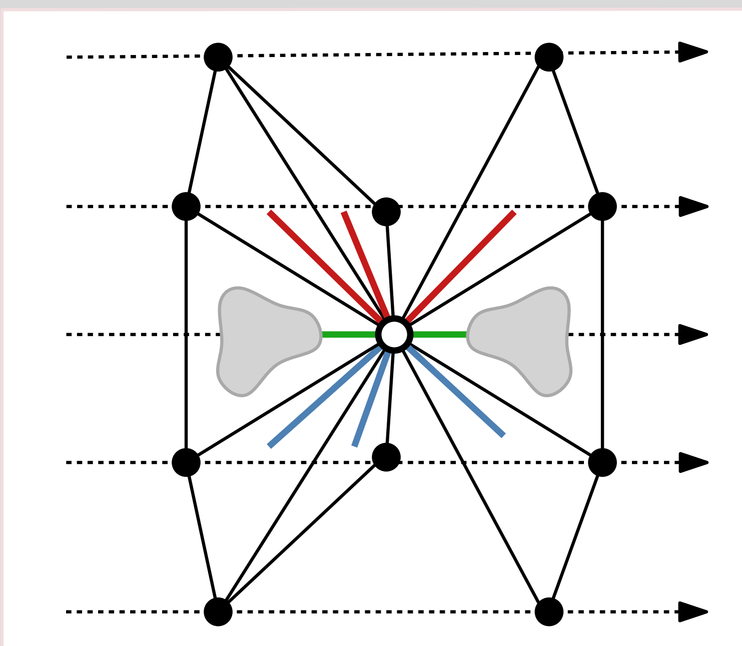


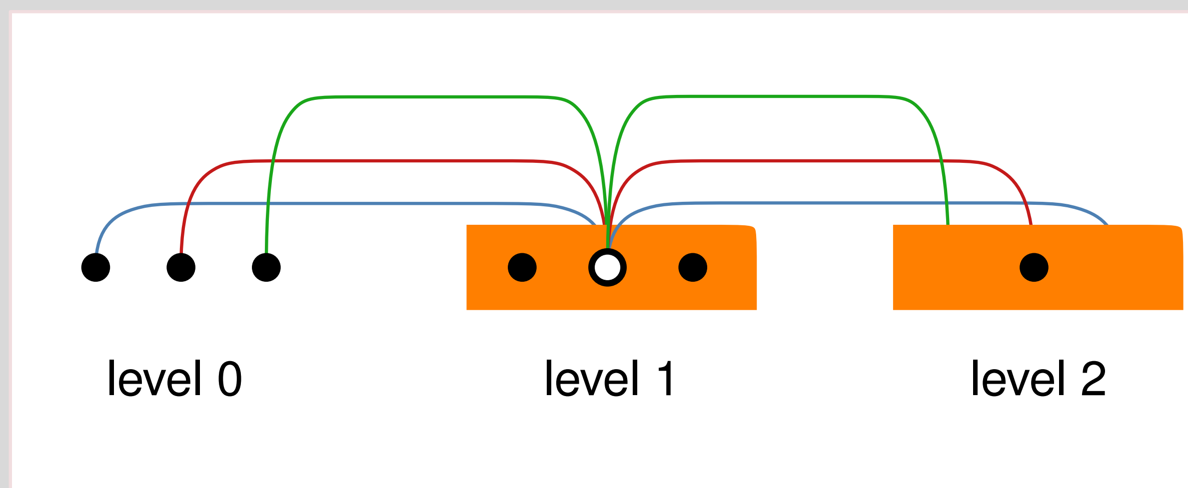
Restrictions

Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

- blue vertices precede and green follow
- exclude queues of the outerplanar subgraph
- other queues satisfy 1.
- green neighbors form two clusters





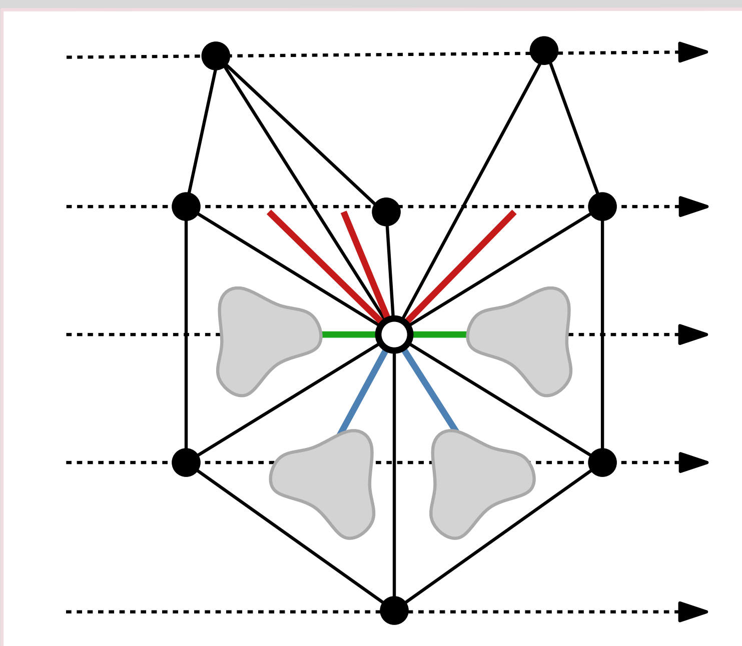
Restrictions

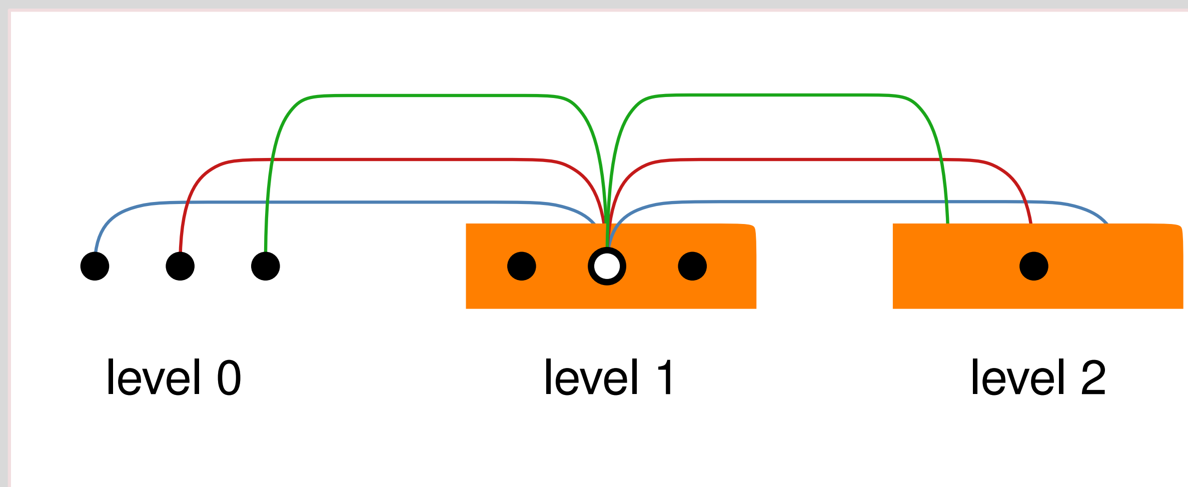
Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

- blue vertices precede and green follow
- exclude queues of the outerplanar subgraph
- other queues satisfy 1.

- green neighbors form two clusters
- do the same for the blue





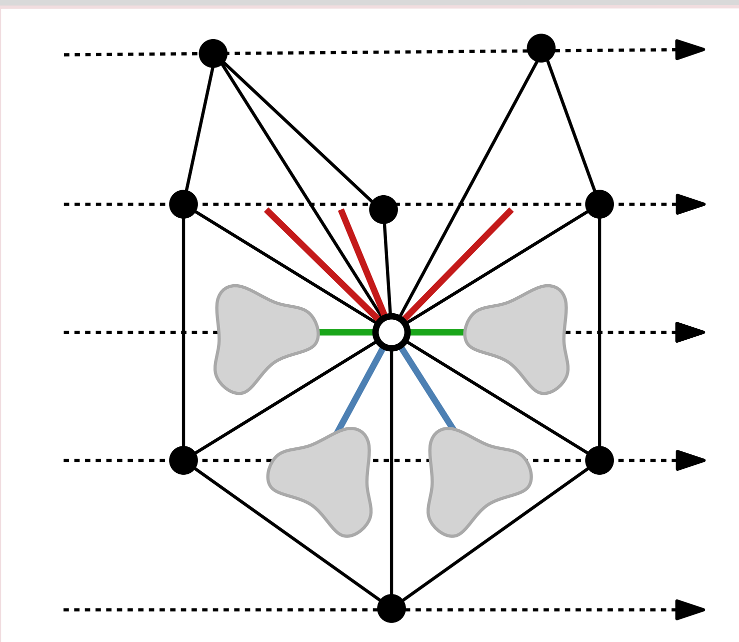
Restrictions

Restrictions

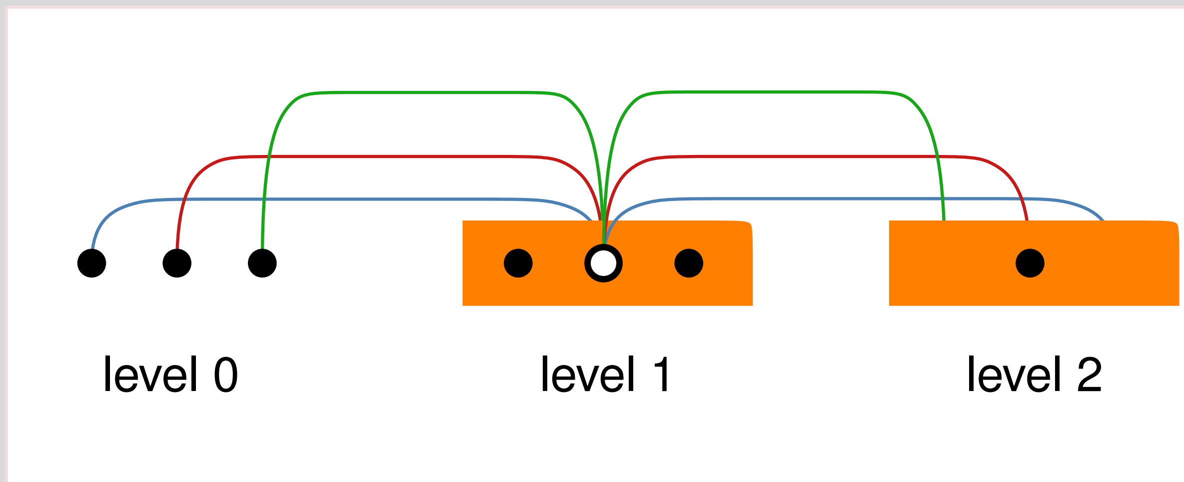
1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

- blue vertices precede and green follow
- exclude queues of the outerplanar subgraph
- other queues satisfy 1.

- green neighbors form two clusters
- do the same for the blue



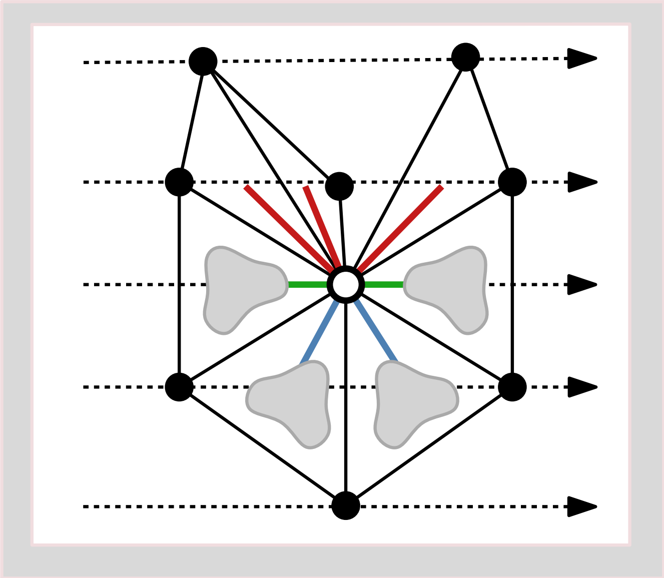
we can improve on two queues of the 5-queue layout of *H*



Restrictions

Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

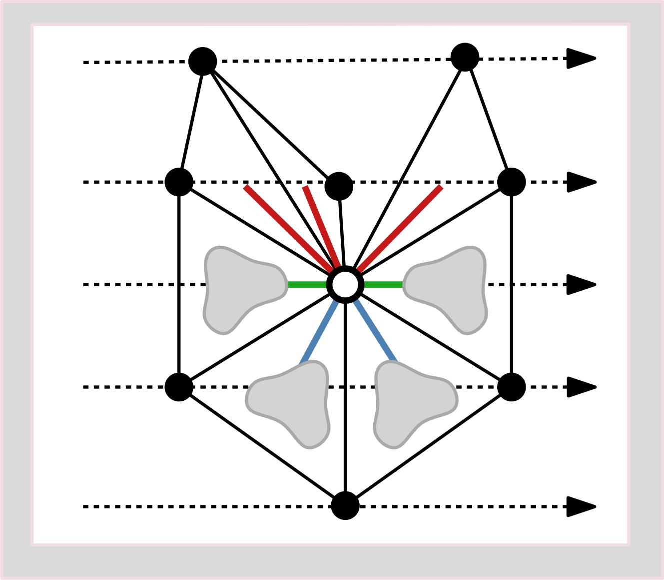


Restrictions

Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children

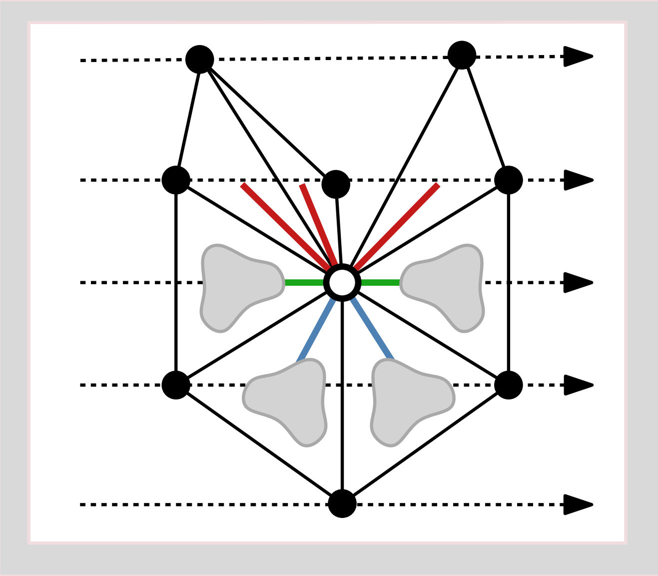
- green and blue clusters to be children
- each cluster to form a group,
  i.e. to not see a path of the tripod



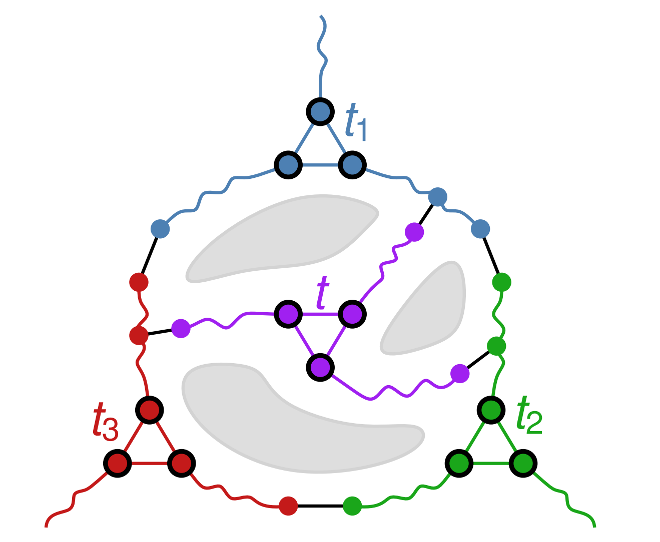
Restrictions

Restrictions

1. Consider only following tripods
2. Split considered nodes into two groups
3. Consider only children



- green and blue clusters to be children
- each cluster to form a group,
  i.e. to not see a path of the tripod



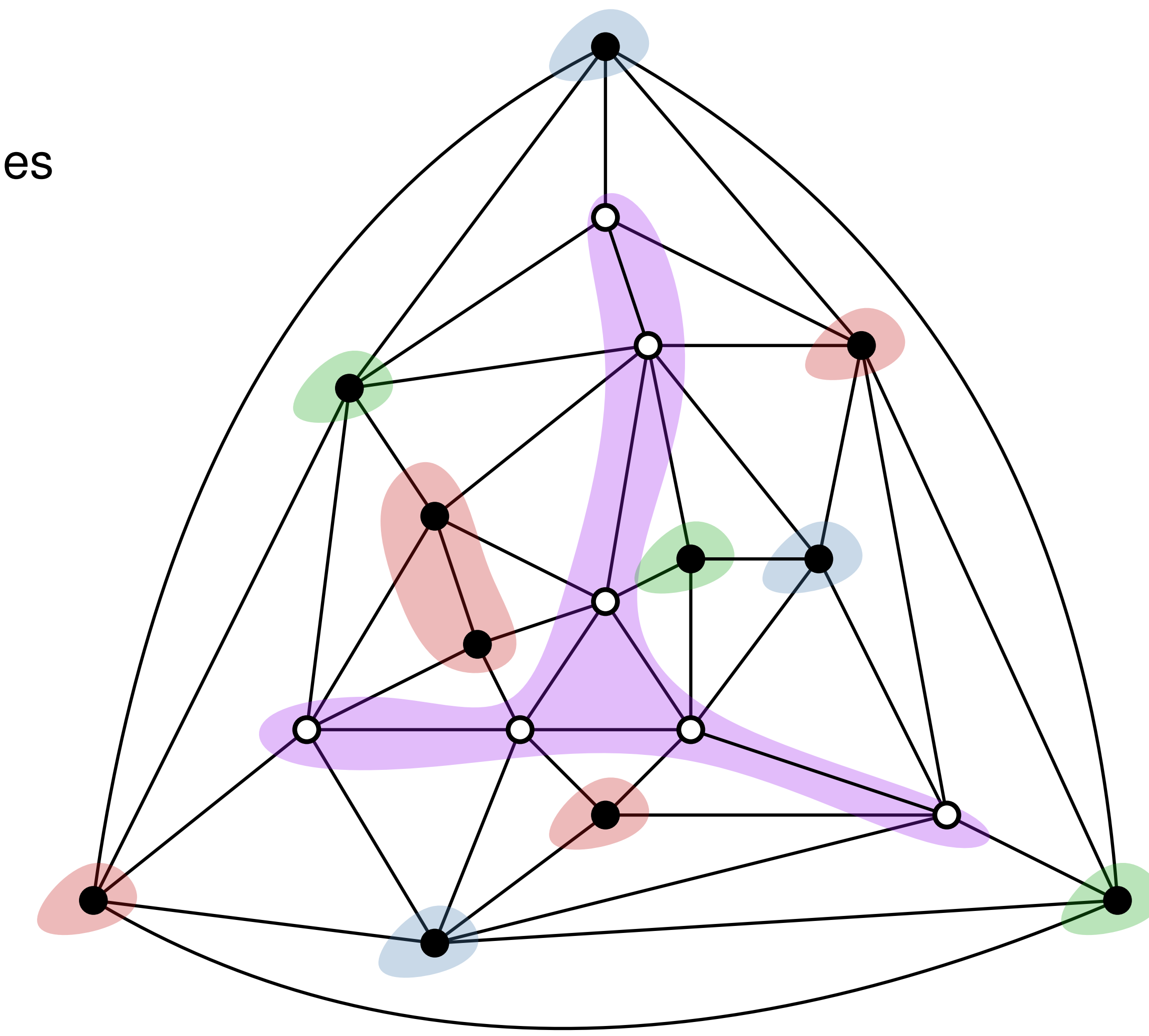
- each cluster to be in a shaded region
- graph *H* must reflect planarity of *G*
- graph *H* must respect parent-child relation
- the level of parent can't be greater than the child's

Improvement

- Contract tripods in *G*
  - preserve embedding
  - remove homotopic parallel edges

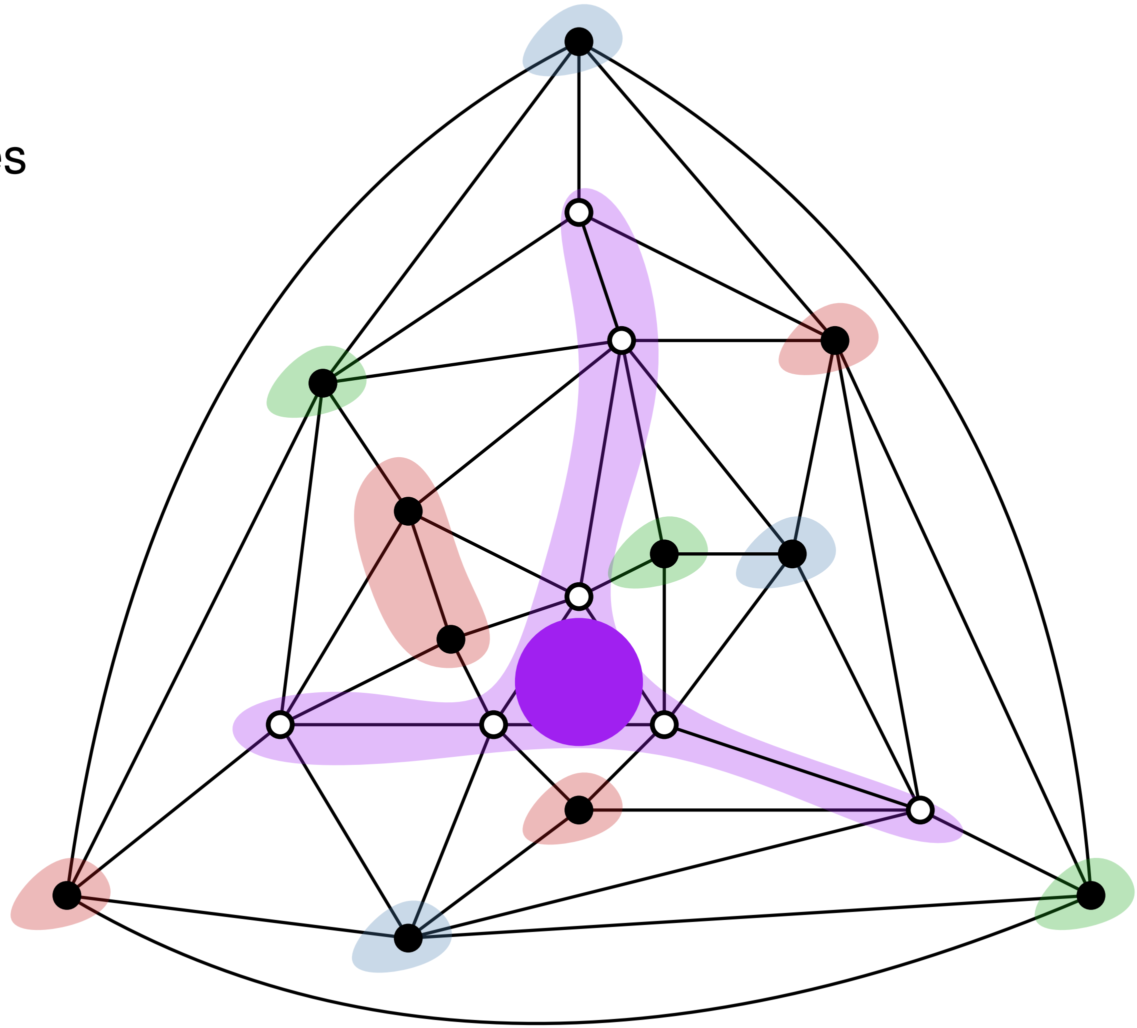
Improvement

- Contract tripods in *G*
  - preserve embedding
  - remove homotopic parallel edges



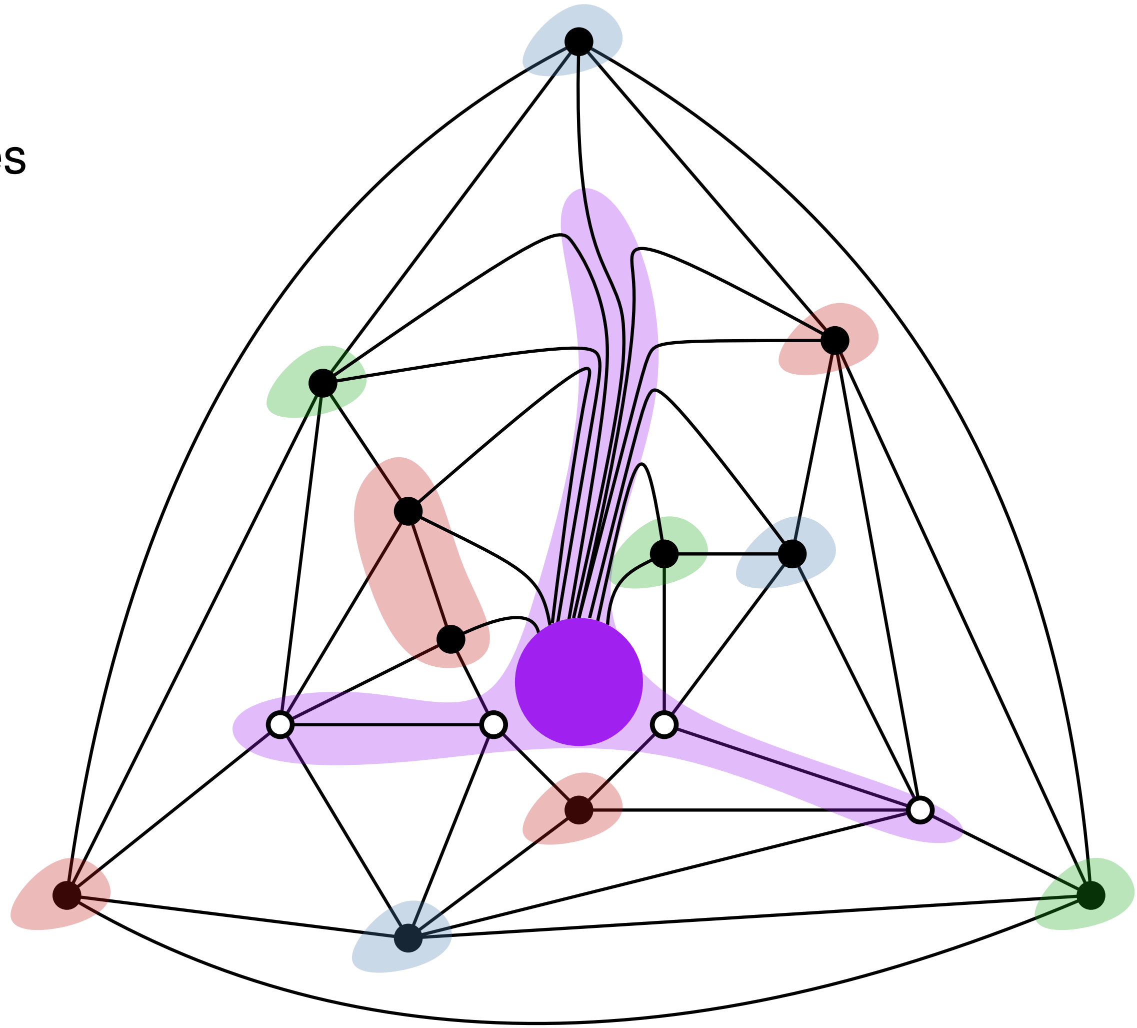
Improvement

- Contract tripods in G
  - preserve embedding
  - remove homotopic parallel edges



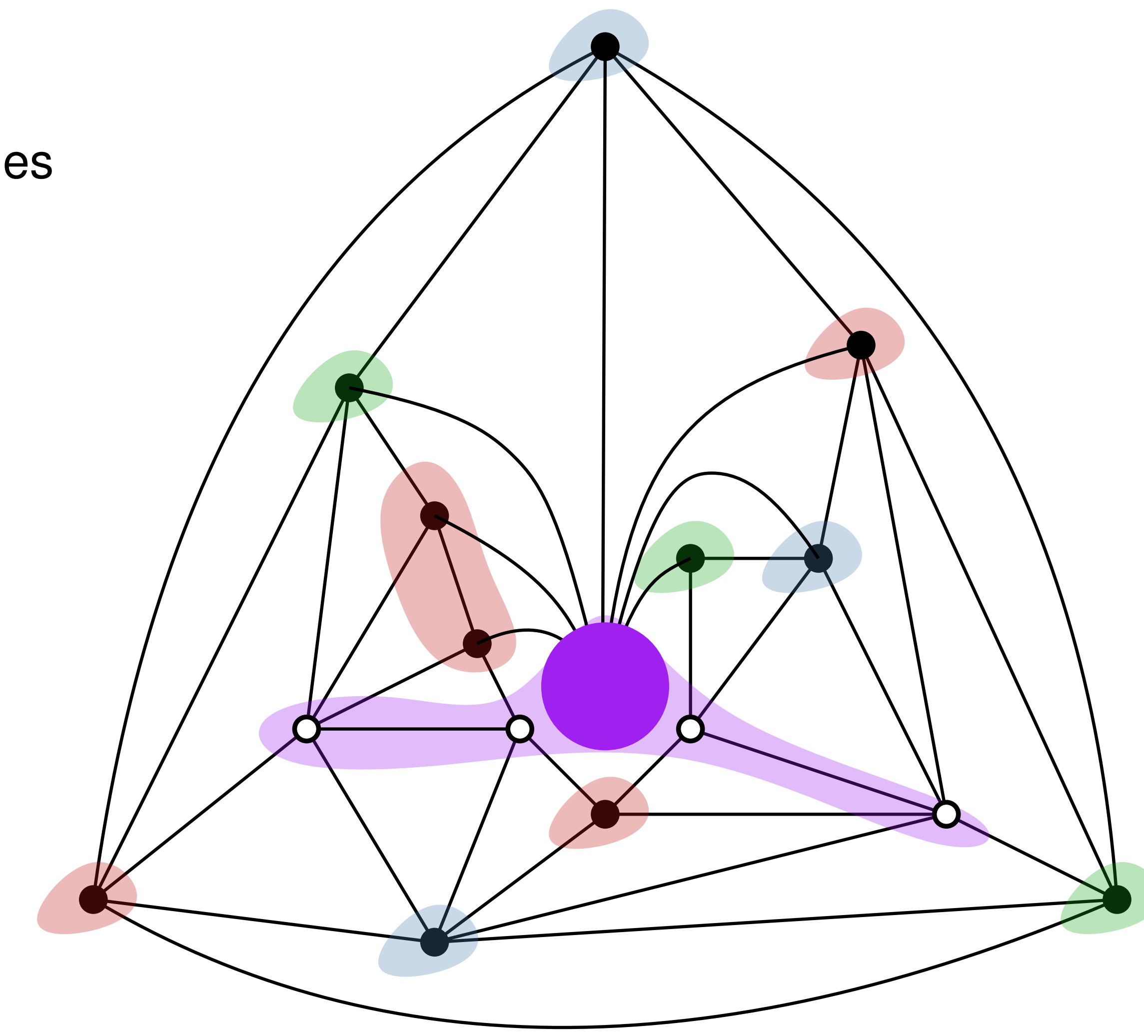
Improvement

- Contract tripods in G
  - preserve embedding
  - remove homotopic parallel edges



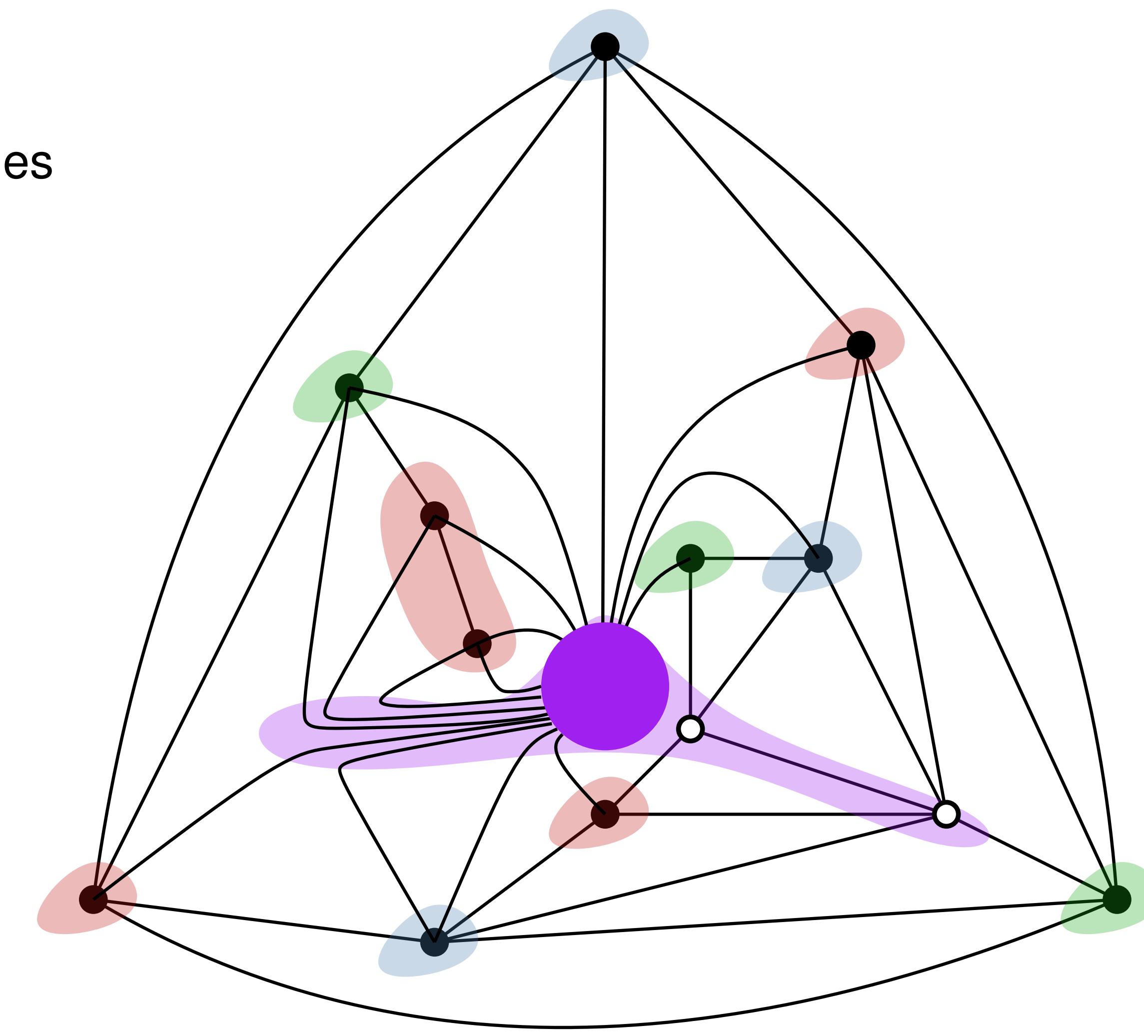
Improvement

- Contract tripods in *G*
  - preserve embedding
  - remove homotopic parallel edges



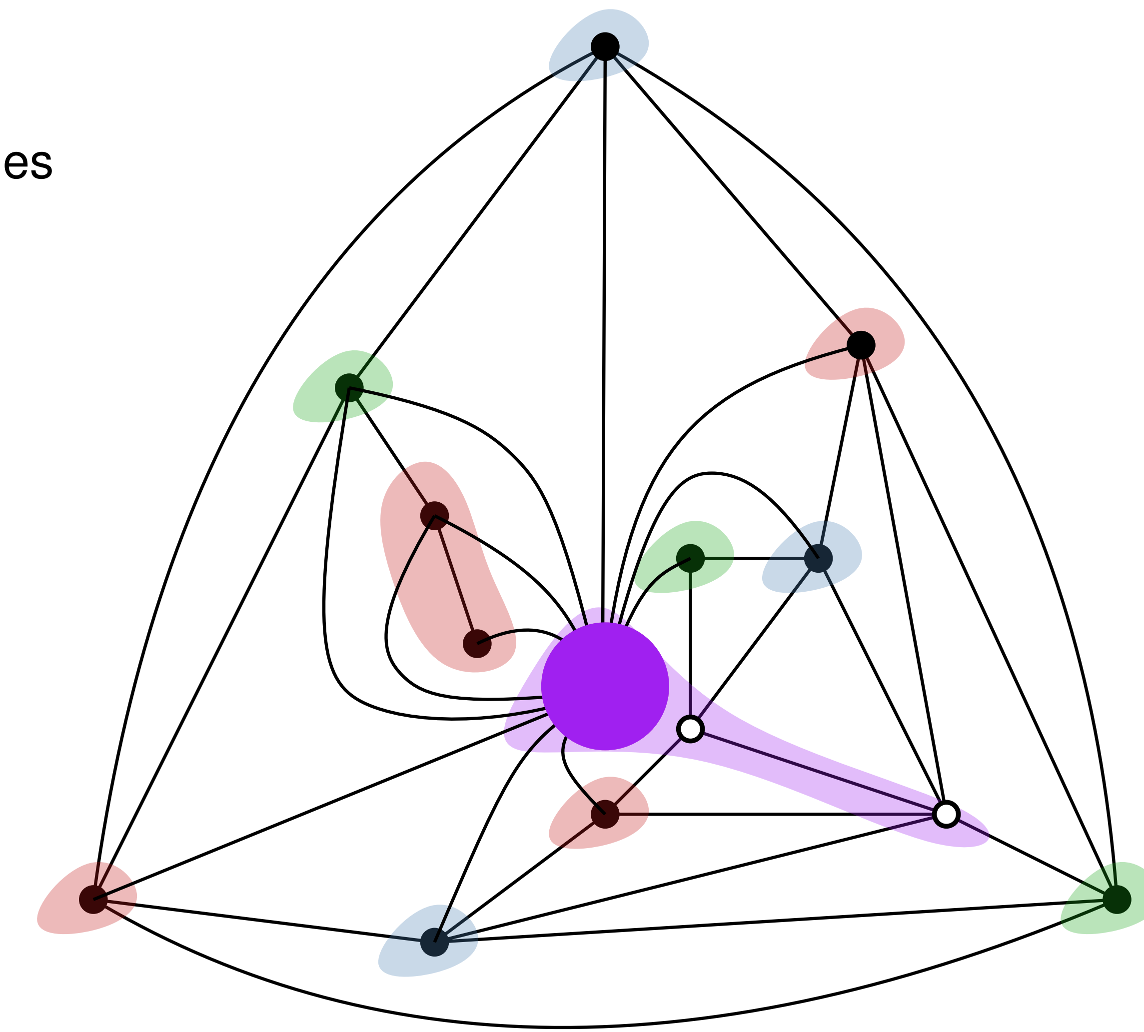
Improvement

- Contract tripods in G
  - preserve embedding
  - remove homotopic parallel edges



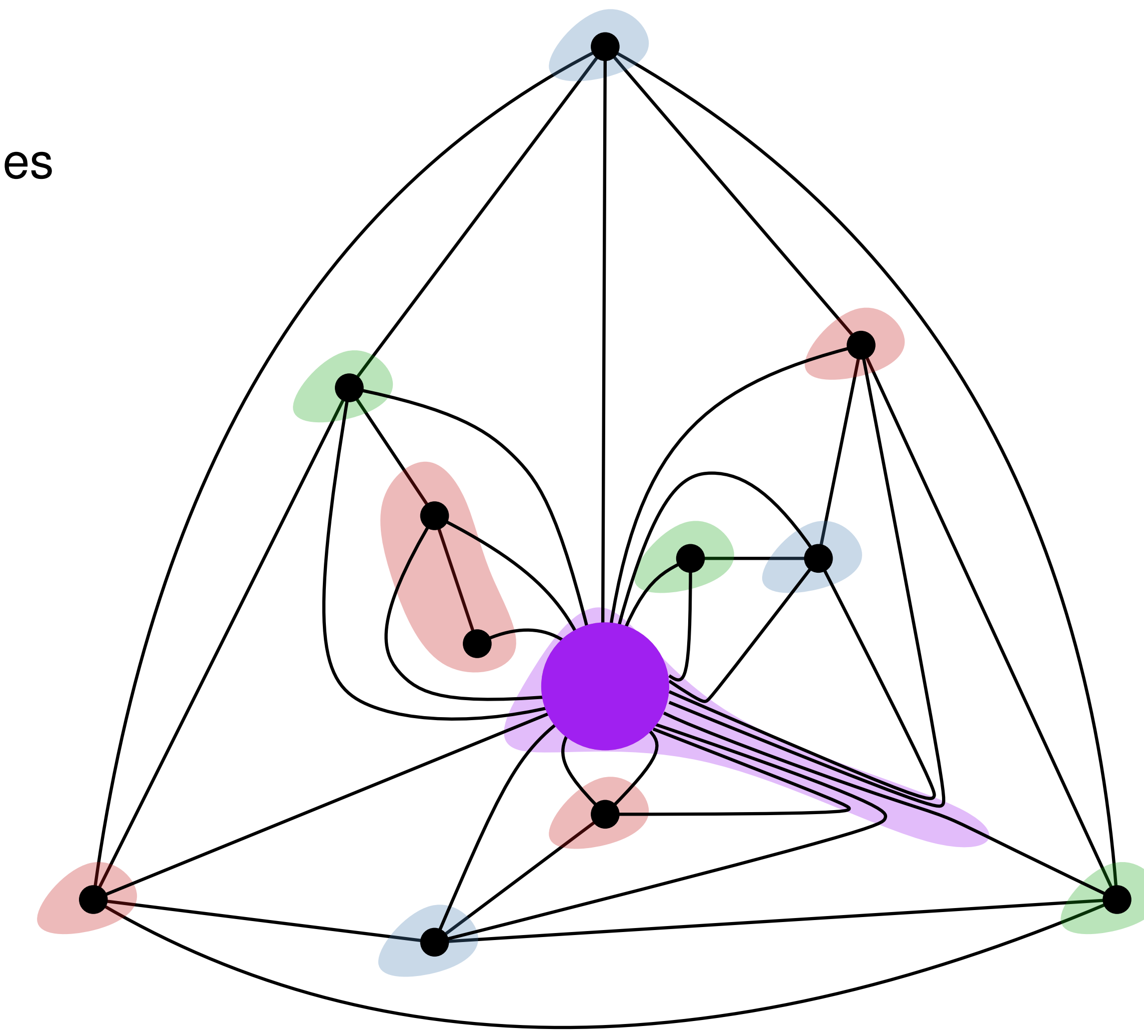
Improvement

- Contract tripods in G
  - preserve embedding
  - remove homotopic parallel edges



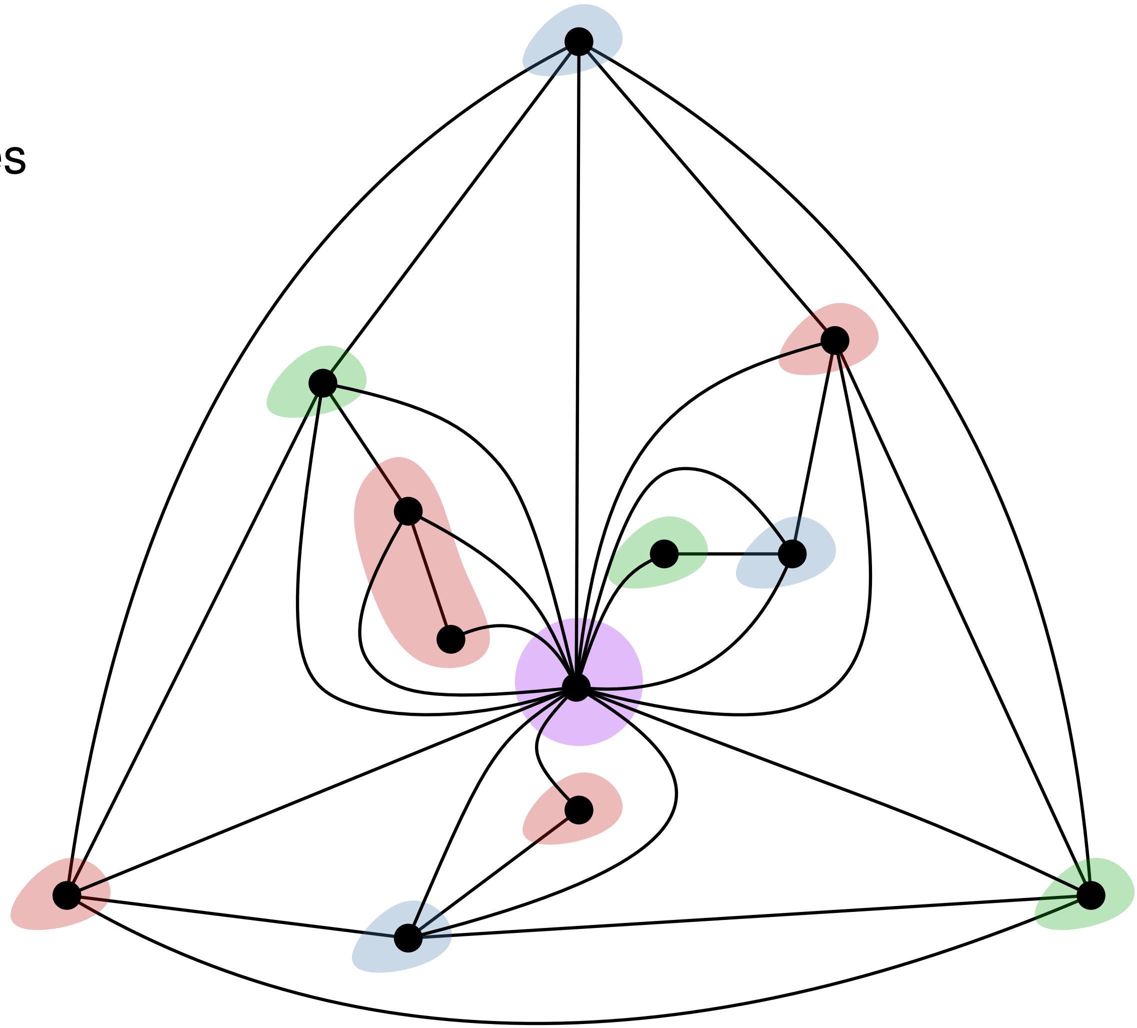
Improvement

- Contract tripods in G
  - preserve embedding
  - remove homotopic parallel edges



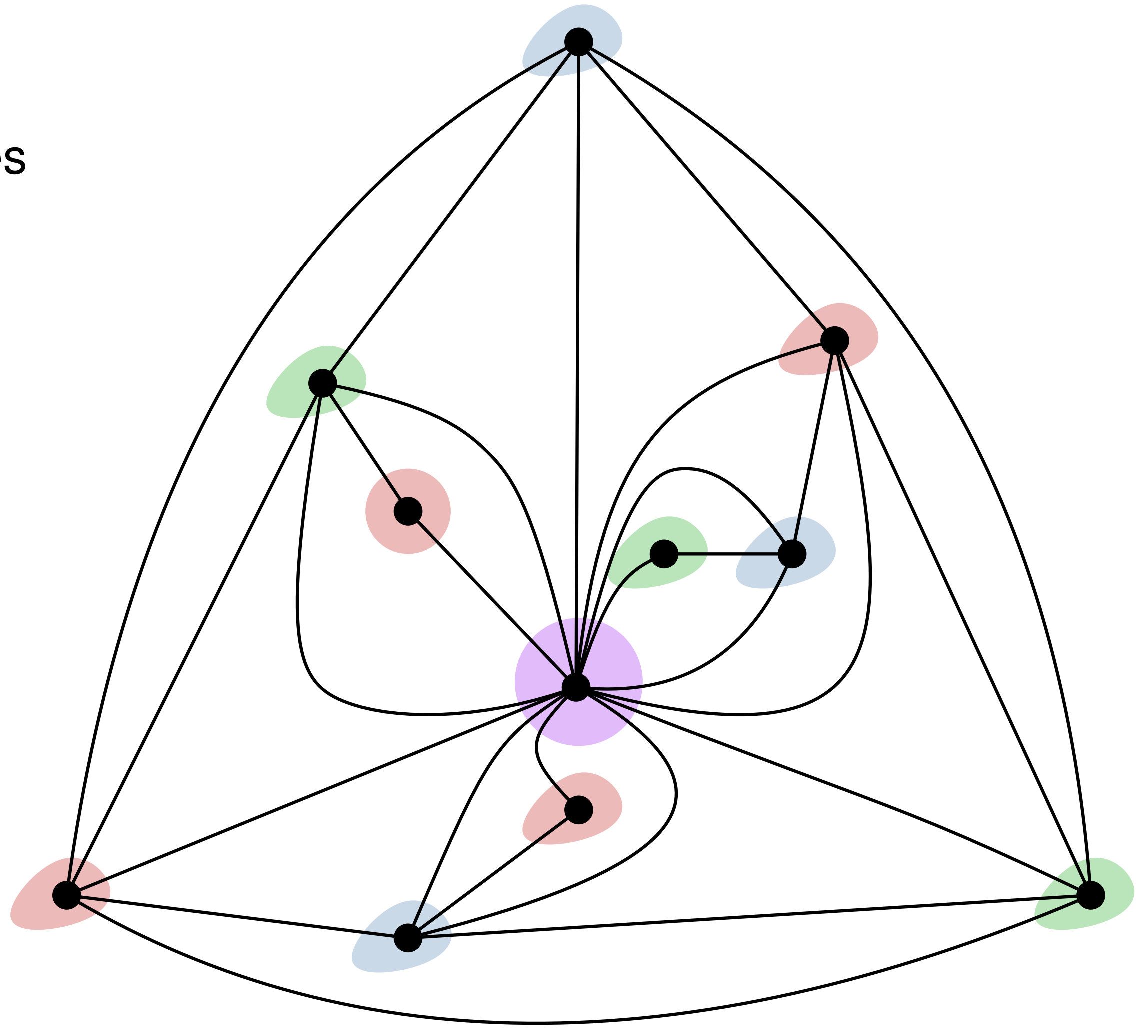
Improvement

- Contract tripods in *G*
  - preserve embedding
  - remove homotopic parallel edges



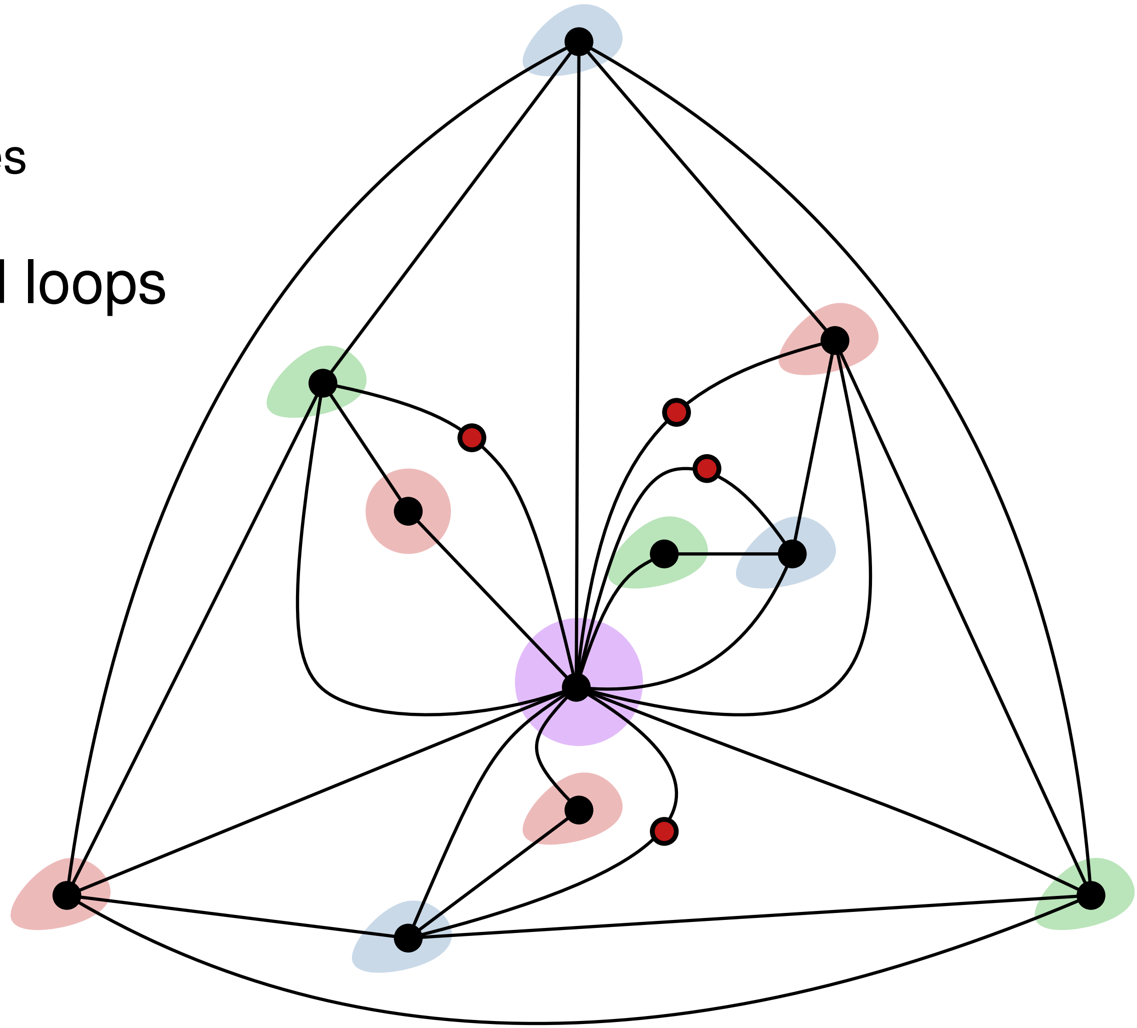
Improvement

- Contract tripods in G
  - preserve embedding
  - remove homotopic parallel edges



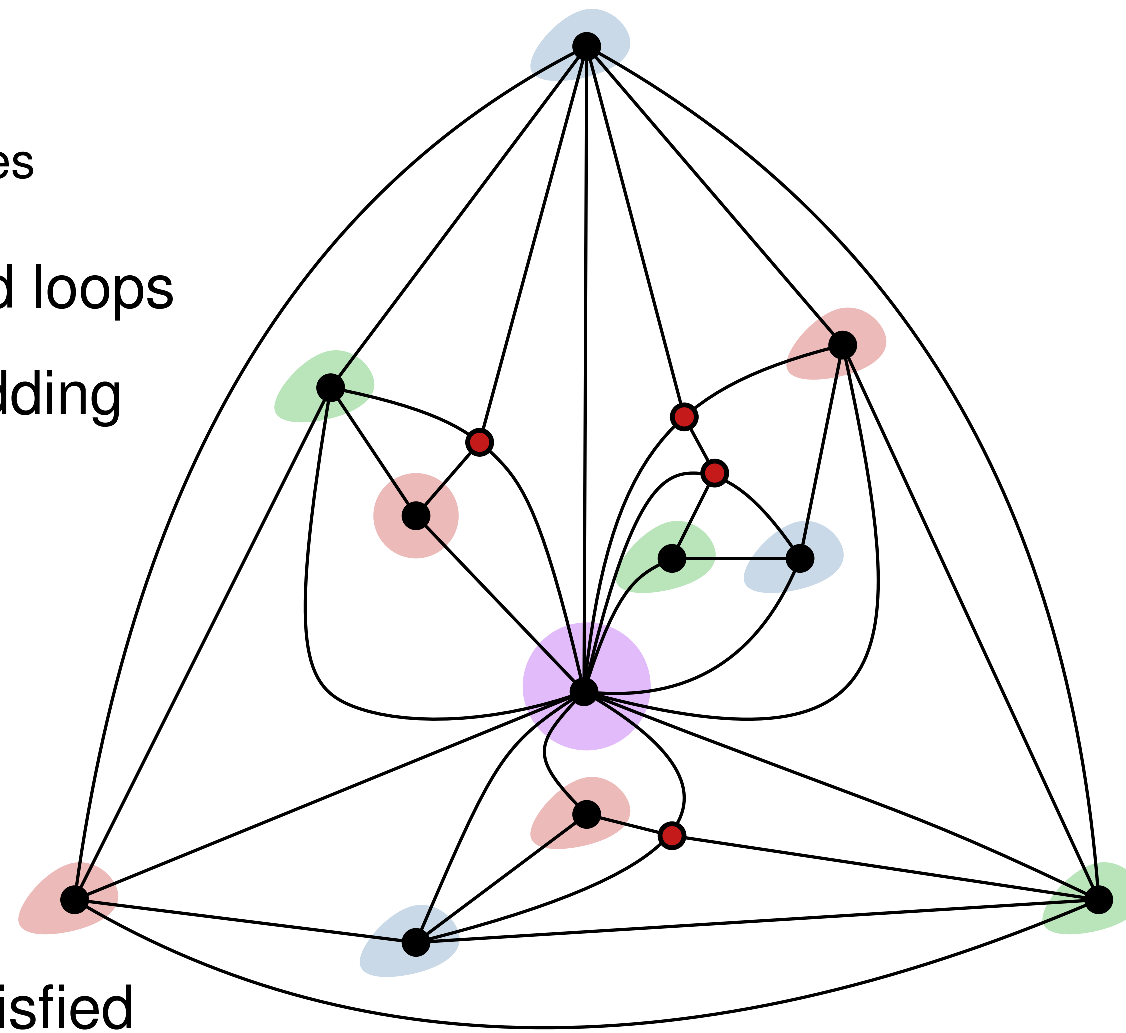
Improvement

- Contract tripods in G
  - preserve embedding
  - remove homotopic parallel edges
- Subdivide parallel edges and loops



Improvement

- Contract tripods in *G*
  - preserve embedding
  - remove homotopic parallel edges
- Subdivide parallel edges and loops
- Extend by preserving embedding



- All three restrictions are satisfied
- The paths can be ordered with no 3-rainbows

Summary

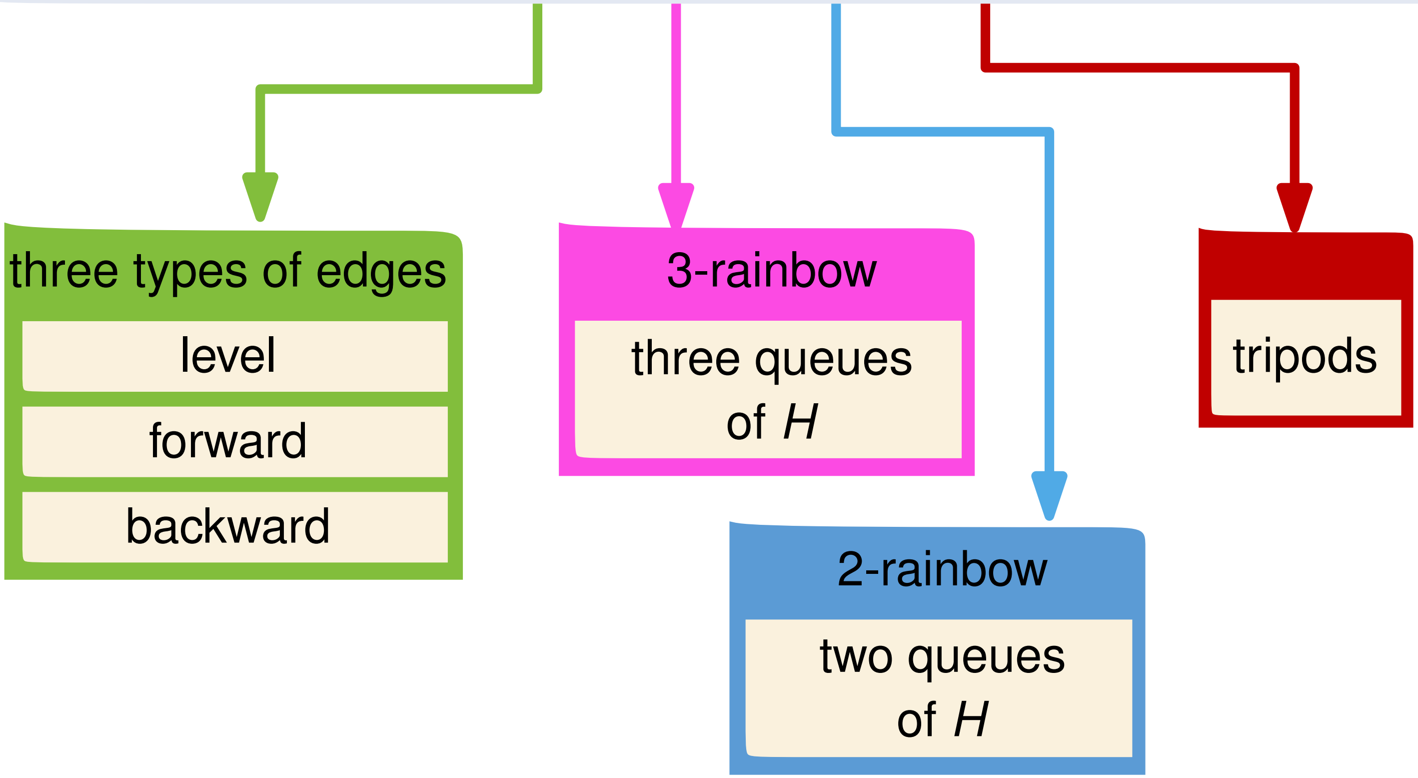
Improved bound:

$$qn(G) \leq 3 \cdot (3 \cdot 3 + 2 \cdot 2) + 3 = 42$$

Summary

Improved bound:

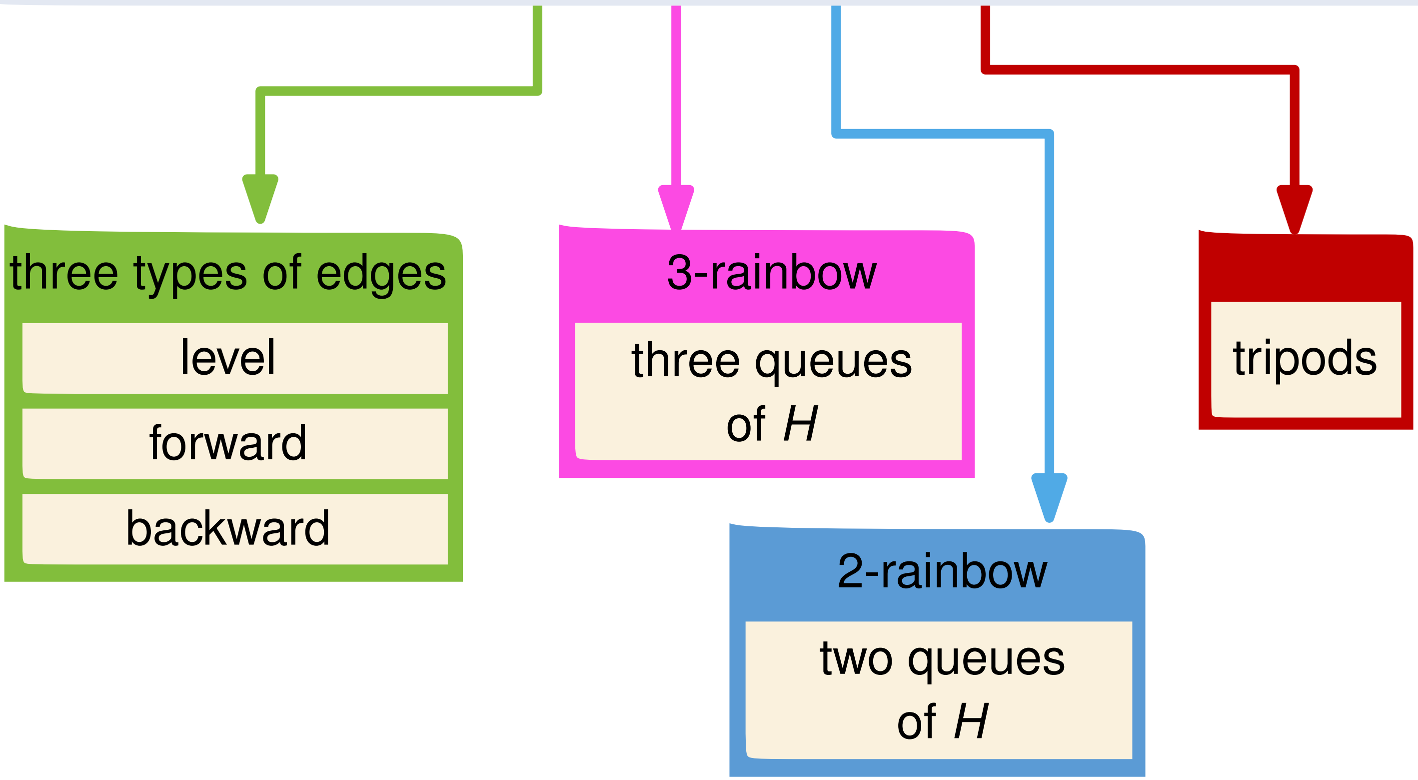
$$qn(G) \leq 3 \cdot (3 \cdot 3 + 2 \cdot 2) + 3 = 42$$



Summary

Improved bound:

$$qn(G) \leq 3 \cdot (3 \cdot 3 + 2 \cdot 2) + 3 = 42$$

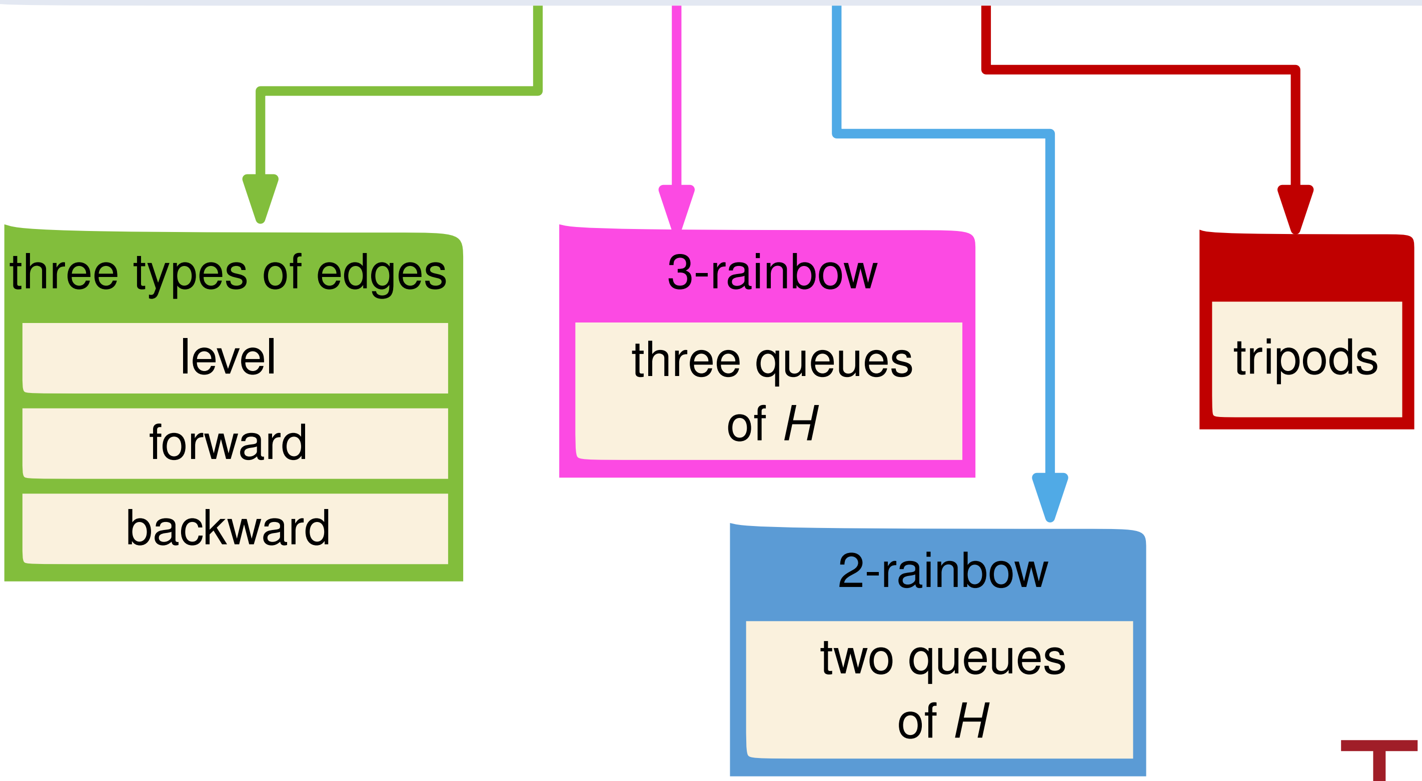


- Further reduce
- Subfamilies

Summary

Improved bound:

$$qn(G) \leq 3 \cdot (3 \cdot 3 + 2 \cdot 2) + 3 = 42$$



- Further reduce
- Subfamilies

Thank you