# Algorithms for Finding Distance-Edge-Colorings of Graphs

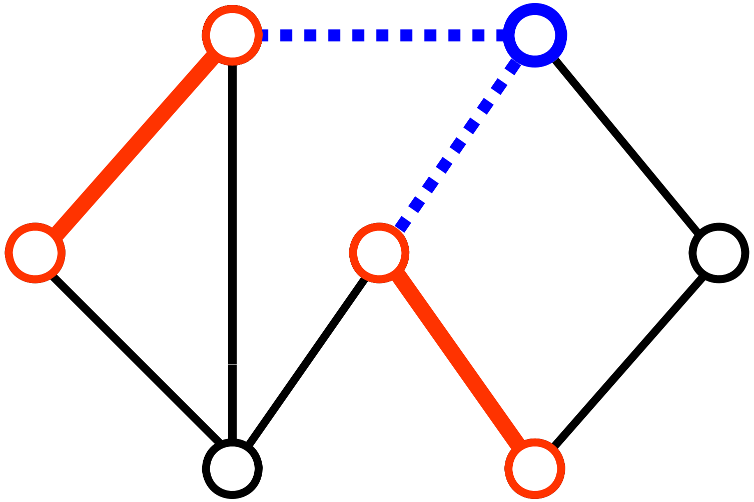
Takehiro Ito Akira Kato

Xiao Zhou Takao Nishizeki

Tohoku University

# Distance between two edges

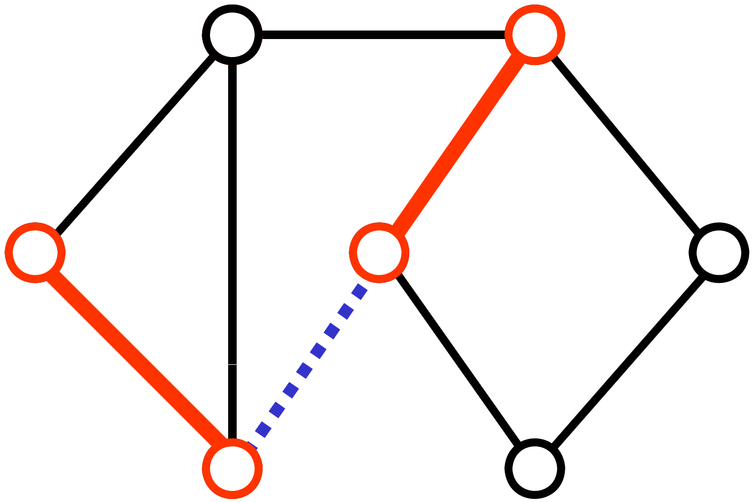
# of edges in a shortest path between two edges



Distance = 2

# Distance between two edges

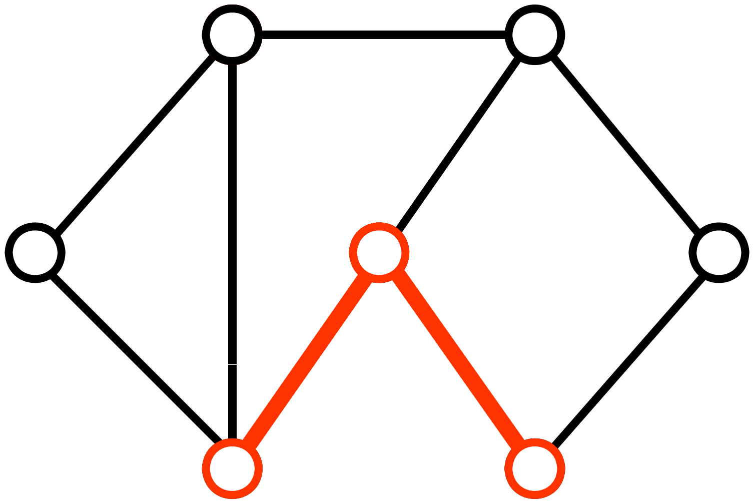
# of edges in a shortest path between two edges



Distance = 1

# Distance between two edges

# of edges in a shortest path between two edges

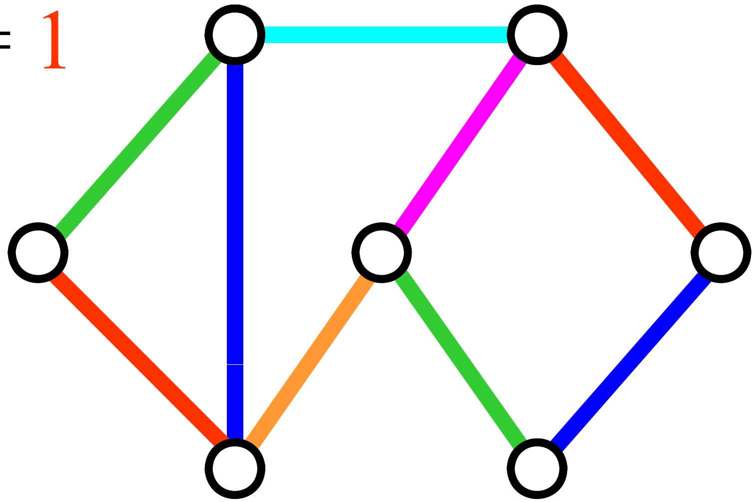


Distance = 0

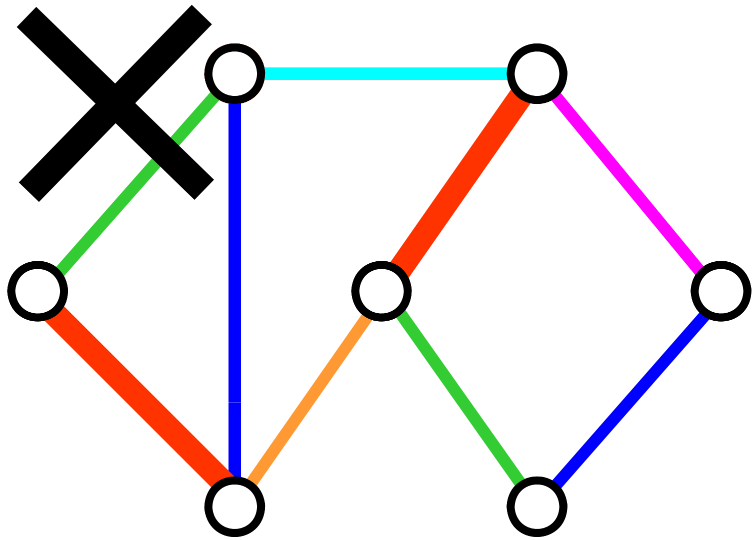
# Distance-Edge-Coloring or *l*-Edge-Coloring

for a given bounded integer *l*,
any two edges within distance *l* have different colors

$l = 1$



1-edge-coloring
with six colors

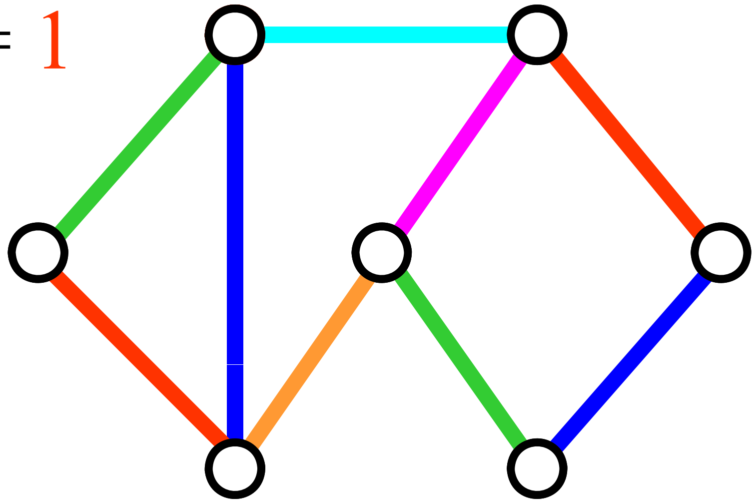


Not 1-edge-coloring

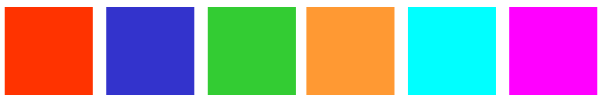
# Distance-Edge-Coloring or $l$-Edge-Coloring

for a given bounded integer $l$,
any two edges within distance $l$ have different colors

$l = 1$



1-edge-coloring
with six colors



0-edge-coloring

⇕

ordinary edge-coloring

1-edge-coloring

⇕

strong edge-coloring

# *l*-Edge-Coloring Problem

find an *l*-edge-coloring with min # of colors

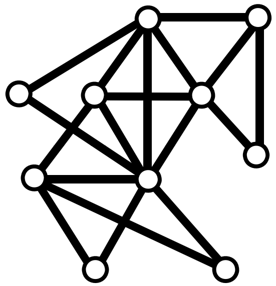
The ordinary edge-coloring problem is NP-hard

The *l*-edge-coloring problem is NP-hard in general

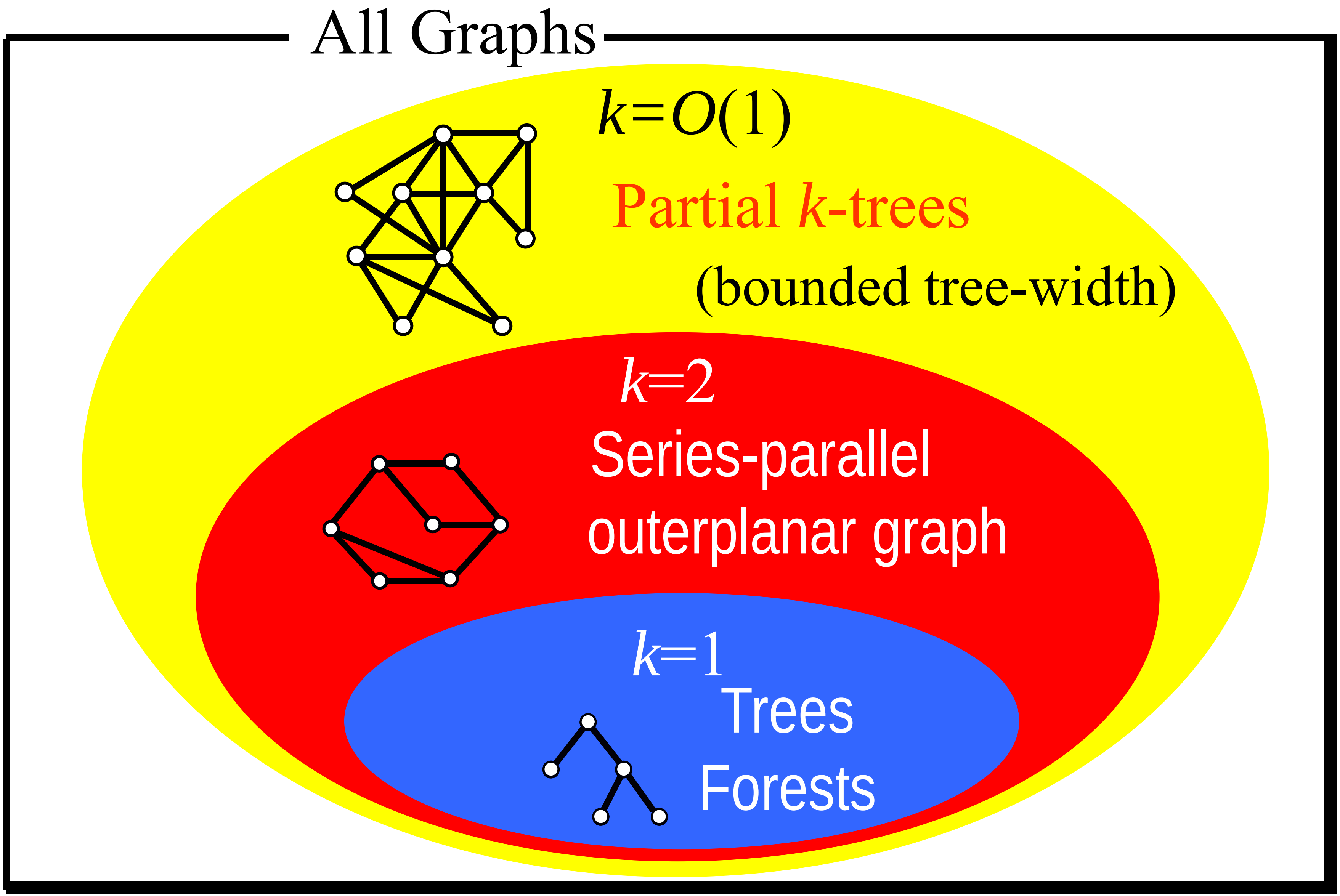
?

restricted class

Partial *k*-trees



# Class of Partial *k*-Trees



# Related Results on Partial *k*-Trees

$l = 0$ (ordinary edge-coloring problem)

Linear-time algorithm [Zhou *et al.* 1996]

$l = 1$ (strong edge-coloring problem)

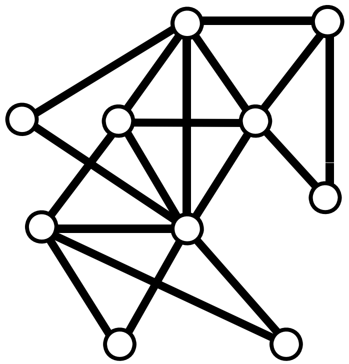
Polynomial-time algorithm [Salavatiour 2004]

*l*-edge-coloring problem ?

# Our Results

Partial $k$-Trees

($k$, $l$ : constant)



Polynomial-time exact algorithm

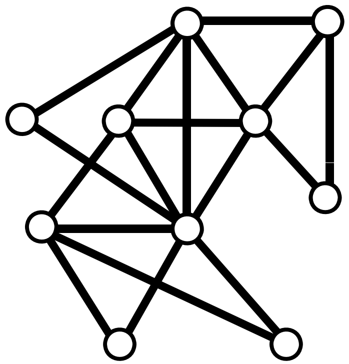
$O(n\,\alpha^{O(1)})$

partial $k$-tree has an $l$-edge-coloring with $\alpha$ colors?

$n$: # of vertices

# Our Results

**Partial $k$-Trees** (*$k$*, *$l$* : constant)



Polynomial-

$2^{2(k+1)(l+1)+1} = O(1)$

$O(\,n\,\alpha^{O(1)}\,)$

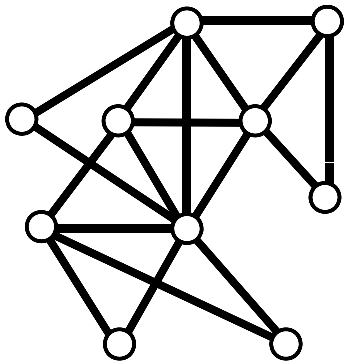
partial $k$-tree has an $l$-edge-coloring with $\alpha$ colors?

$n$: # of vertices

# Our Results

Partial $k$-Trees

($k$, $l$ : constant)



Polynomial-time exact algorithm

$$O(n\,\alpha^{O(1)})$$

partial $k$-tree has an $l$-edge-coloring with $\alpha$ colors?

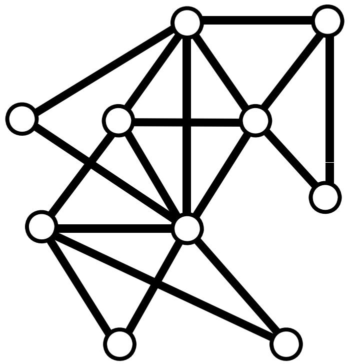
min # of colors

$\alpha \leq$ (# of edges)

$\alpha = O(1)$ ⇒ $O(n)$ time

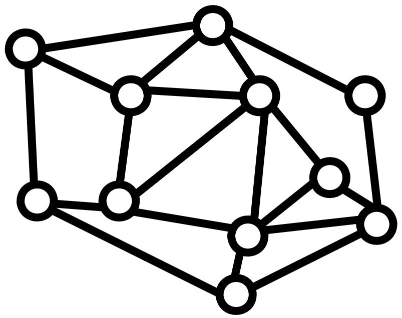
# Our Results

Partial *k*-Trees



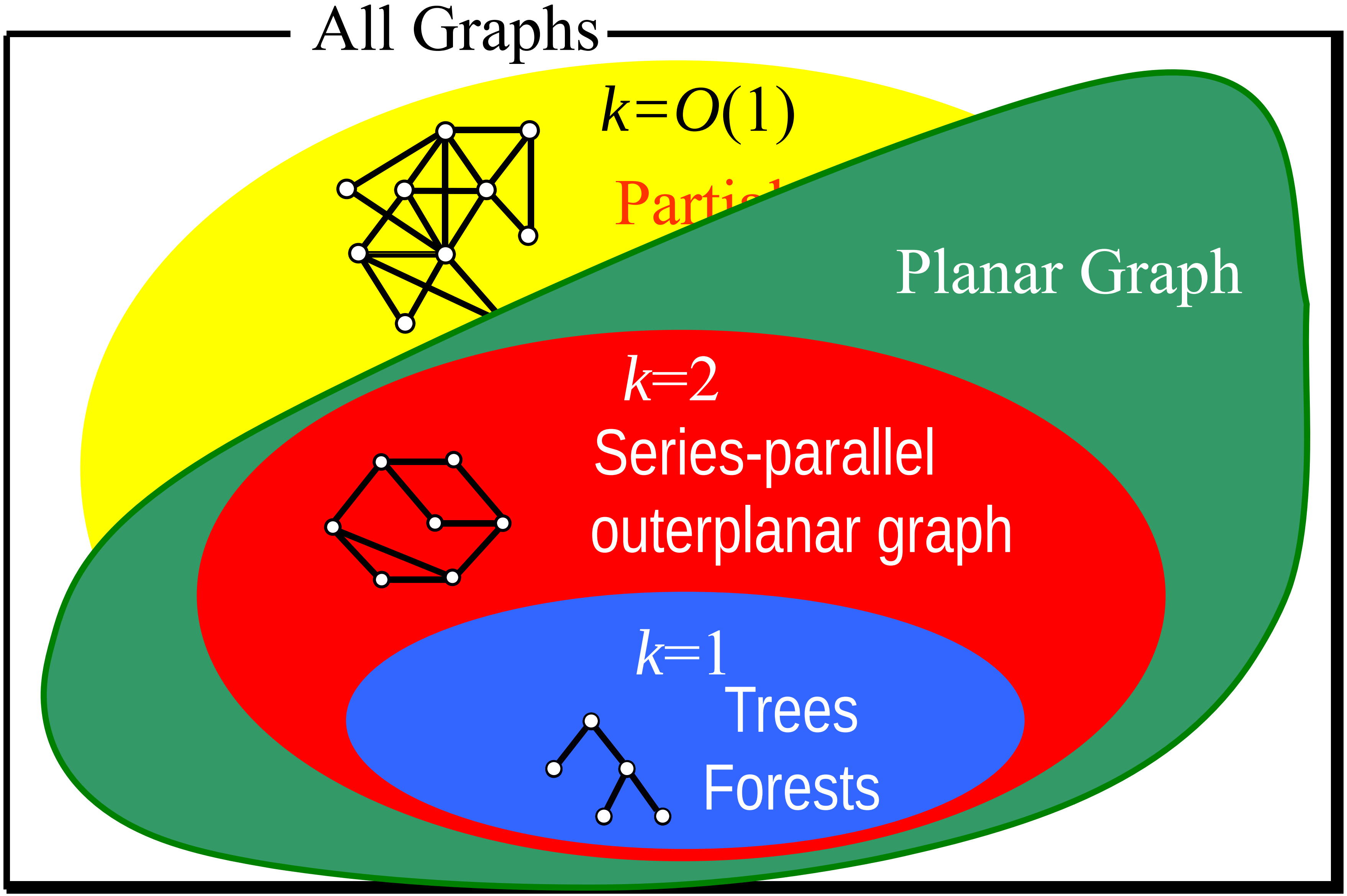
Polynomial-time exact algorithm

Planar Graphs



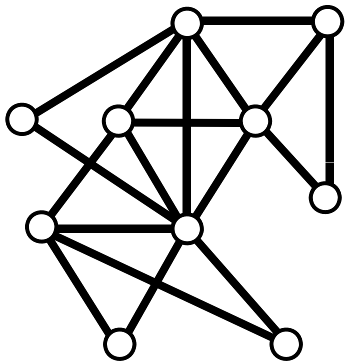
Polynomial-time
2-approximation algorithm

# Class of Partial *k*-Trees



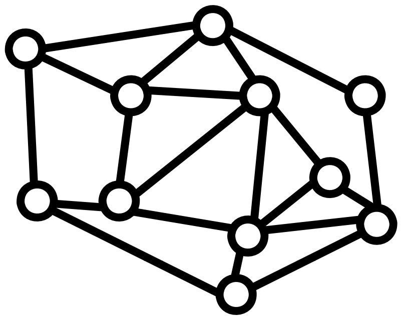
# Our Results

Partial *k*-Trees



Polynomial-time exact algorithm

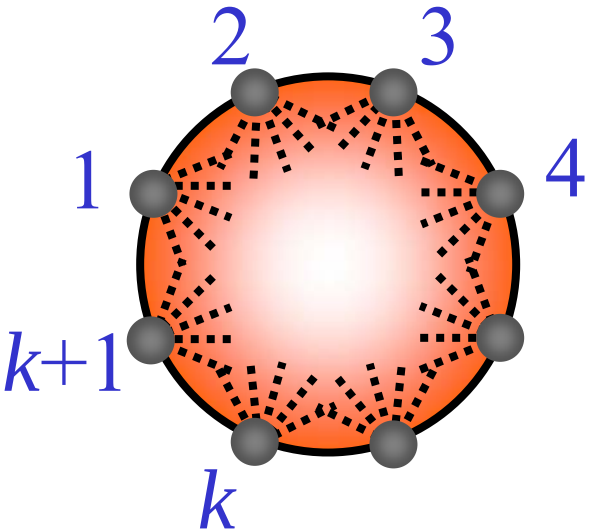
Planar Graphs



Polynomial-time
2-approximation algorithm

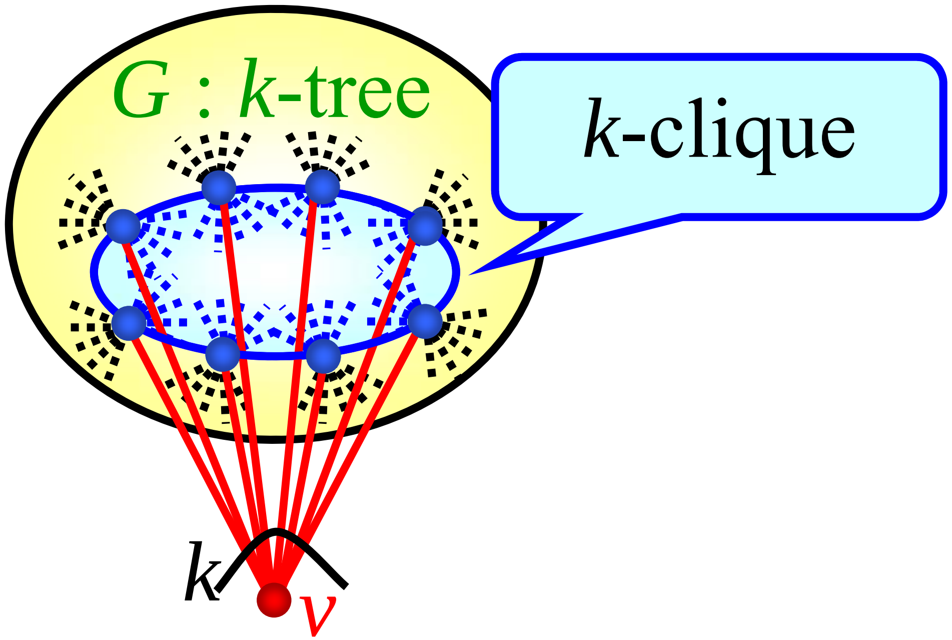
# Partial *k*-Trees

*k*-tree (recursive definition)

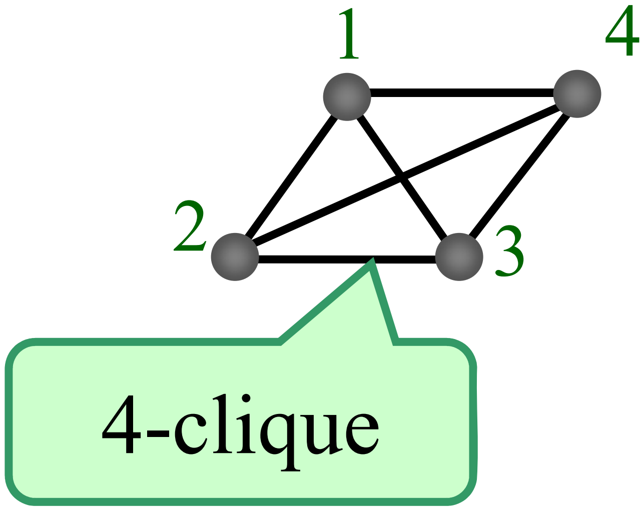



(*k*+1)-clique

A complete graph with *k*+1 vertices is a *k*-tree.



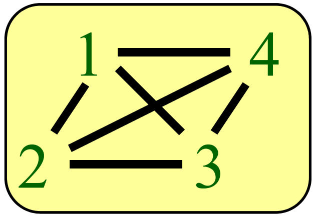

A graph obtained from *G* by adding a new vertex *v* and joining it with each of the *k* vertices is a *k*-tree.

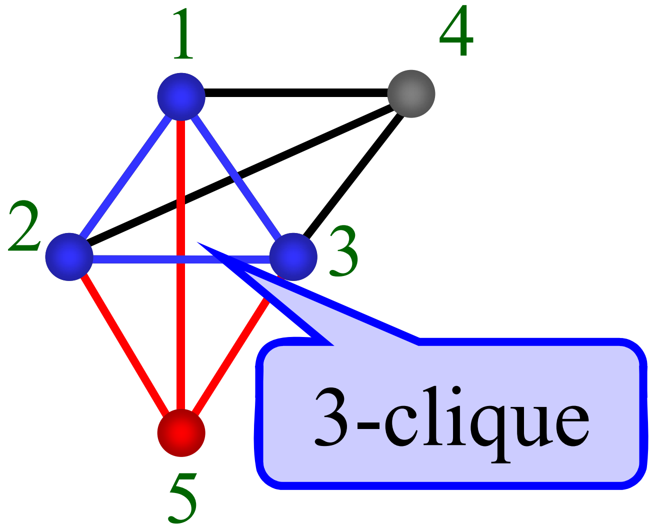



3-tree

## Tree-decomposition

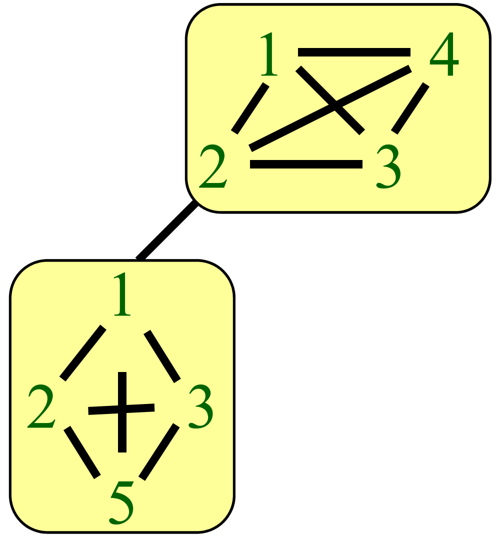
node

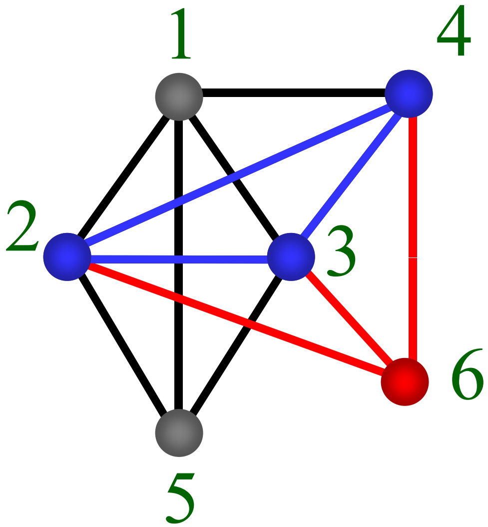




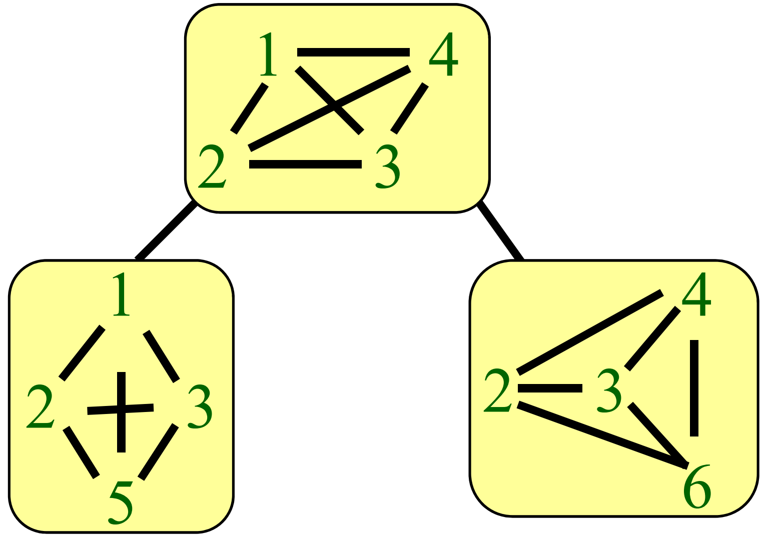

3-tree

## Tree-decomposition

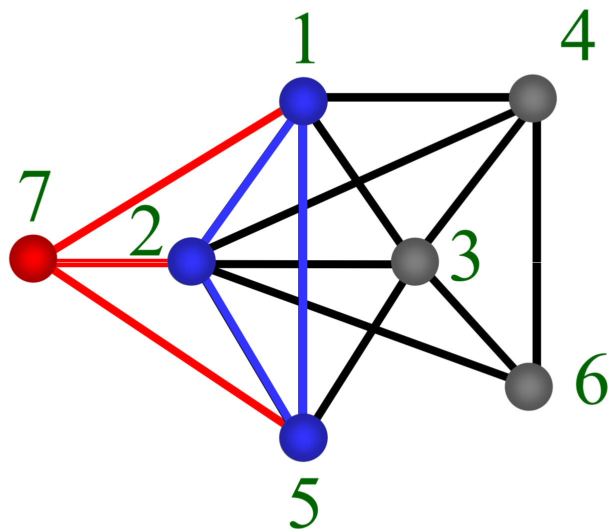





## Tree-decomposition

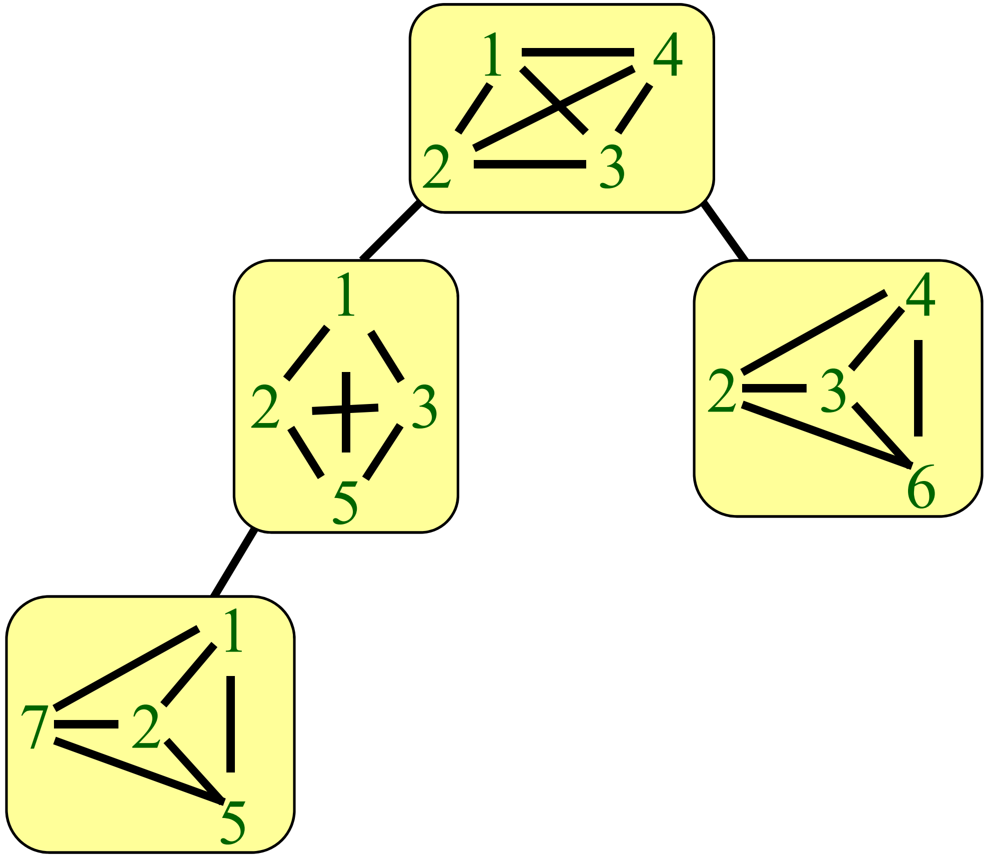



3-tree

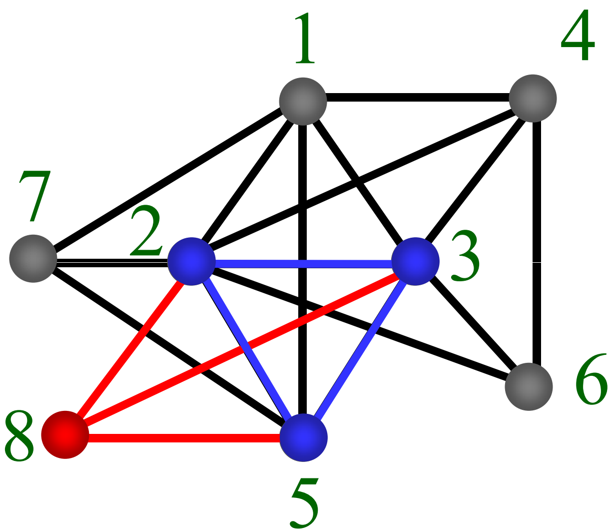



3-tree

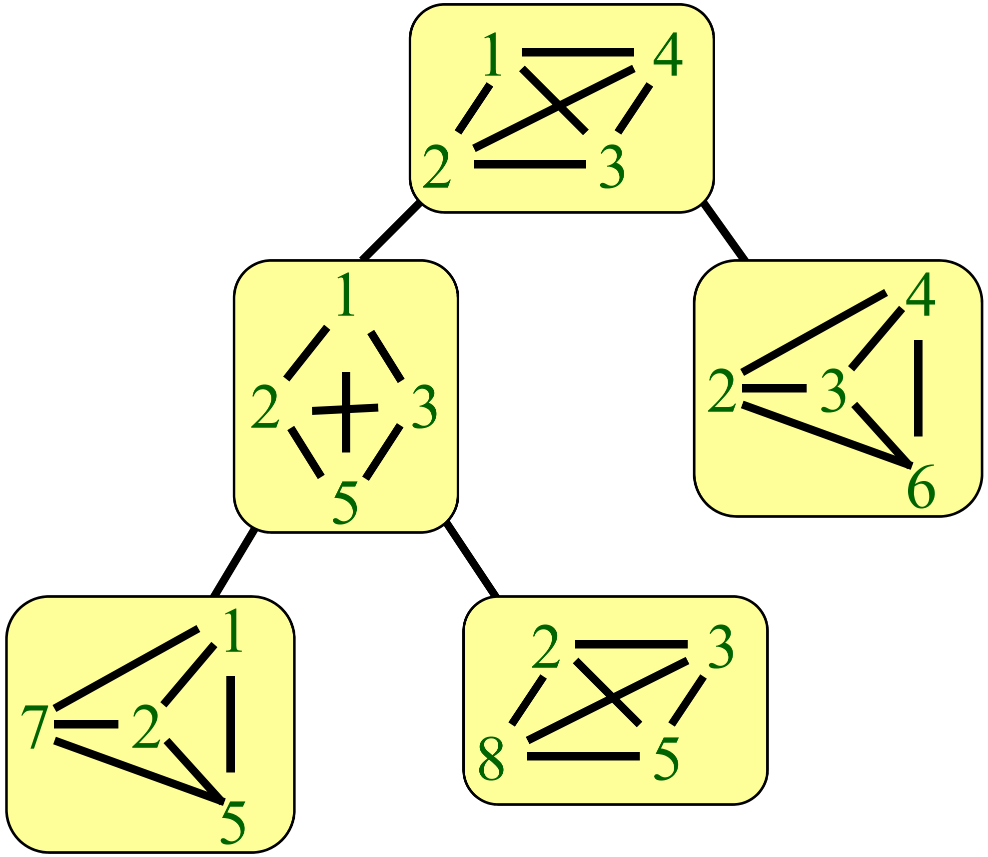
Tree-decomposition

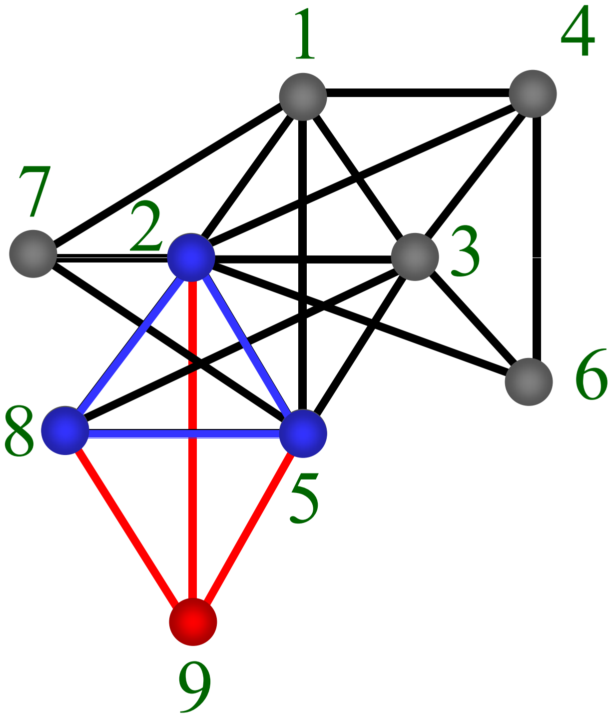


# Tree-decomposition



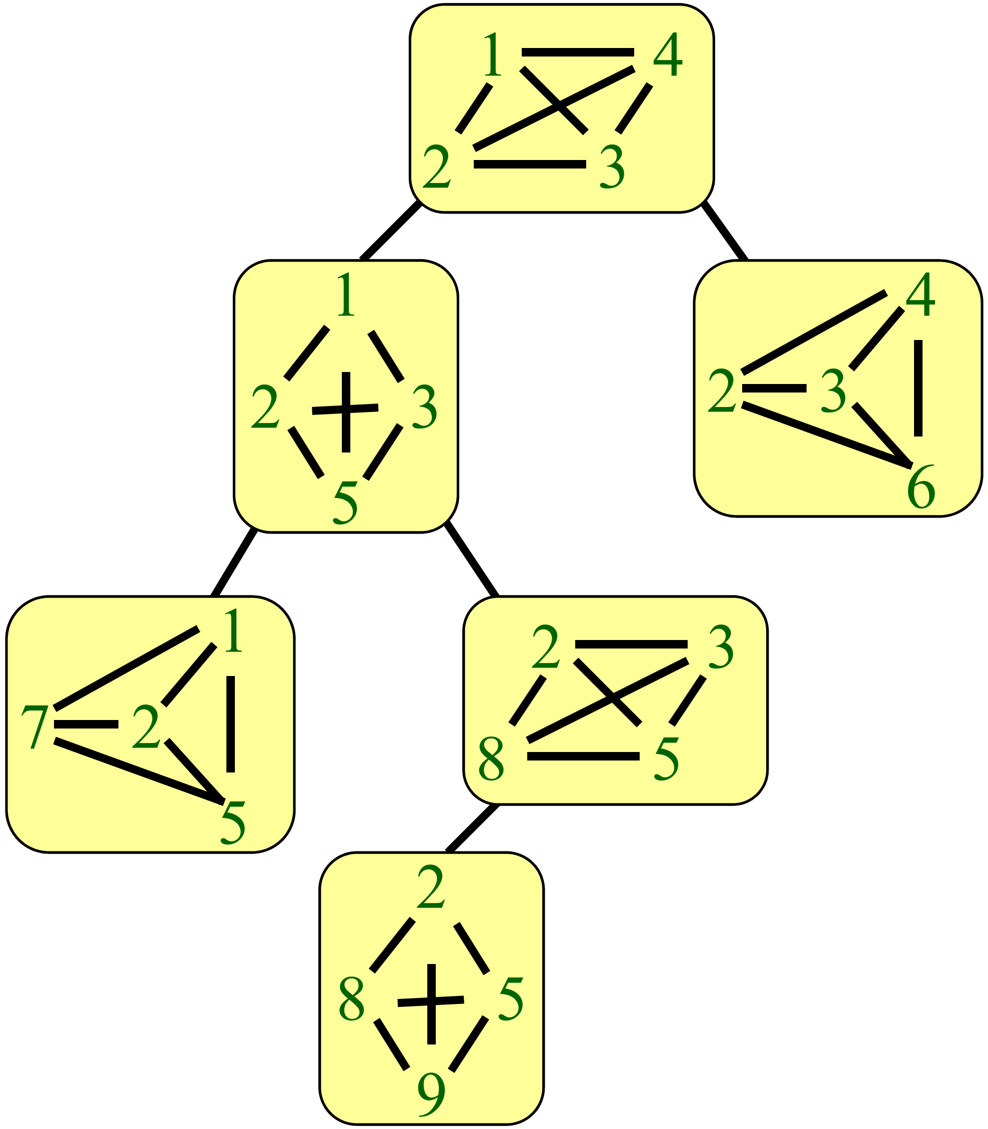

3-tree



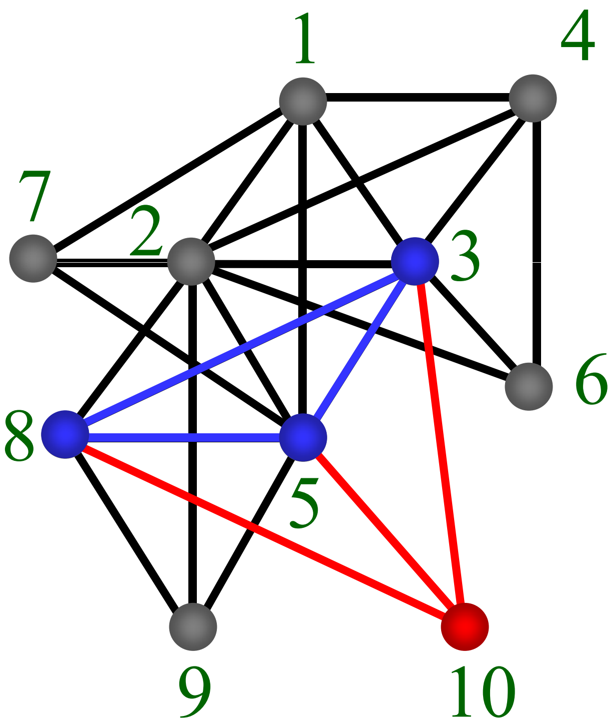



3-tree

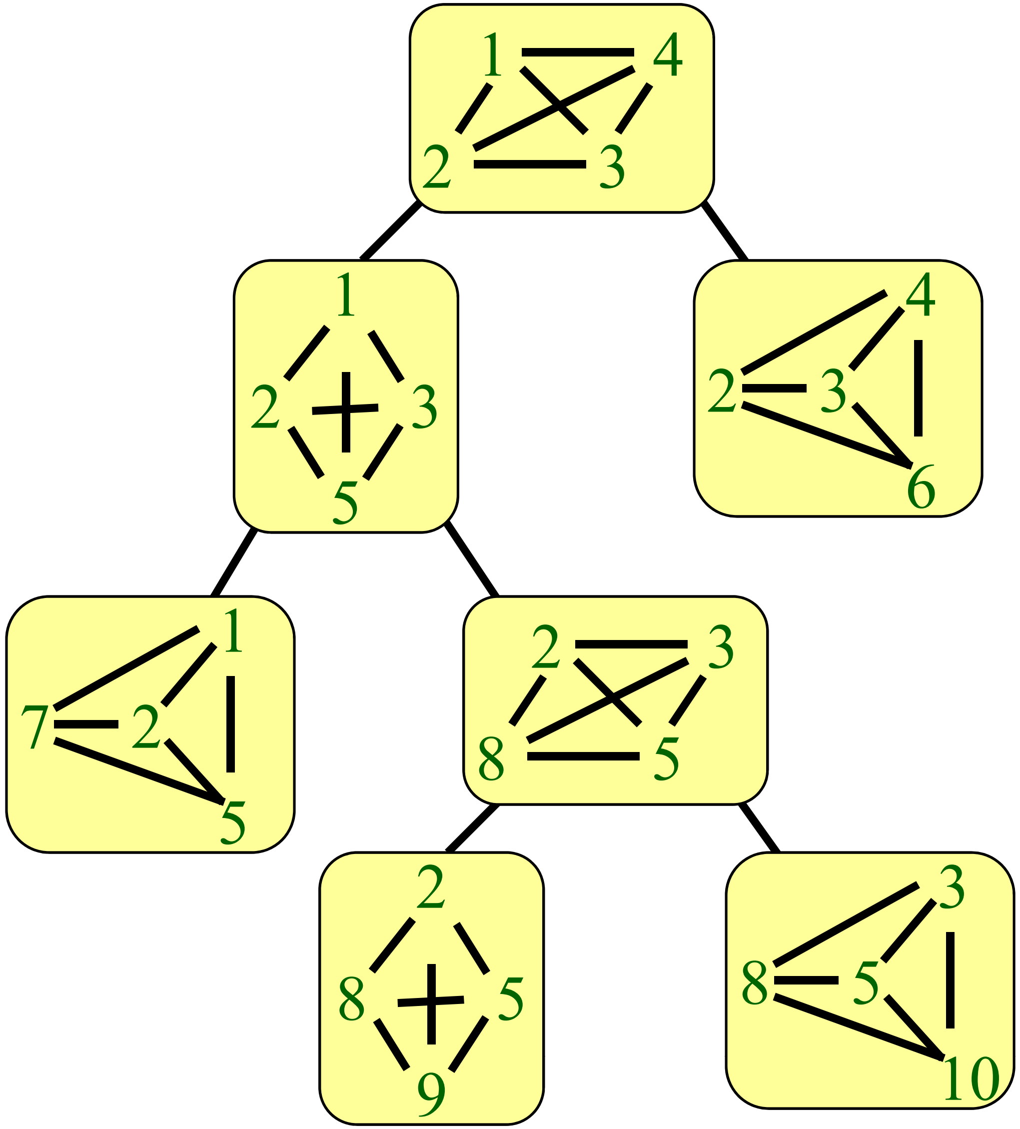
## Tree-decomposition

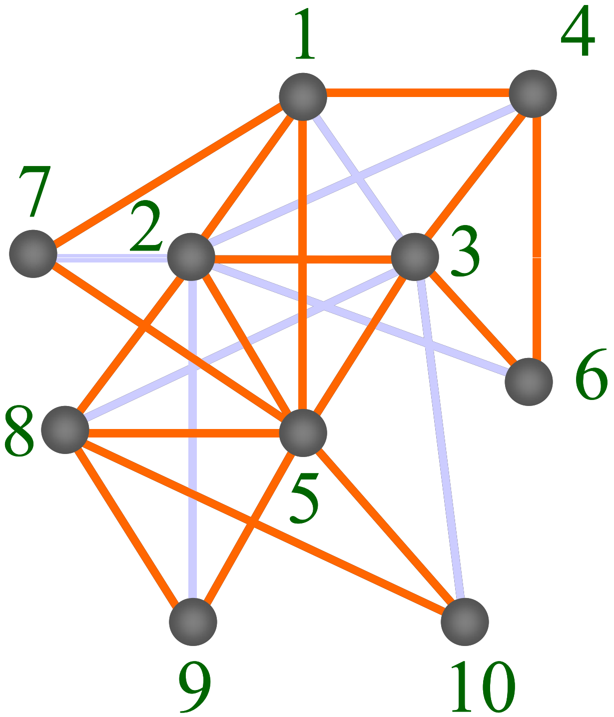


## Tree-decomposition

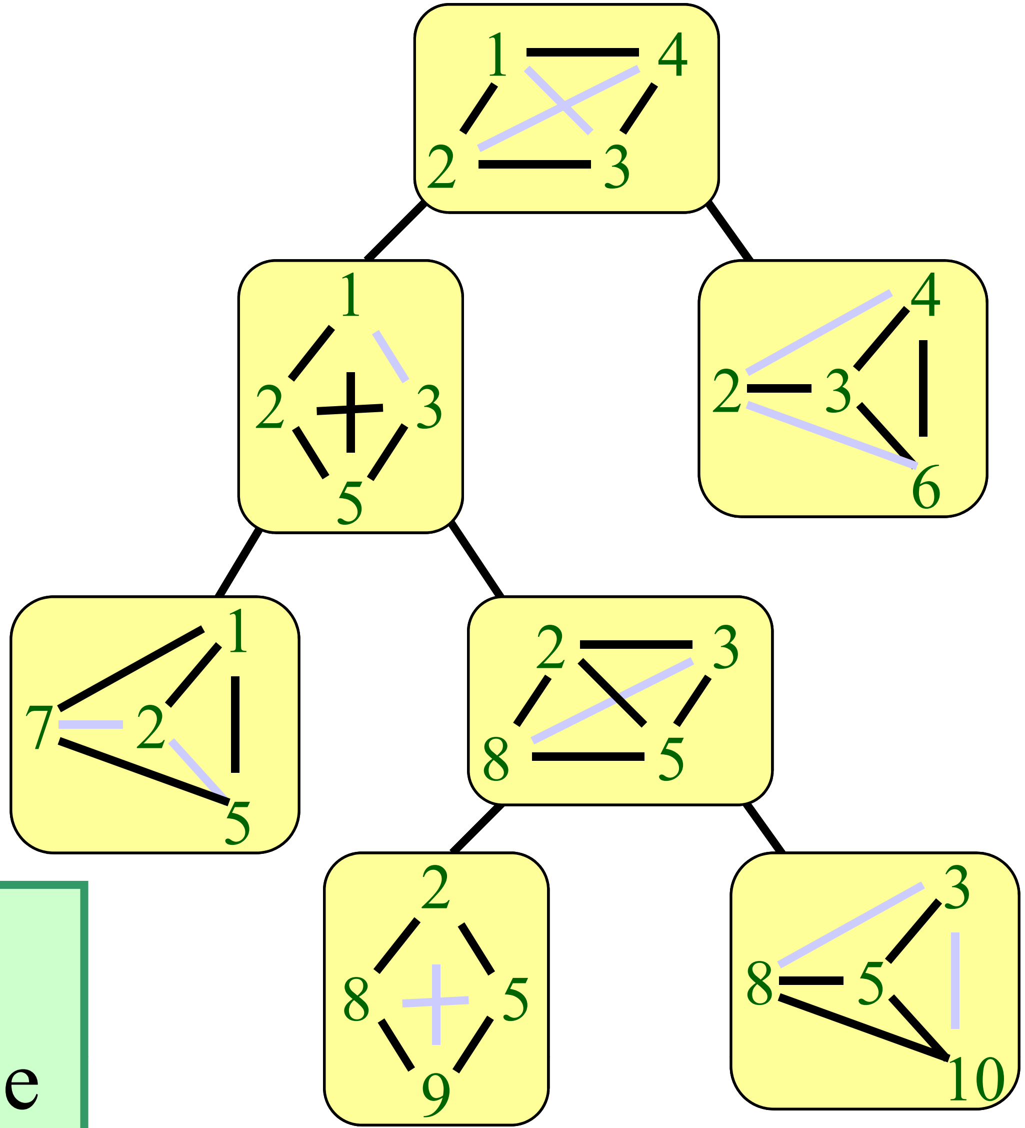



3-tree



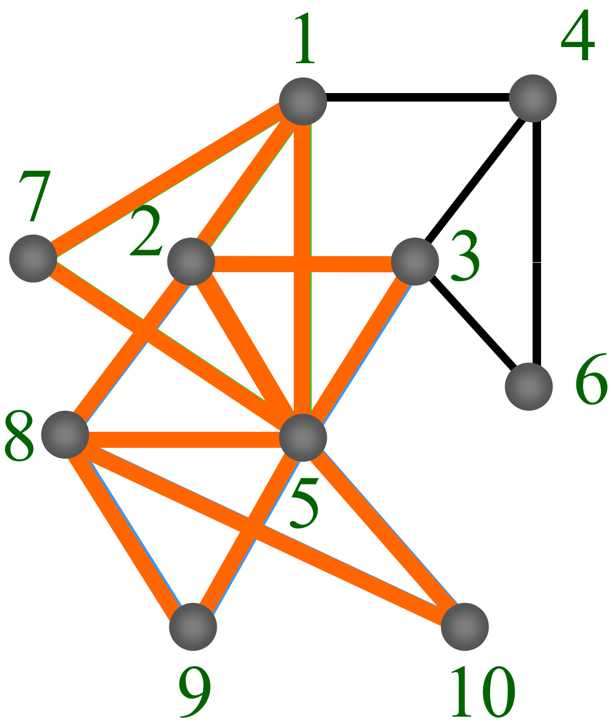



## Tree-decomposition



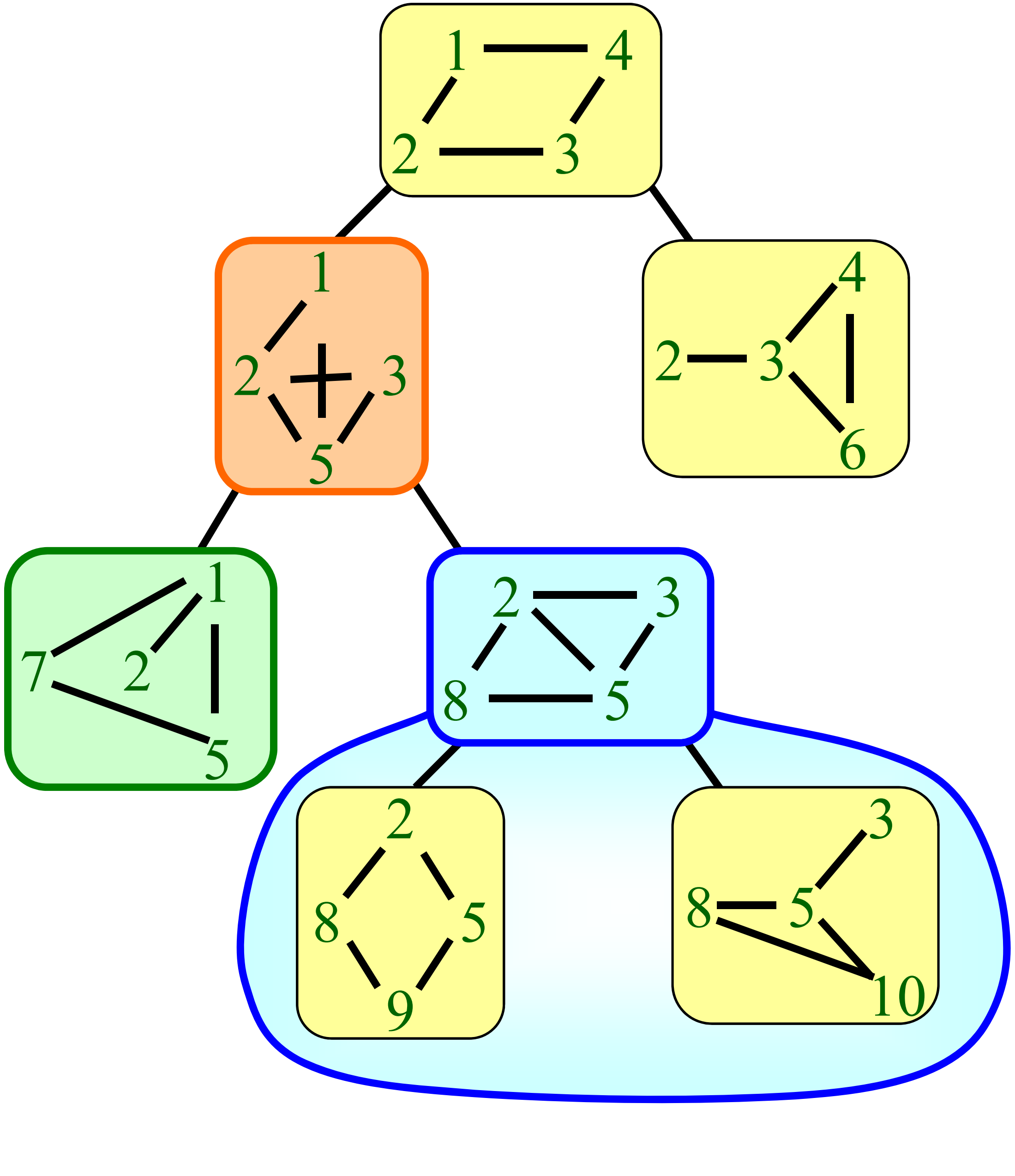

Partial *k*-Tree

a subgraph of *k*-tree



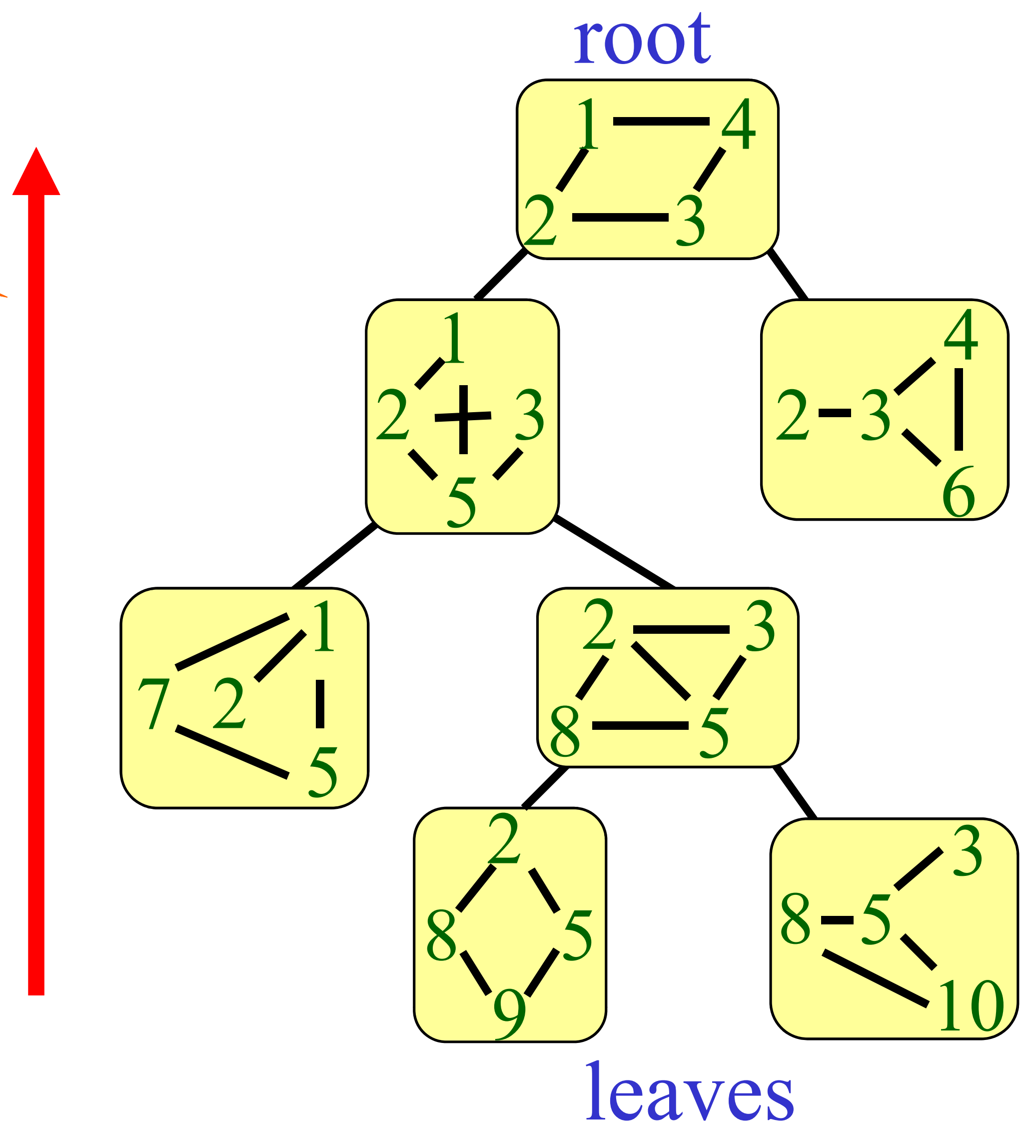

Partial 3-tree

Tree-decomposition



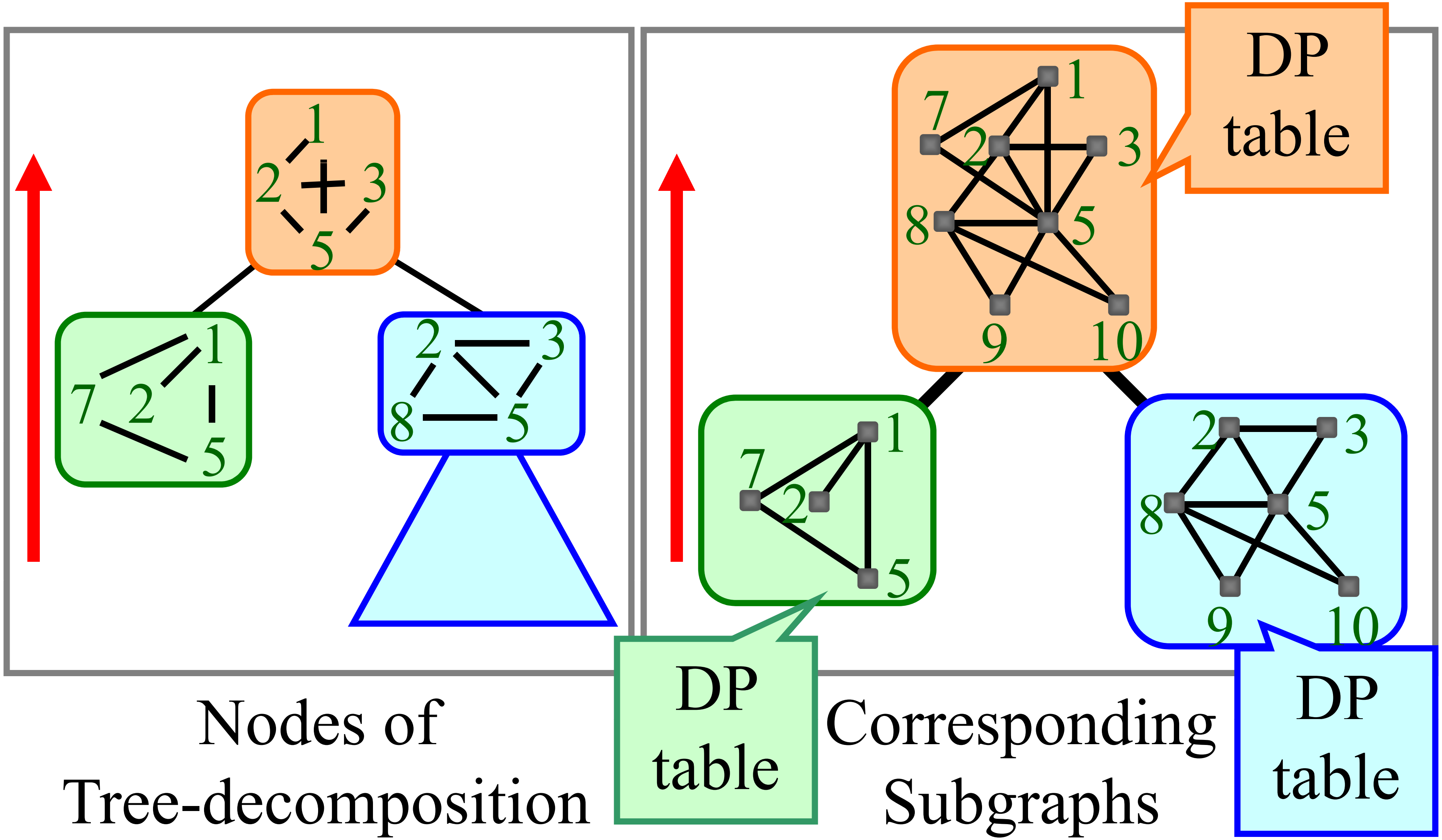
# Algorithm for Partial *k*-Trees

Dynamic Programming

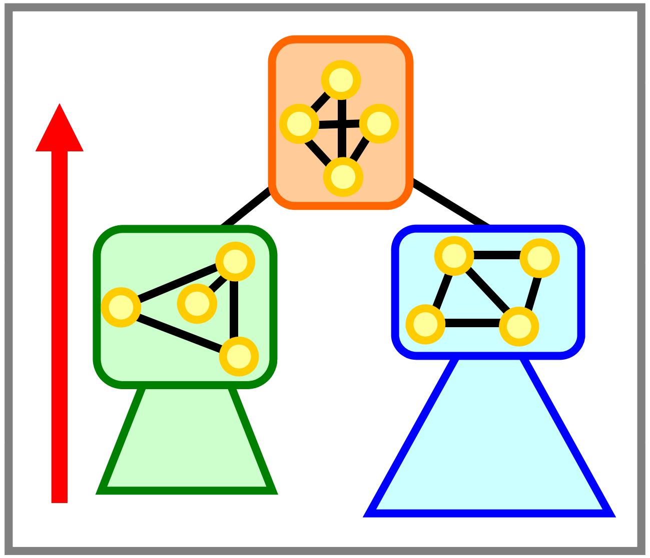


# Algorithm for Partial *k*-Trees

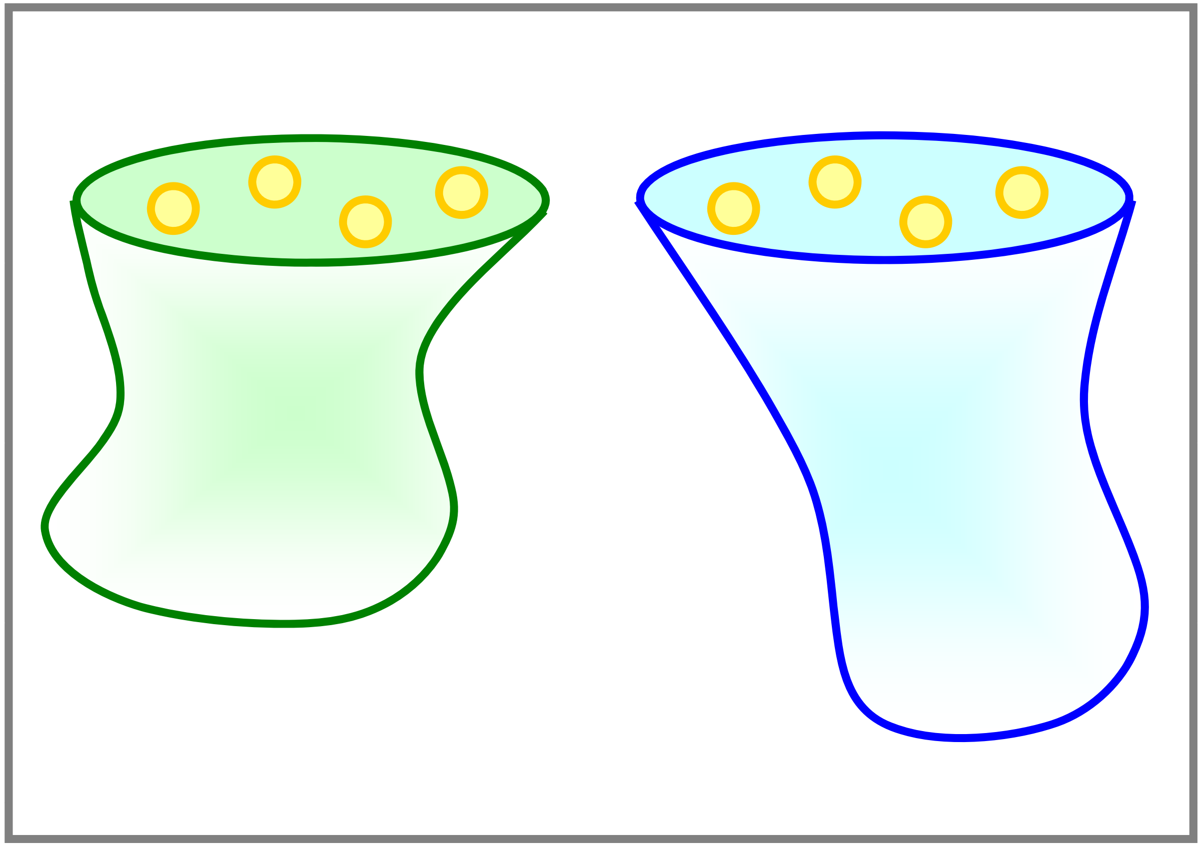
What should we store in the table for the problem ?



# Algorithm for Partial *k*-Trees

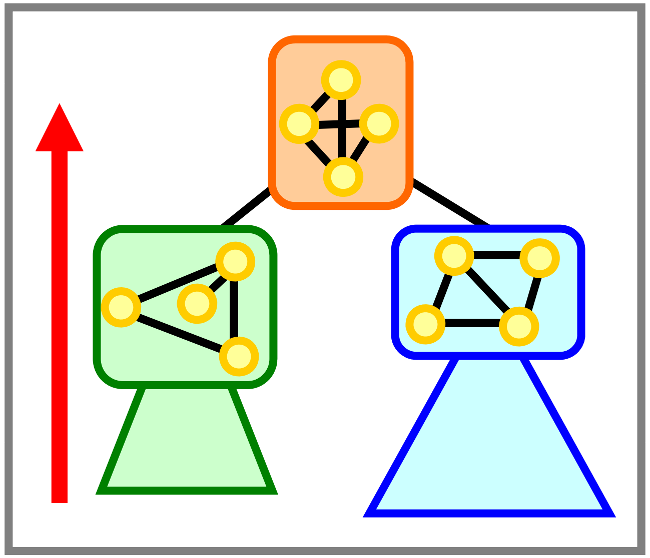


Tree-
Decomposition

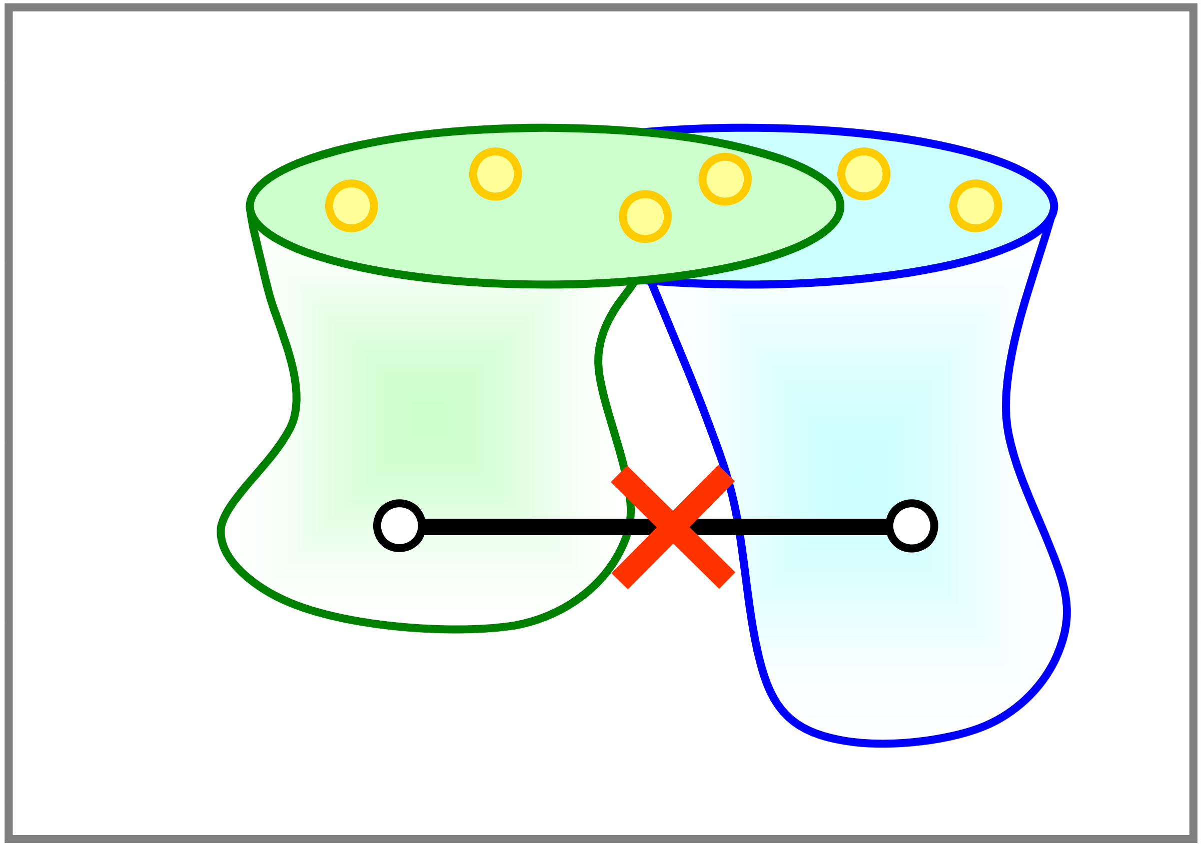


Corresponding Subgraphs

# Algorithm for Partial *k*-Trees



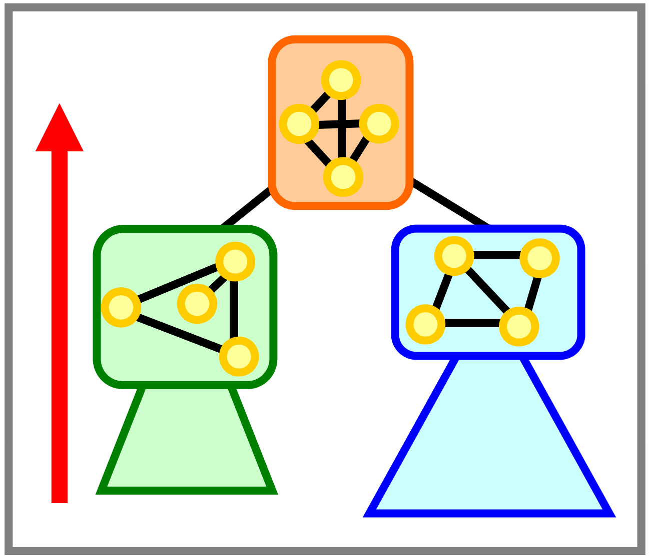
Tree-Decomposition



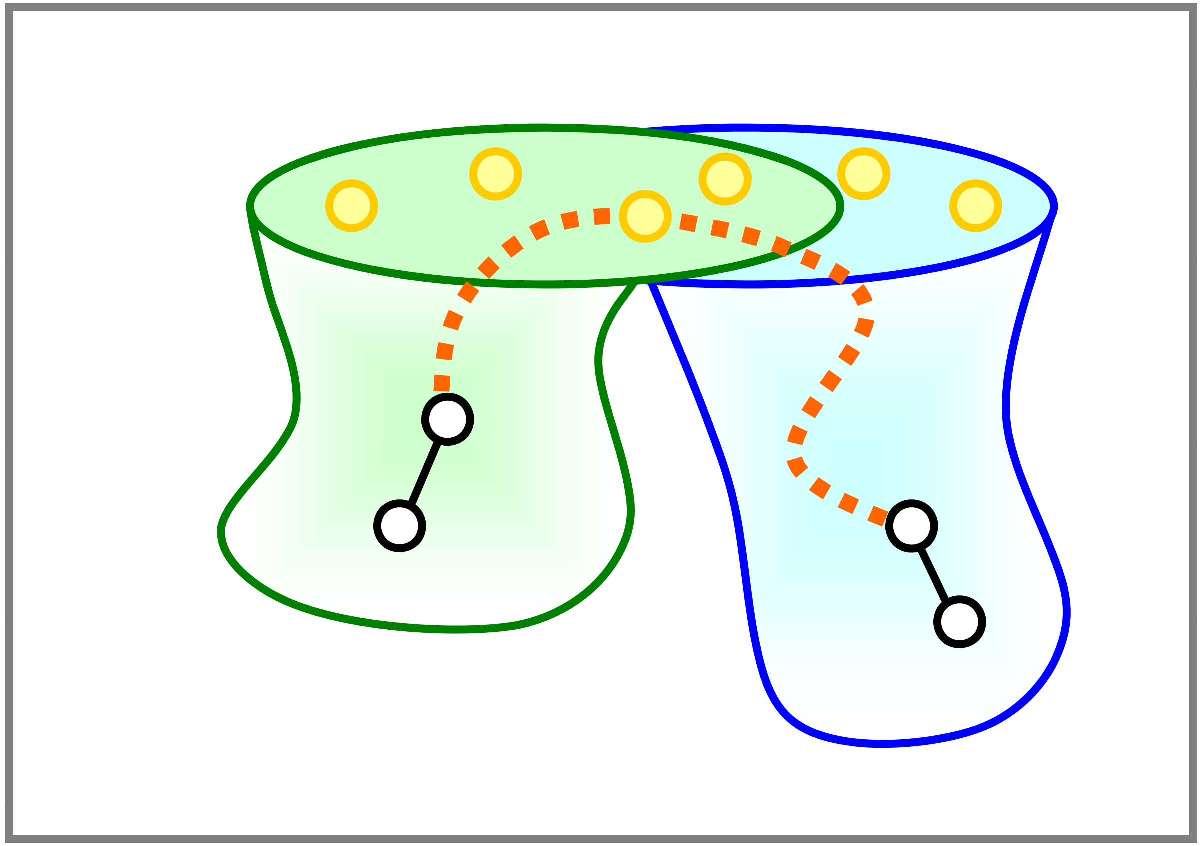
Corresponding Subgraph

# Algorithm for Partial *k*-Trees

store all colors which are assigned to edges with (distance from a vertex) $\leq l$



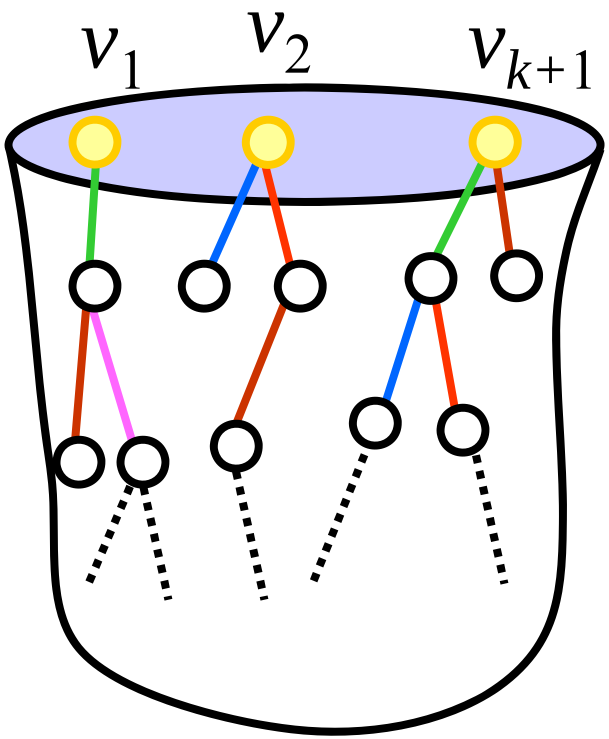
Tree-Decomposition



Corresponding Subgraph

# Algorithm for Partial *k*-Trees

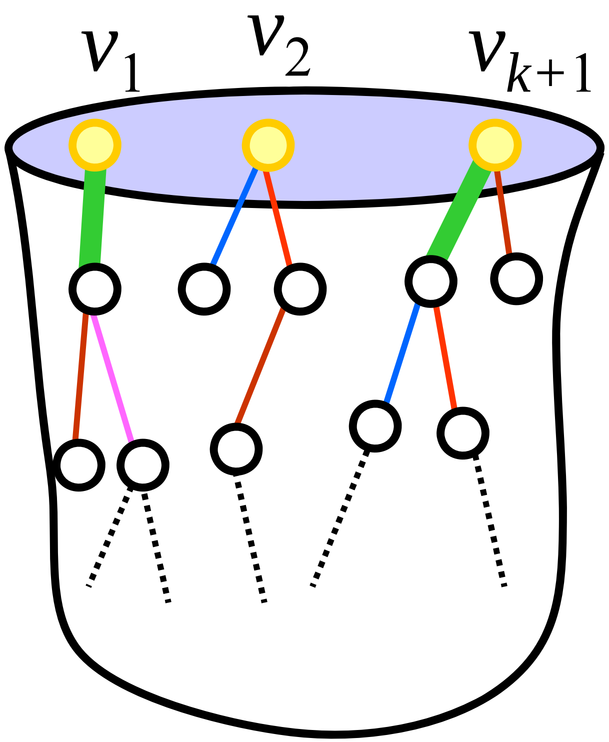
store all colors which are assigned to edges with
(distance from a vertex) $\leq l$




| | $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| color | 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |

# Algorithm for Partial $k$-Trees

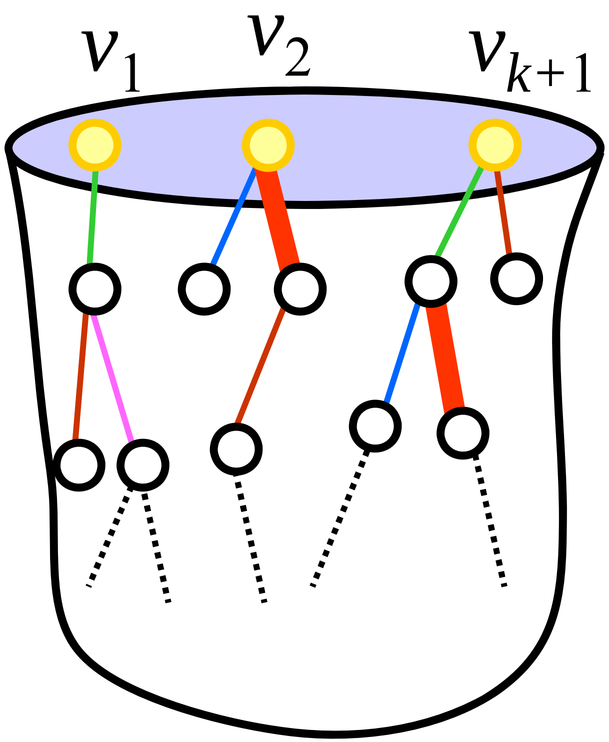
store all colors which are assigned to edges with
(distance from a vertex) $\leq l$




| | $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| color | 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ |
| ■ | ○ | × | ⋯ | × | × | × | ⋯ | × | | ○ | × | ⋯ | × |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |

# Algorithm for Partial *k*-Trees

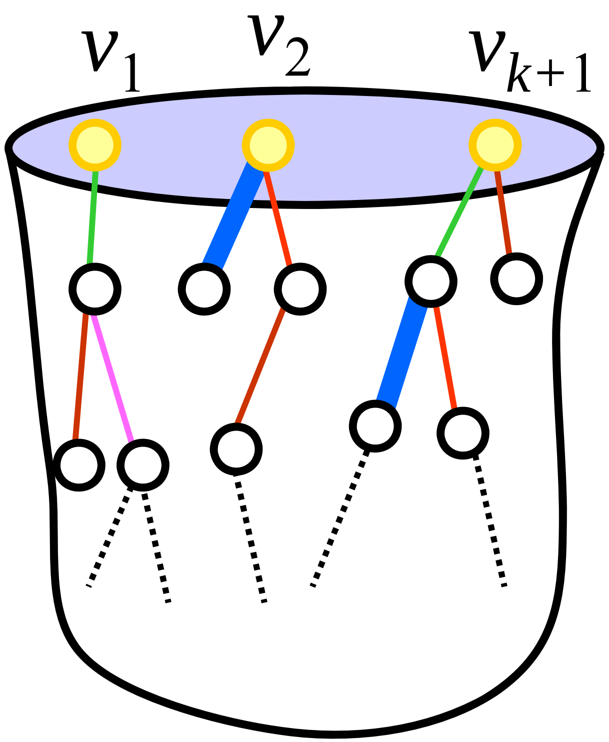
store all colors which are assigned to edges with (distance from a vertex) $\leqq l$




| | $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| color | 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ |
| | ○ | × | … | × | × | × | … | × | | ○ | × | … | × |
| | × | × | … | × | ○ | × | … | × | | × | ○ | … | × |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |

# Algorithm for Partial $k$-Trees

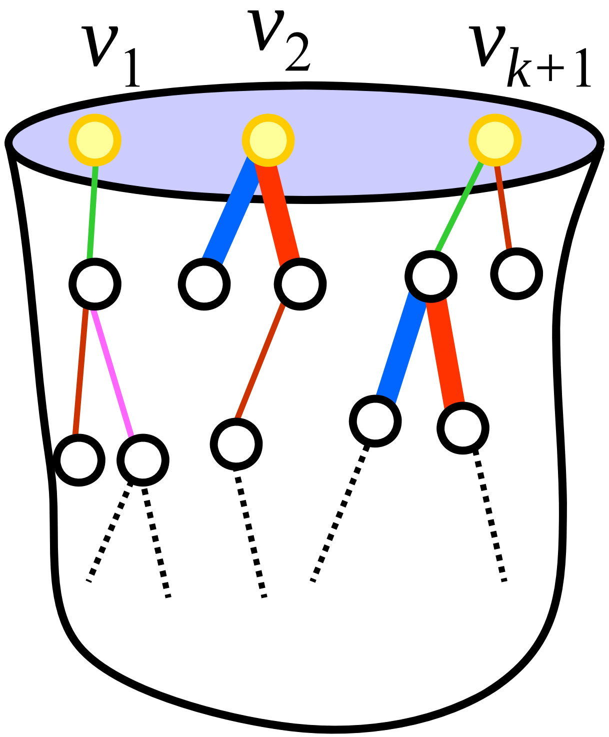
store all colors which are assigned to edges with
(distance from a vertex) $\leq l$




| | $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| color | 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ |
| ■ (green) | ○ | × | ⋯ | × | × | × | ⋯ | × | | ○ | × | ⋯ | × |
| ■ (red) | × | × | ⋯ | × | ○ | × | ⋯ | × | | × | ○ | ⋯ | × |
| ■ (blue) | × | × | ⋯ | × | ○ | × | ⋯ | × | | × | ○ | ⋯ | × |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |

# Algorithm for Partial $k$-Trees

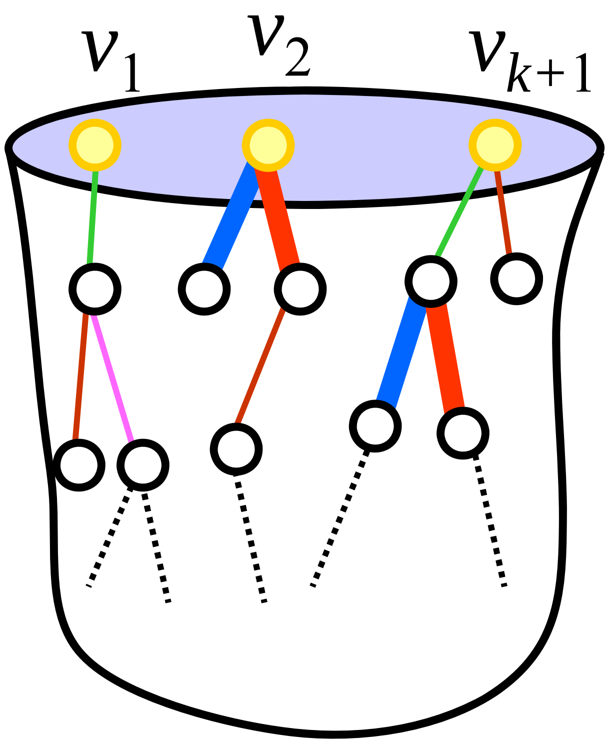
store all colors which are assigned to edges with (distance from a vertex) $\leq l$




| | $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| color | 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ |
| ■ (green) | ○ | × | … | × | × | × | … | × | | ○ | × | … | × |
| ■ (red) | × | × | … | × | ○ | × | … | × | | × | ○ | … | × |
| ■ (blue) | × | × | … | × | ○ | × | … | × | | × | ○ | … | × |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |

# Algorithm for Partial *k*-Trees

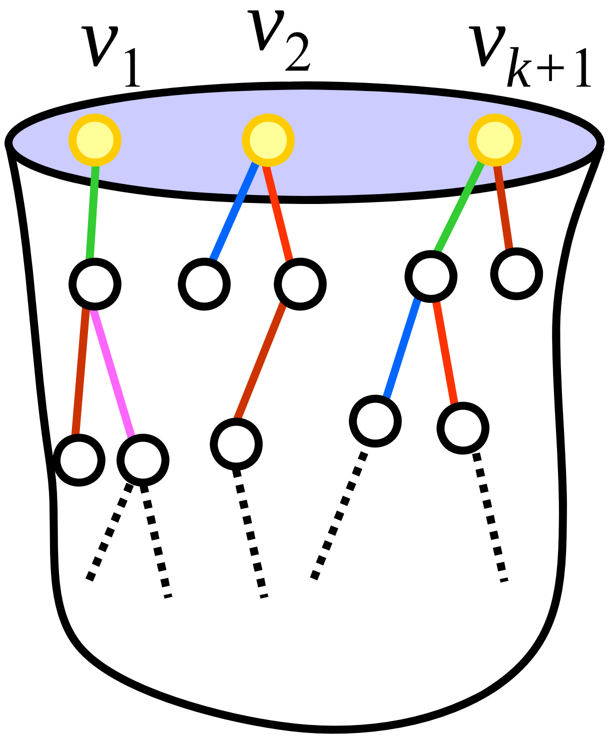
store all colors which are assigned to edges with
(distance from a vertex) $\leq l$



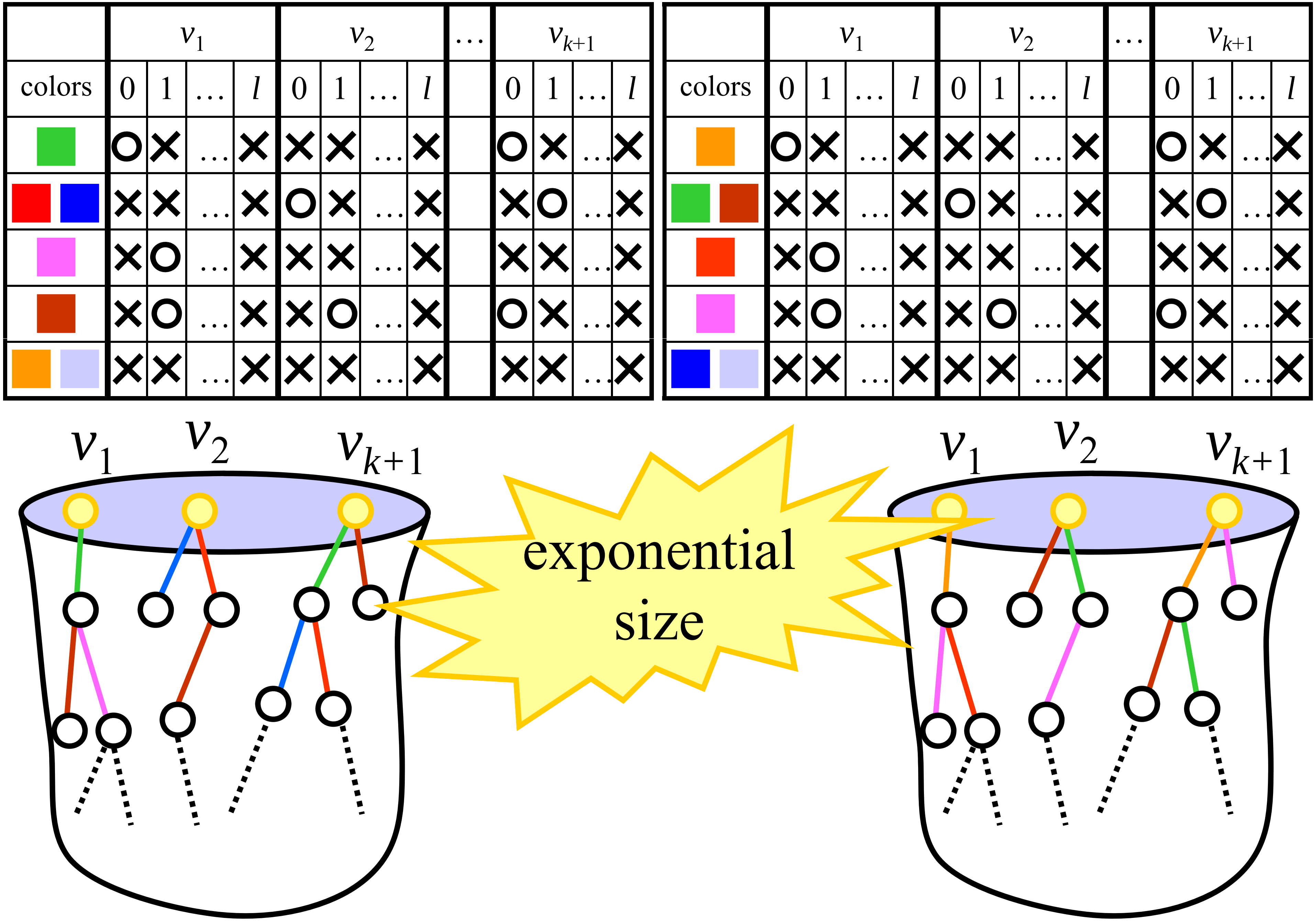

| | $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| colors | 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ |
| ■ | ○ | × | ⋯ | × | × | × | ⋯ | × | | ○ | × | ⋯ | × |
| ■ ■ | × | × | ⋯ | × | ○ | × | ⋯ | × | | × | ○ | ⋯ | × |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |

# Algorithm for Partial *k*-Trees

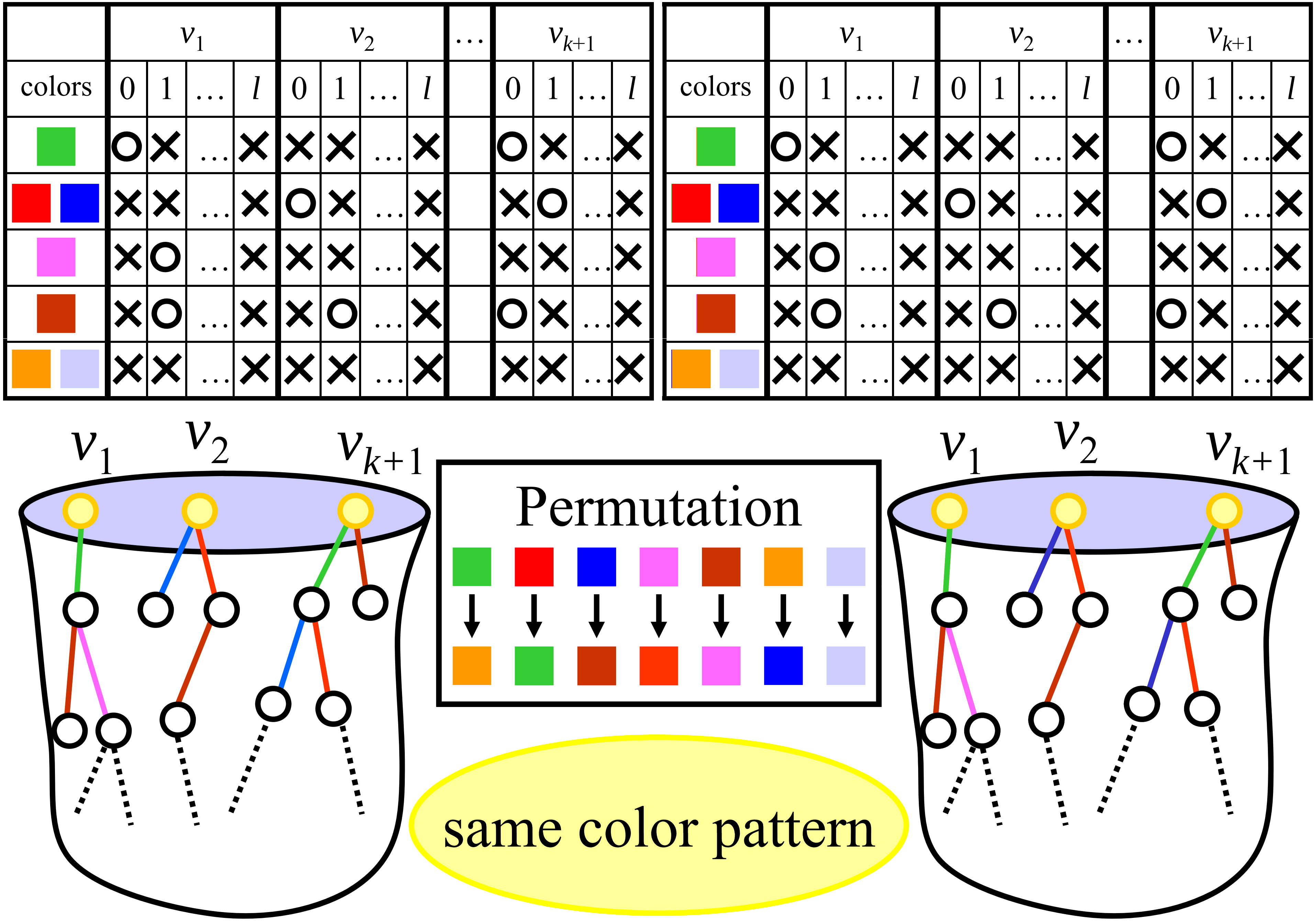
store all colors which are assigned to edges with
(distance from a vertex) $\leq l$



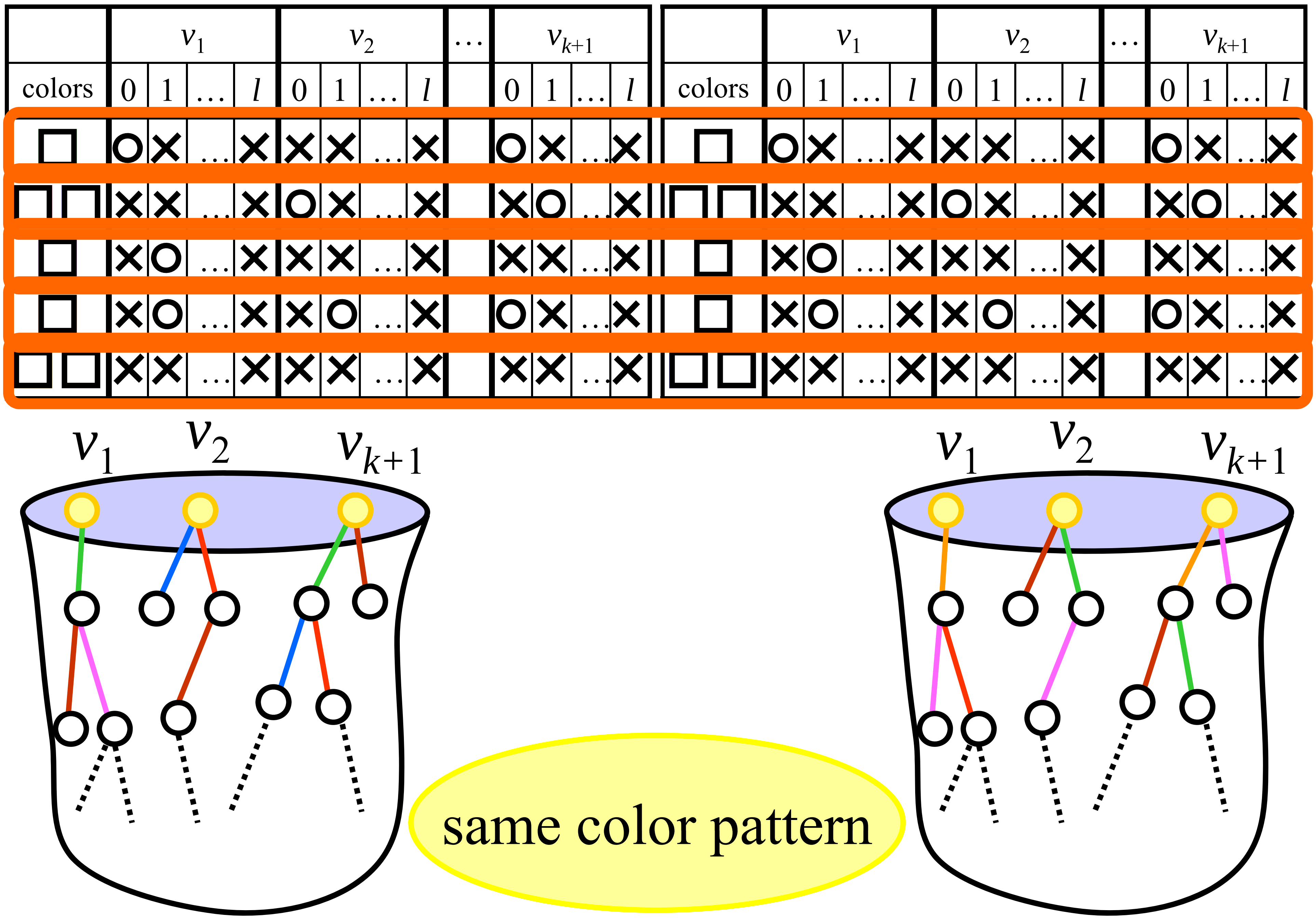

| | $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| colors | 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ |
| | ○ | × | ⋯ | × | × | × | ⋯ | × | | ○ | × | ⋯ | × |
| | × | × | ⋯ | × | ○ | × | ⋯ | × | | × | ○ | ⋯ | × |
| | × | ○ | ⋯ | × | × | × | ⋯ | × | | × | × | ⋯ | × |
| | × | ○ | ⋯ | × | × | ○ | ⋯ | × | | ○ | × | ⋯ | × |
| | × | × | ⋯ | × | × | × | ⋯ | × | | × | × | ⋯ | × |



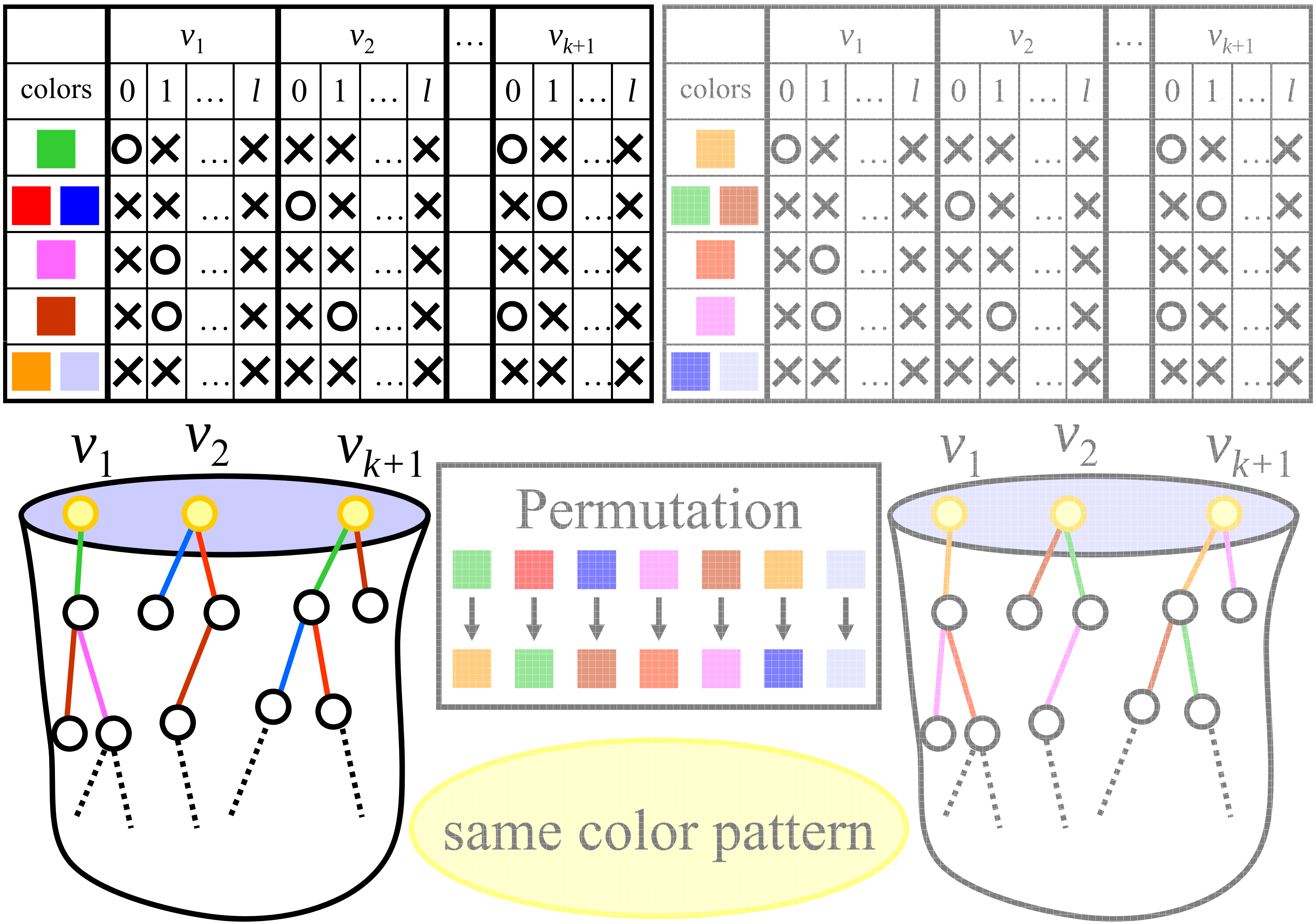
$v_1$
$v_2$
$\dots$
$v_{k+1}$
colors
0
1
$\dots$
$l$
exponential
size



colors
$v_1$
$v_2$
…
$v_{k+1}$
0
1
…
$l$
Permutation
same color pattern

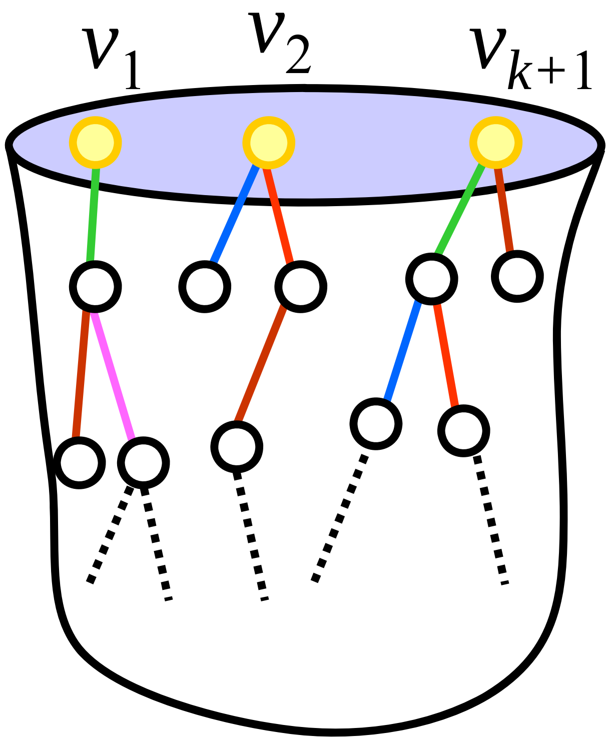


colors
v1
v2
vk+1
0
1
l
same color pattern



colors
v1
v2
…
vk+1
0
1
…
l
Permutation
same color pattern

| $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ | # of colors |
| ○ | × | … | × | × | × | … | × | | ○ | × | … | × | 1 |
| × | × | … | × | ○ | × | … | × | | × | ○ | … | × | 2 |
| × | ○ | … | × | × | × | … | × | | × | × | … | × | 1 |
| × | ○ | … | × | × | ○ | … | × | | ○ | × | … | × | 1 |
| × | × | … | × | × | × | … | × | | × | × | … | × | 2 |



# Algorithm for Partial $k$-Trees

| $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ | # |
| ○ | × | … | × | × | × | … | × | | ○ | × | … | × | 1 |
| × | × | … | × | ○ | × | … | × | | × | ○ | … | × | 2 |
| × | ○ | … | × | × | × | … | × | | × | × | … | × | 1 |
| × | ○ | … | × | × | ○ | … | × | | ○ | × | … | × | 1 |
| × | × | … | × | × | × | … | × | | × | × | … | × | 2 |
| other sets of {○, ×} | | | | | | | | | | | | | 0 |

$2^{O(1)}$

$(k+1)(l+1) = O(1)$

# Algorithm for Partial $k$-Trees

$2^{O(1)}$

| $v_1$ | | | | $v_2$ | | | | … | $v_{k+1}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ | # |
| ○ | × | … | × | × | × | … | × | | ○ | × | … | × | 1 |
| × | × | … | × | ○ | × | … | × | | × | ○ | … | × | 2 |
| × | ○ | … | × | × | × | … | × | | × | × | … | × | 1 |
| × | ○ | … | × | × | ○ | … | × | | ○ | × | … | × | 1 |
| × | × | … | × | × | × | … | × | | × | × | … | × | 2 |
| other sets of {○ , ×} | | | | | | | | | | | | | 0 |

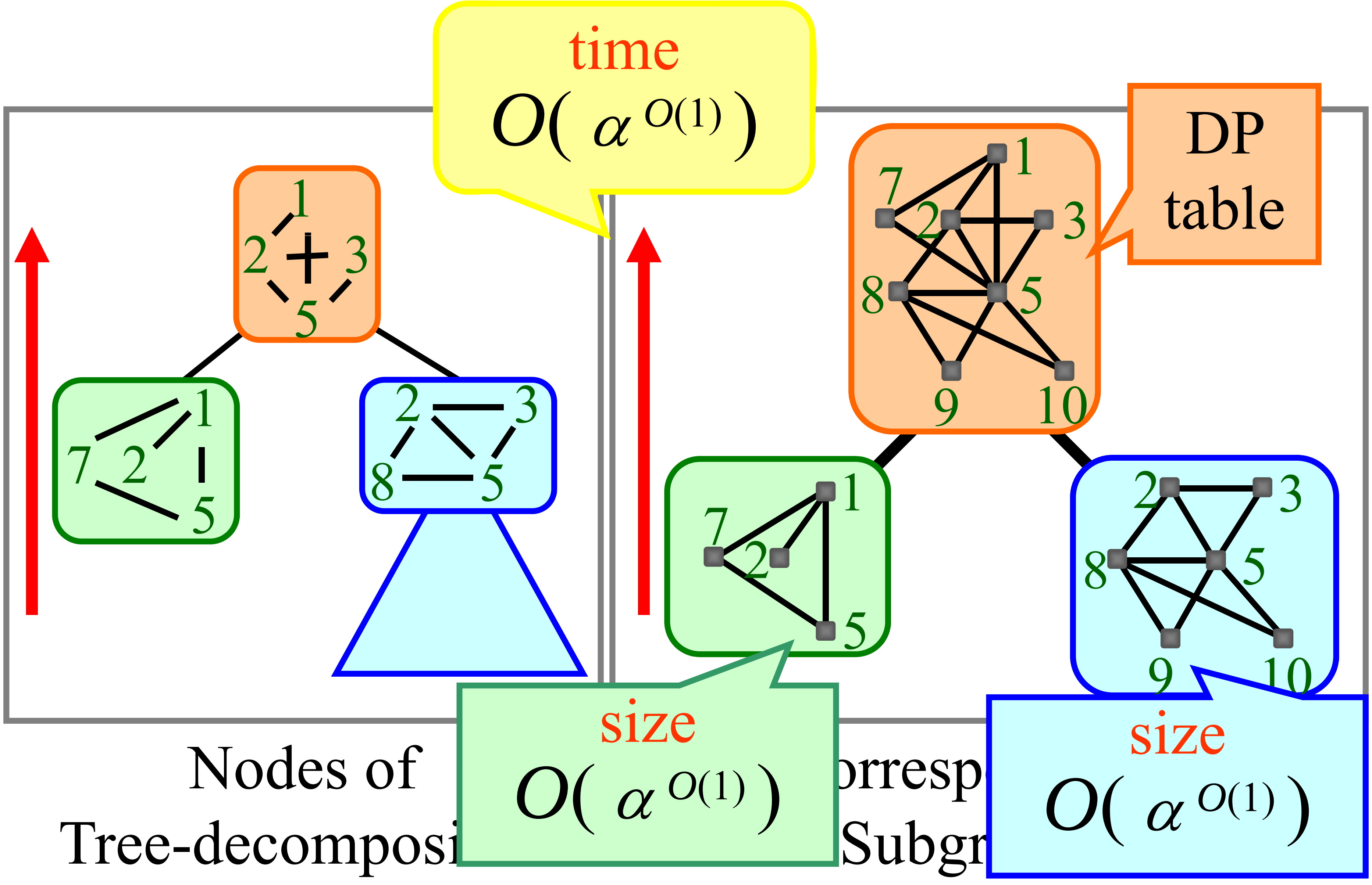
color pattern: $2^{O(1)}$ rows ⟶ $\{0, 1, \dots, \alpha\}$

# Algorithm for Partial $k$-Trees

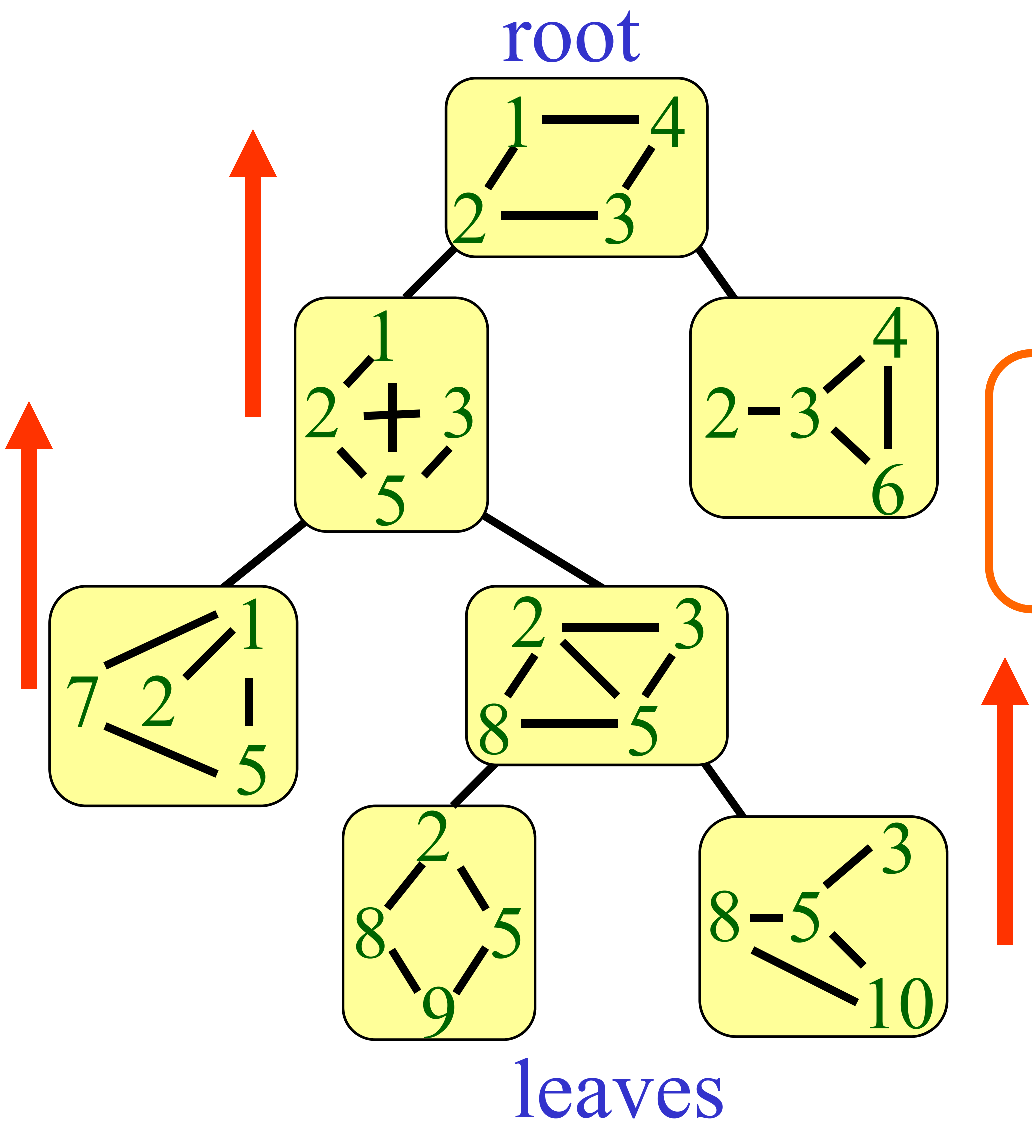
| $v_1$ | | | | $v_2$ | | | | $\cdots$ | $v_{k+1}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | … | $l$ | 0 | 1 | … | $l$ | | 0 | 1 | … | $l$ | # |
| ○ | × | … | × | × | × | … | × | | ○ | × | … | × | 3 |
| × | × | … | × | ○ | × | … | × | | × | ○ | … | × | 1 |
| × | ○ | … | × | × | × | … | × | | × | × | … | × | [illegible] |
| × | ○ | … | × | × | ○ | … | × | | ○ | × | … | × | [illegible] |
| × | × | … | × | × | × | … | × | | × | × | … | × | 1 |
| other sets of {○ , ×} | | | | | | | | | | | | | 0 |

$O(\alpha^{O(1)})$

# Algorithm for Partial *k*-Trees



# Algorithm for Partial $k$-Trees



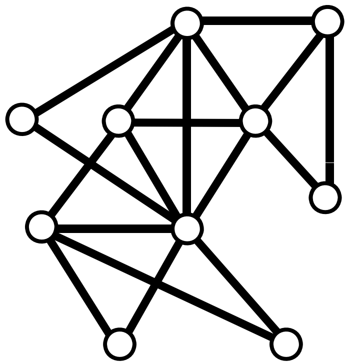

# of nodes $= O(n)$

time $O(\alpha^{O(1)})$

time $O(n\,\alpha^{O(1)})$

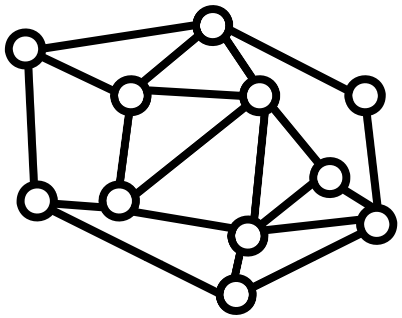
# Our Results

Partial *k*-Trees



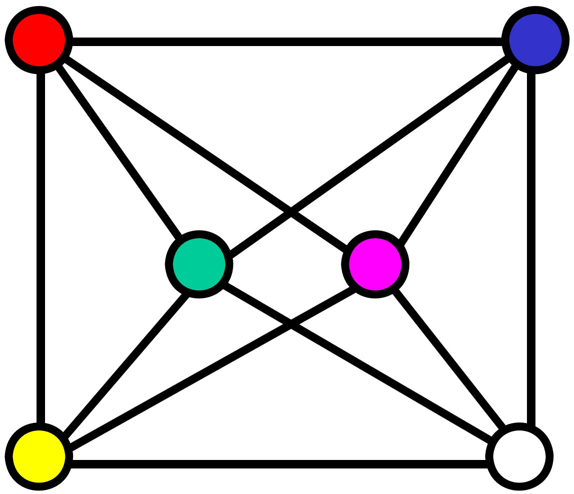
Polynomial-time exact algorithm

Planar Graphs

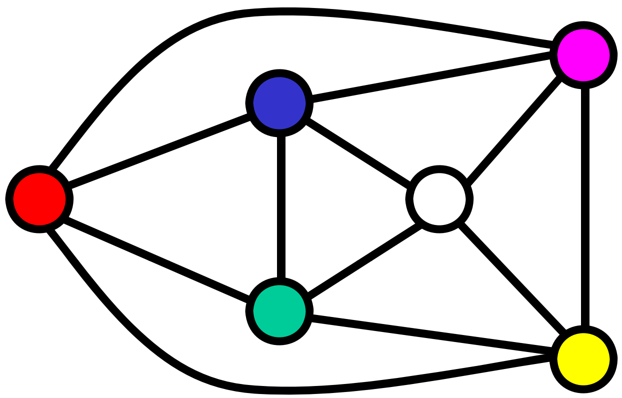


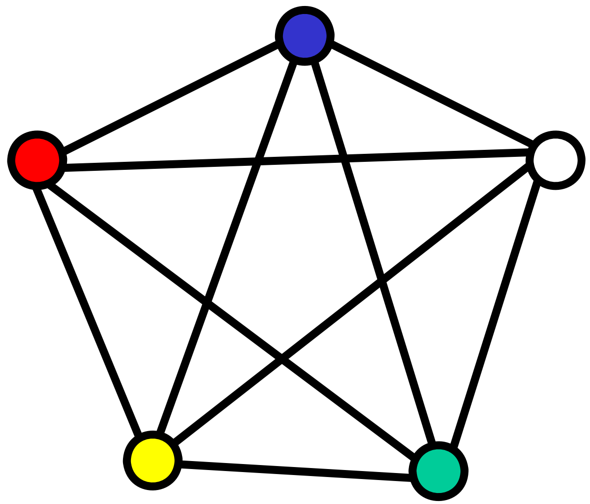
Polynomial-time
2-approximation algorithm

# Planar Graphs

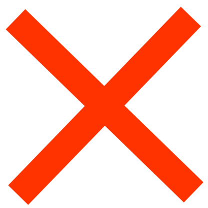


planar graph



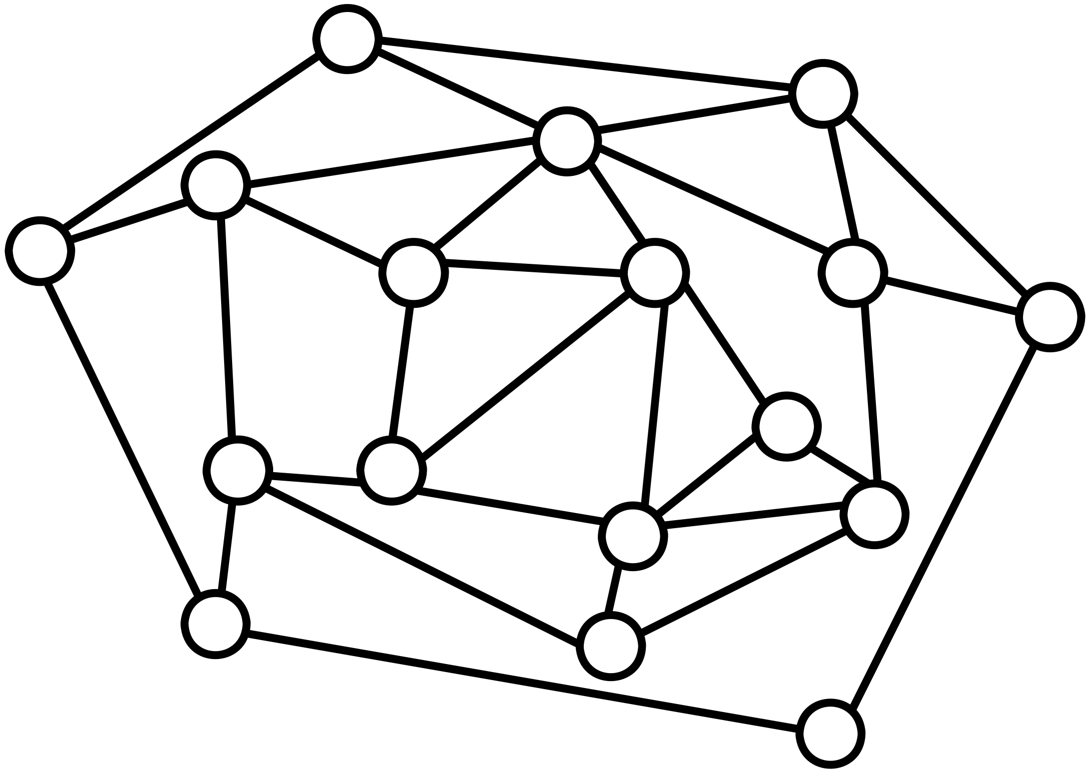


non-planar graph



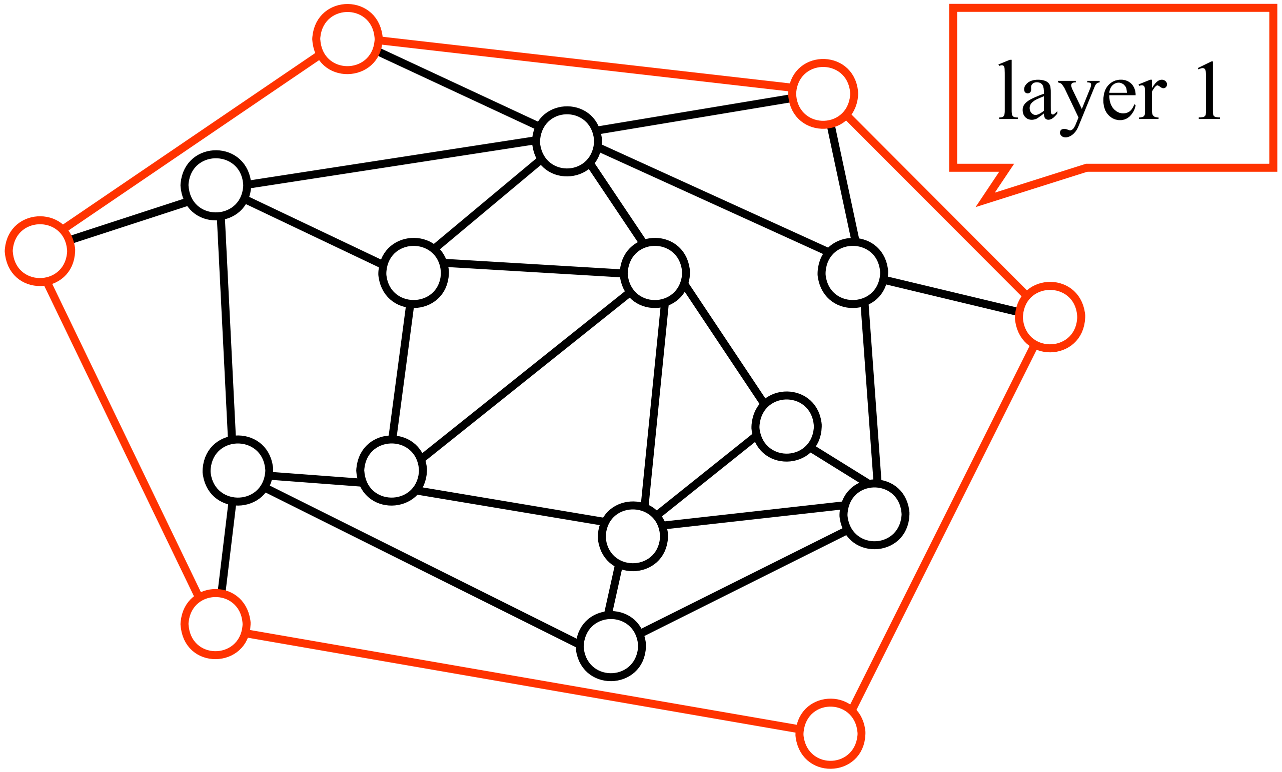
# 2-Approx. Algorithm for Planar Graphs

Planar graph can be partitioned into layers



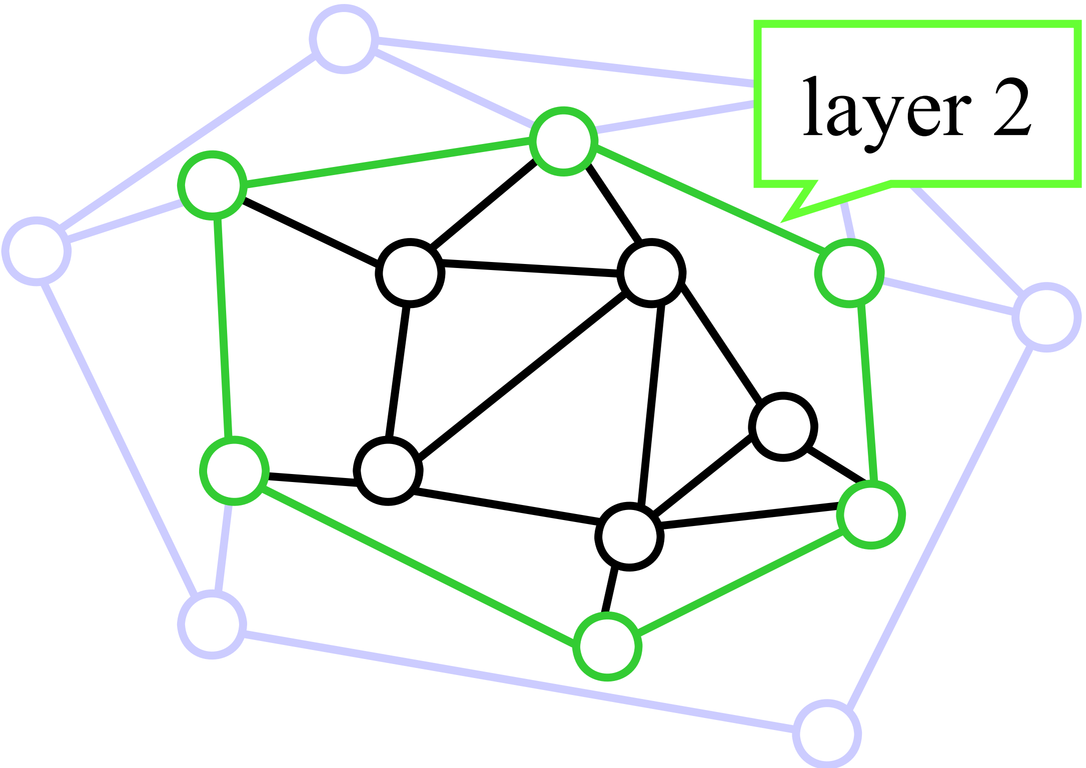
# 2-Approx. Algorithm for Planar Graphs

Planar graph can be partitioned into layers



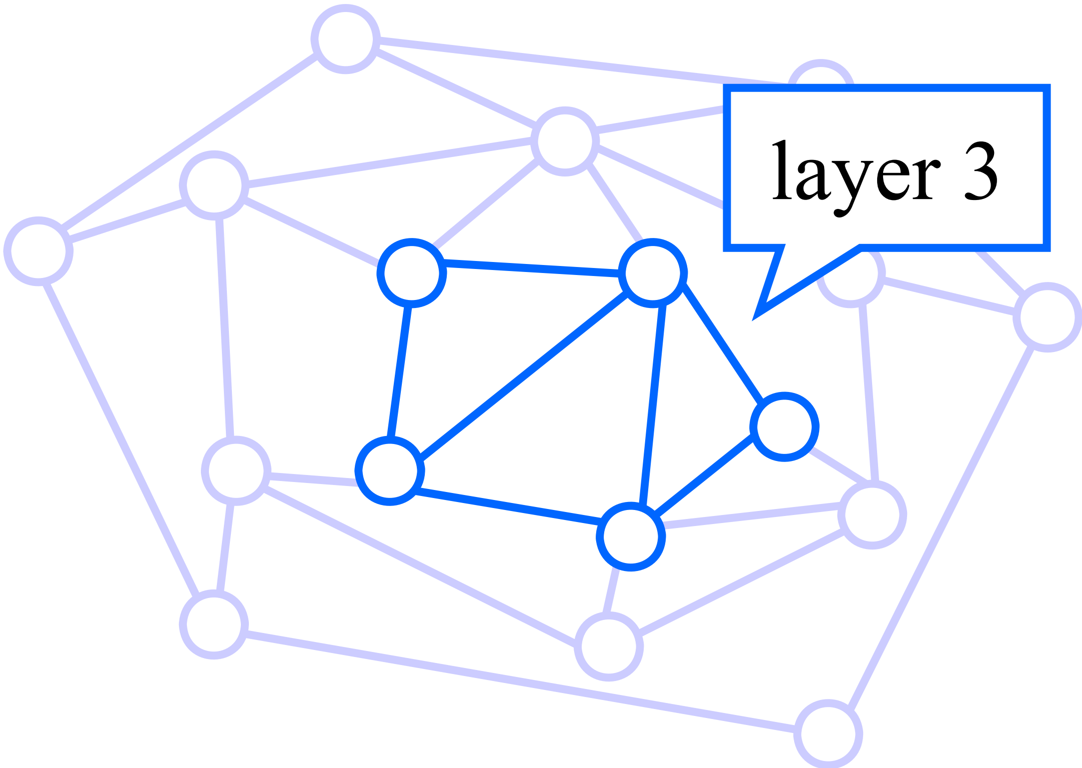
# 2-Approx. Algorithm for Planar Graphs

Planar graph can be partitioned into layers



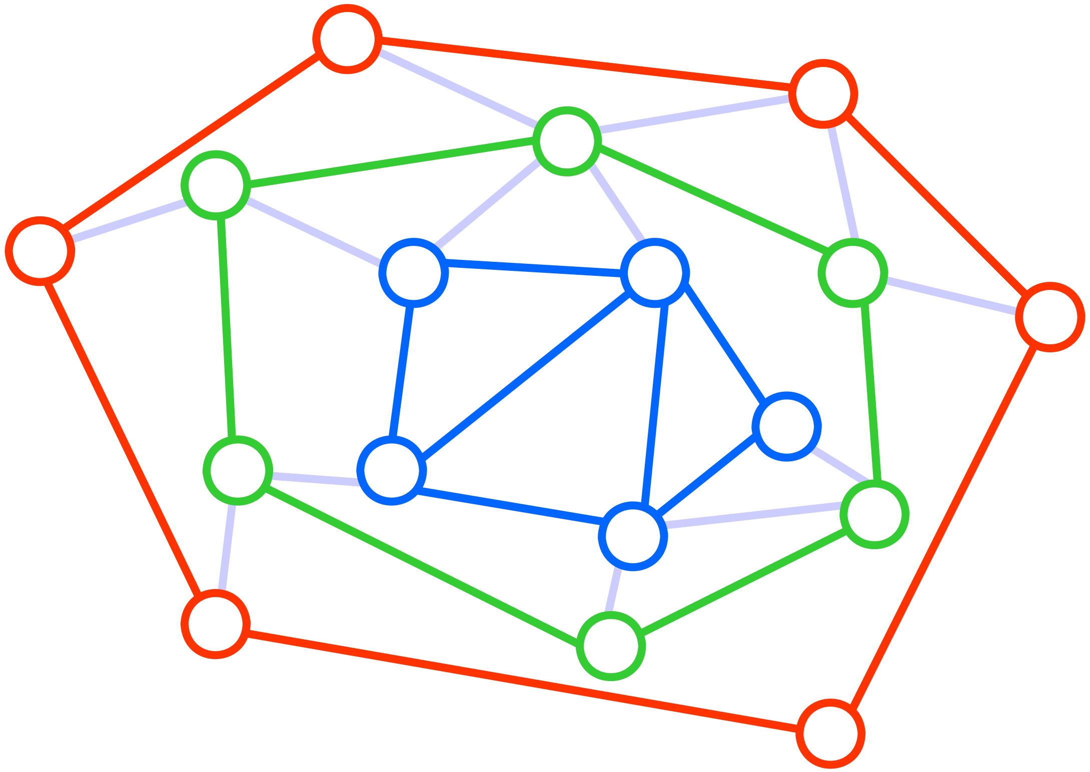
# 2-Approx. Algorithm for Planar Graphs

Planar graph can be partitioned into layers

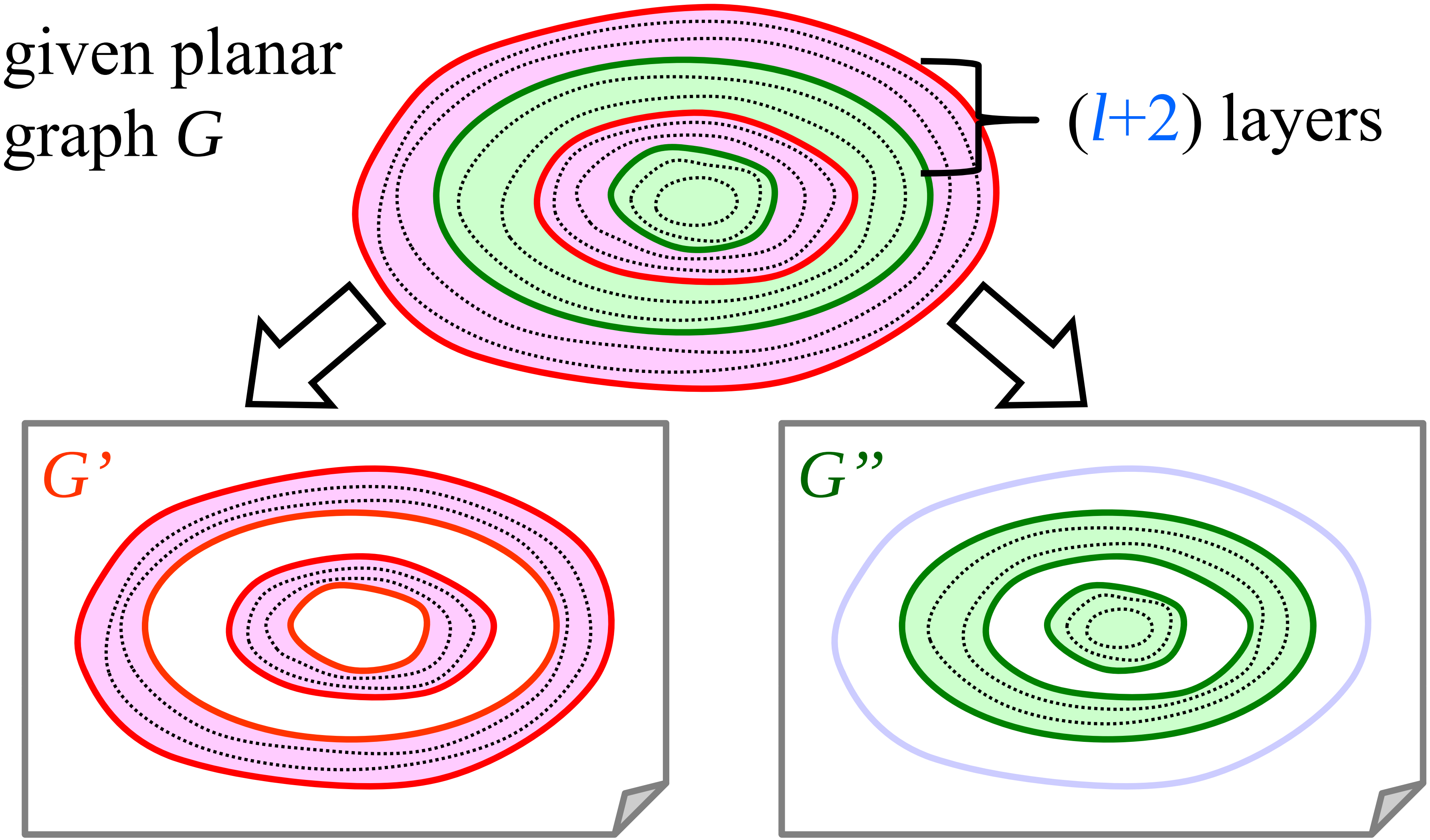


# 2-Approx. Algorithm for Planar Graphs

Planar graph can be partitioned into layers

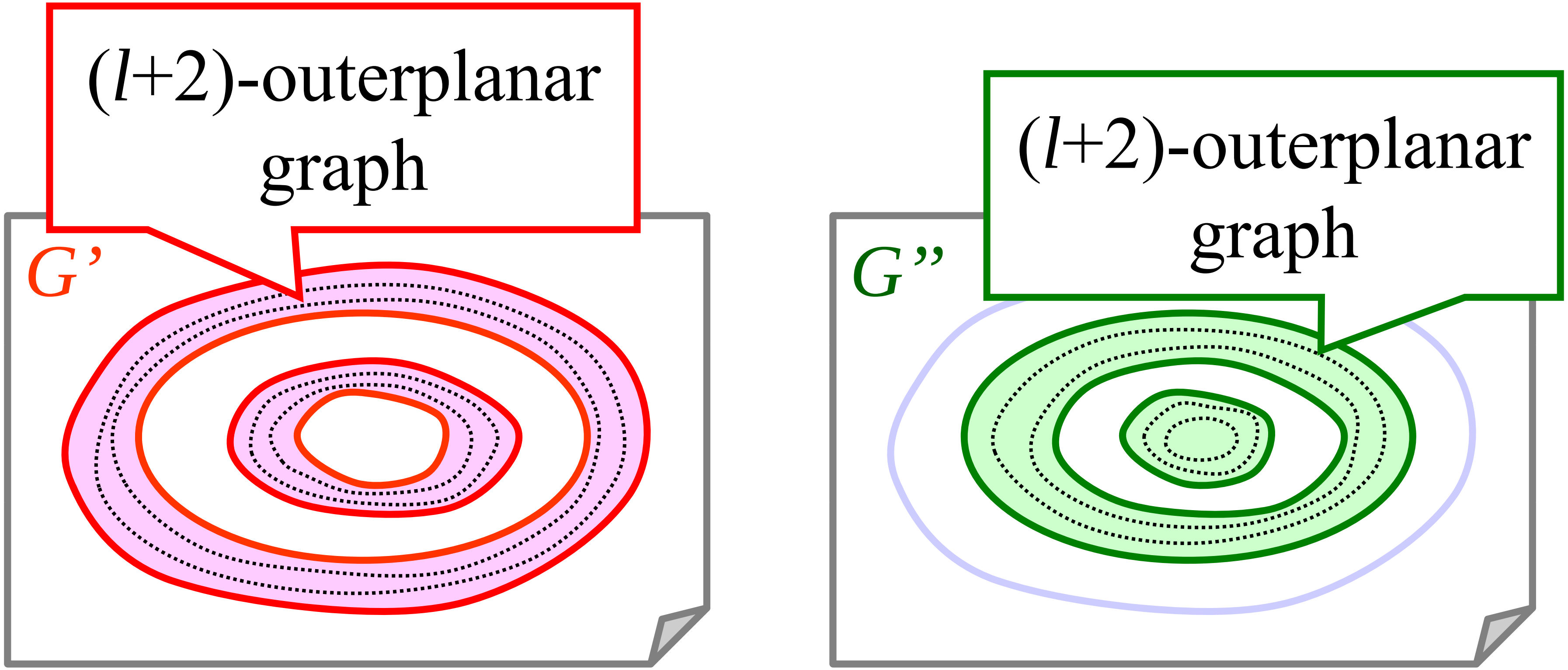


# 2-Approx. Algorithm for Planar Graphs

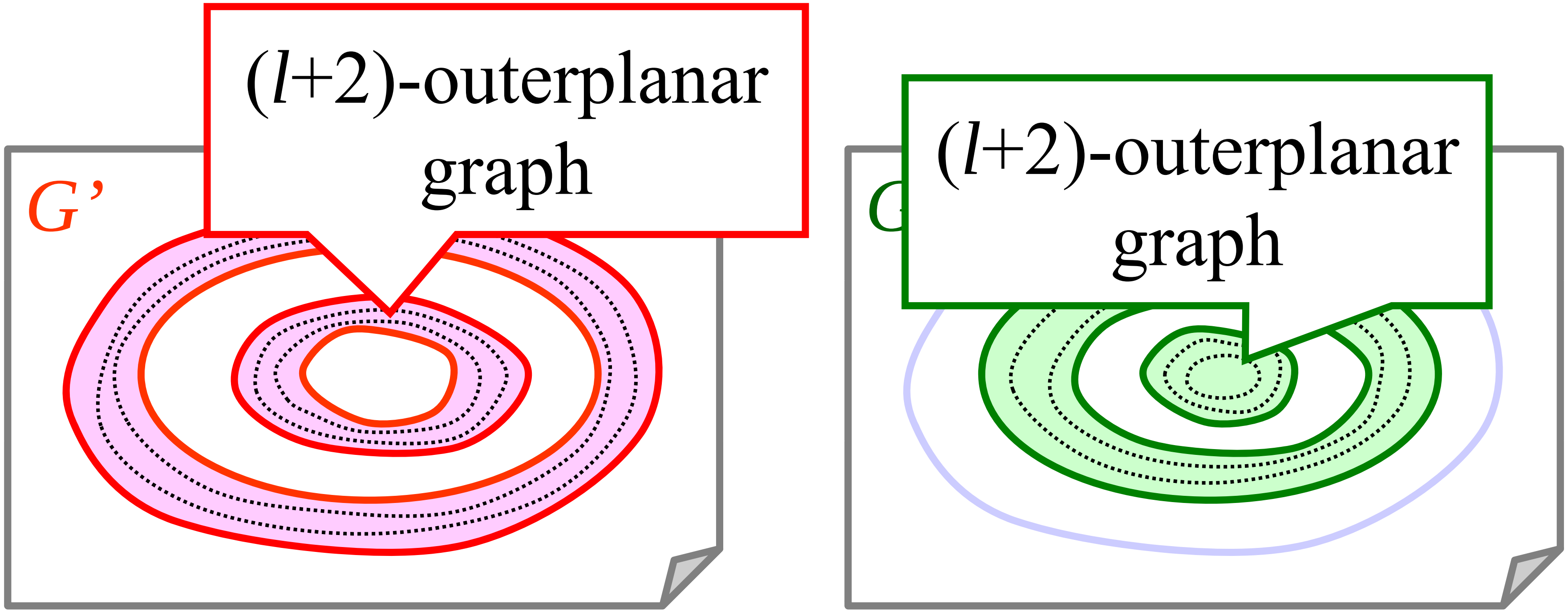



colorings of components can be found separately

# 2-Approx. Algorithm for Planar Graphs

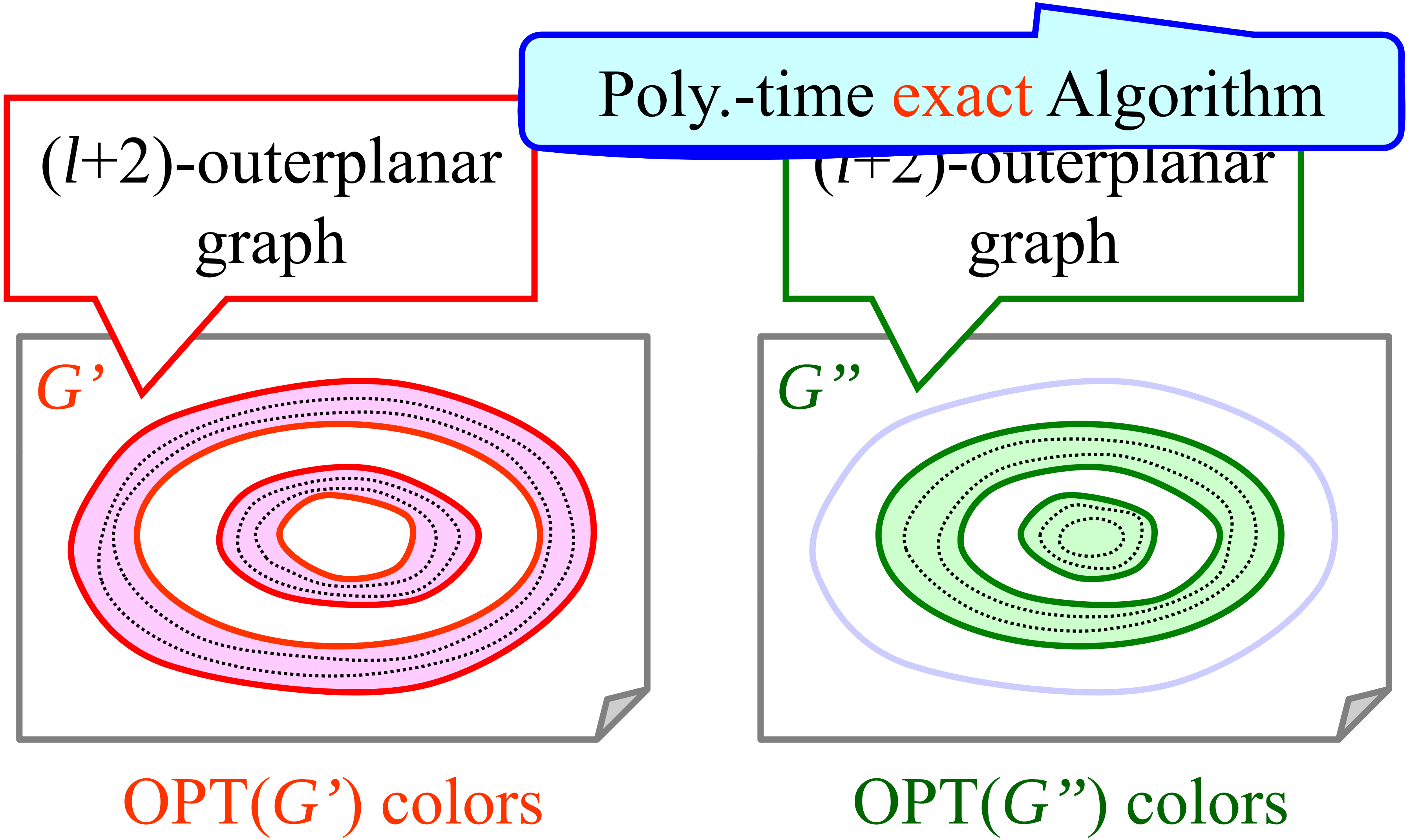


# 2-Approx. Algorithm for Planar Graphs



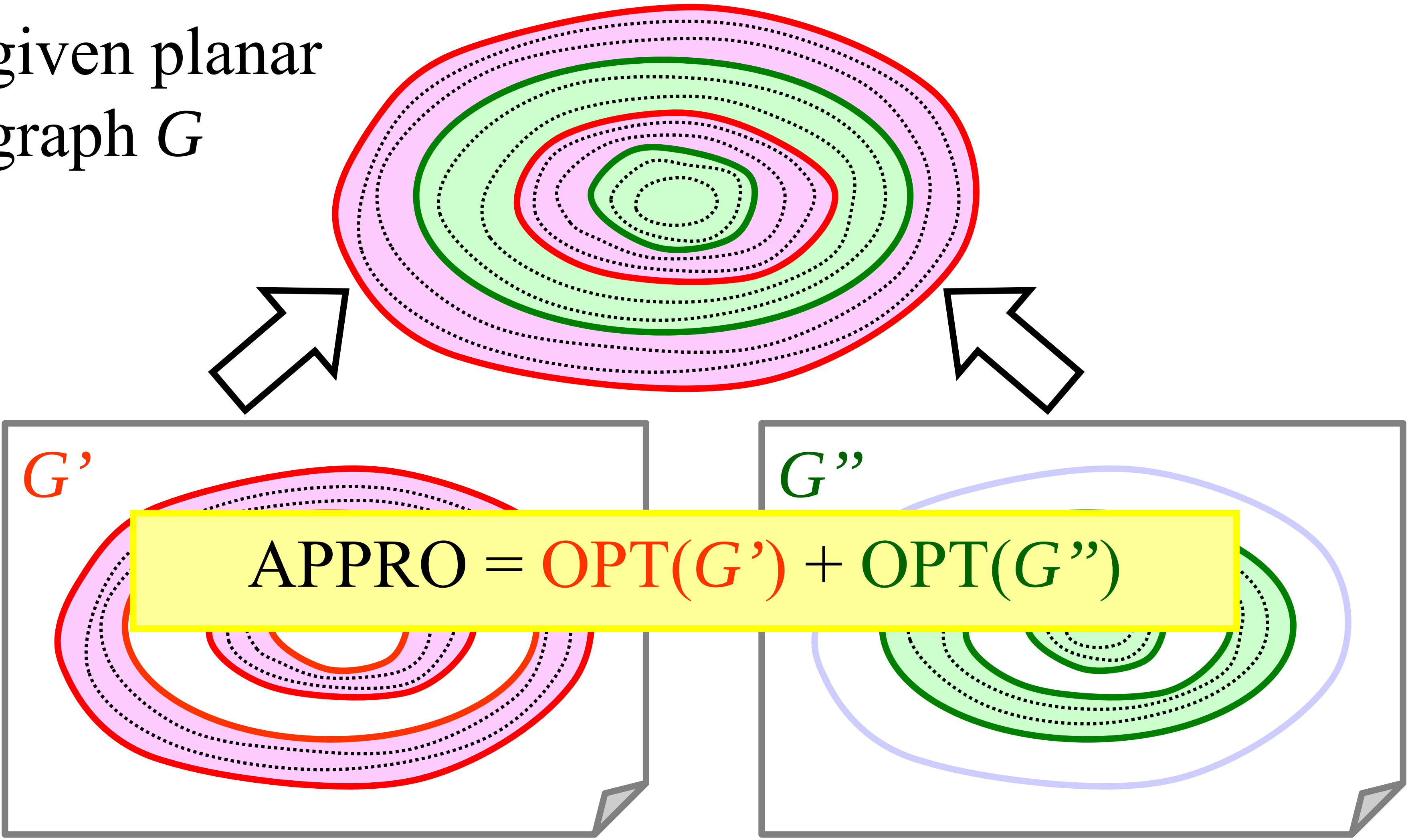
# 2-Approx. Algorithm for Planar Graphs

$(l+2)$-outerplanar graphs ⇨ partial $(3l+5)$-trees



# 2-Approx. Algorithm for Planar Graphs

given planar graph $G$

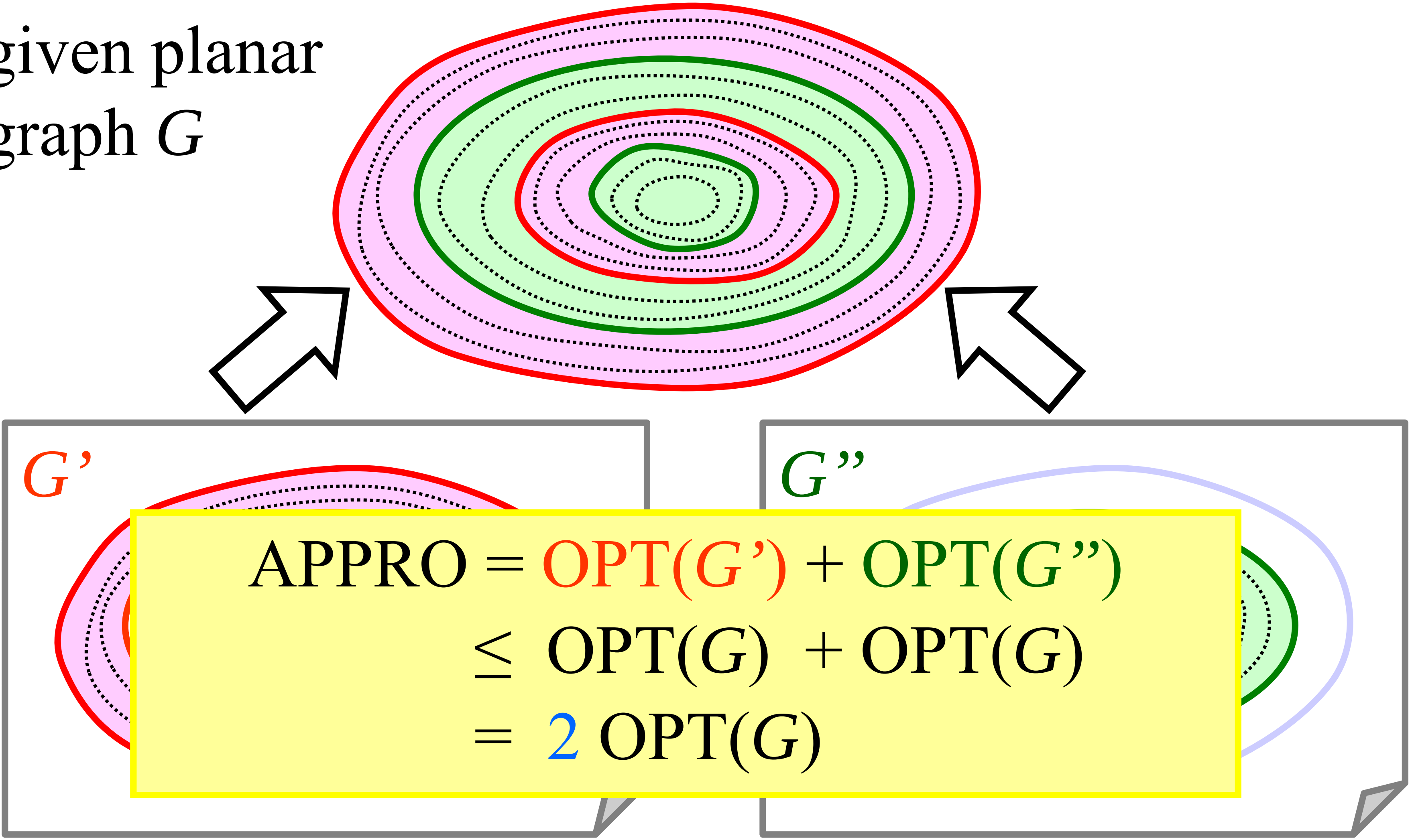



APPRO = OPT($G'$) + OPT($G''$)

OPT($G'$) $\leq$ OPT($G$)    OPT($G''$) $\leq$ OPT($G$)

# 2-Approx. Algorithm for Planar Graphs

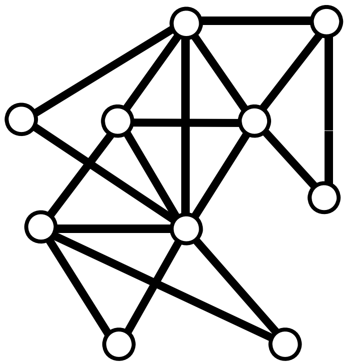
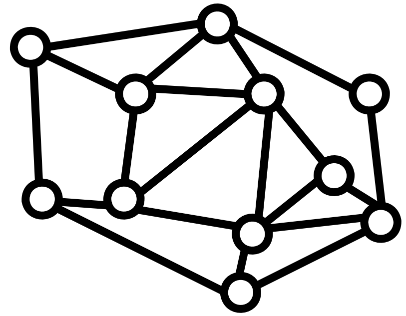
given planar graph $G$




$$\begin{aligned}\text{APPRO} &= \text{OPT}(G') + \text{OPT}(G'') \\ &\le \text{OPT}(G) + \text{OPT}(G) \\ &= 2\,\text{OPT}(G)\end{aligned}$$

$\text{OPT}(G') \le \text{OPT}(G)$ $\quad$ $\text{OPT}(G'') \le \text{OPT}(G)$

# Conclusions

| Partial *k*-Trees | Planar Graphs |
| --- | --- |
|  |  |
| Polynomial-time exact algorithm | Polynomial-time 2-approximation algorithm |

It is easy to modify all algorithms so that it actually finds an *l*-edge-coloring.

END