Group Steiner Problems on Bounded Treewidth Graphs

Syamantak Das
IIIT Delhi
Jointly With

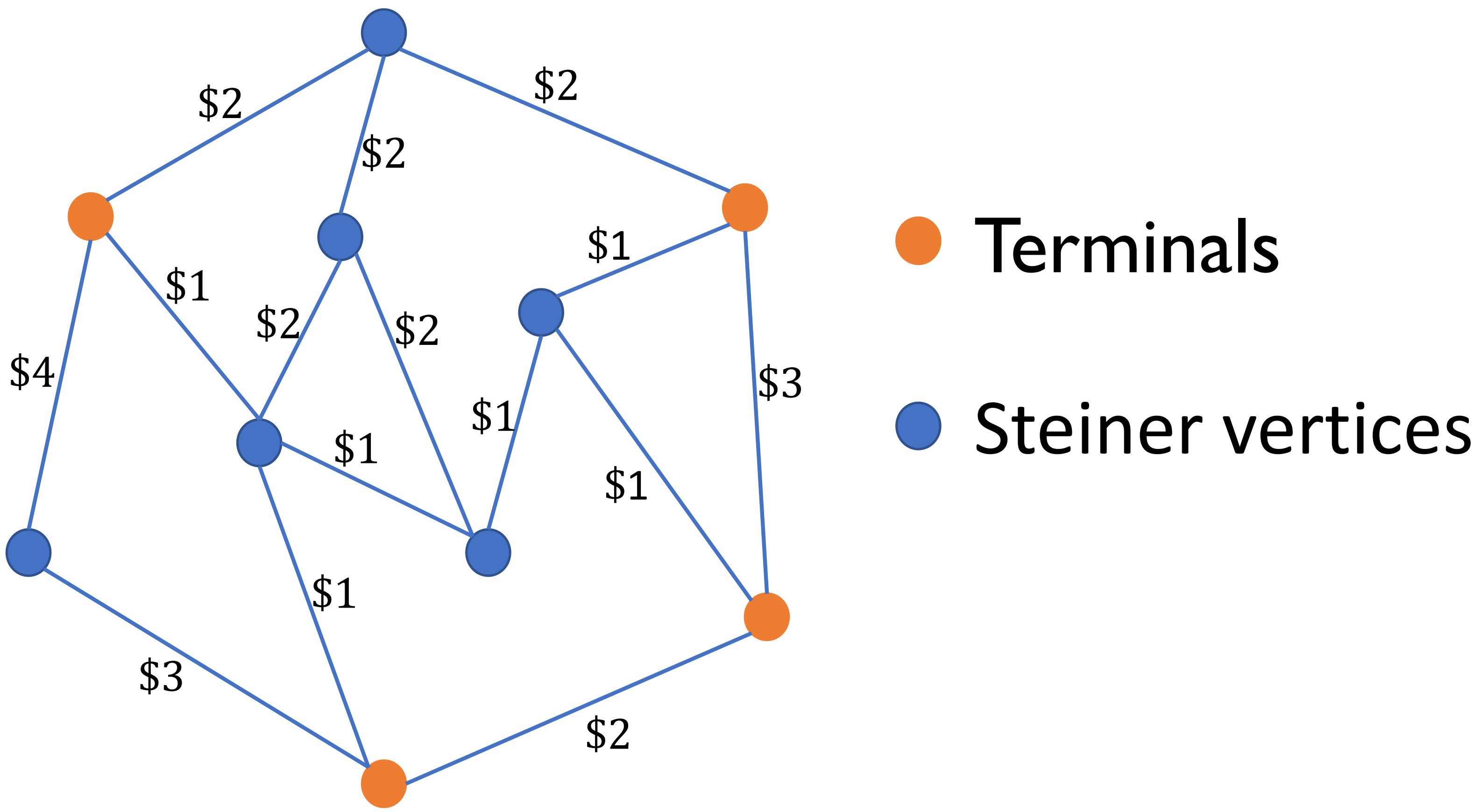
Parinya Chalermsook
Aalto University
Finland

Guy Even
Tel-Aviv University
Israel

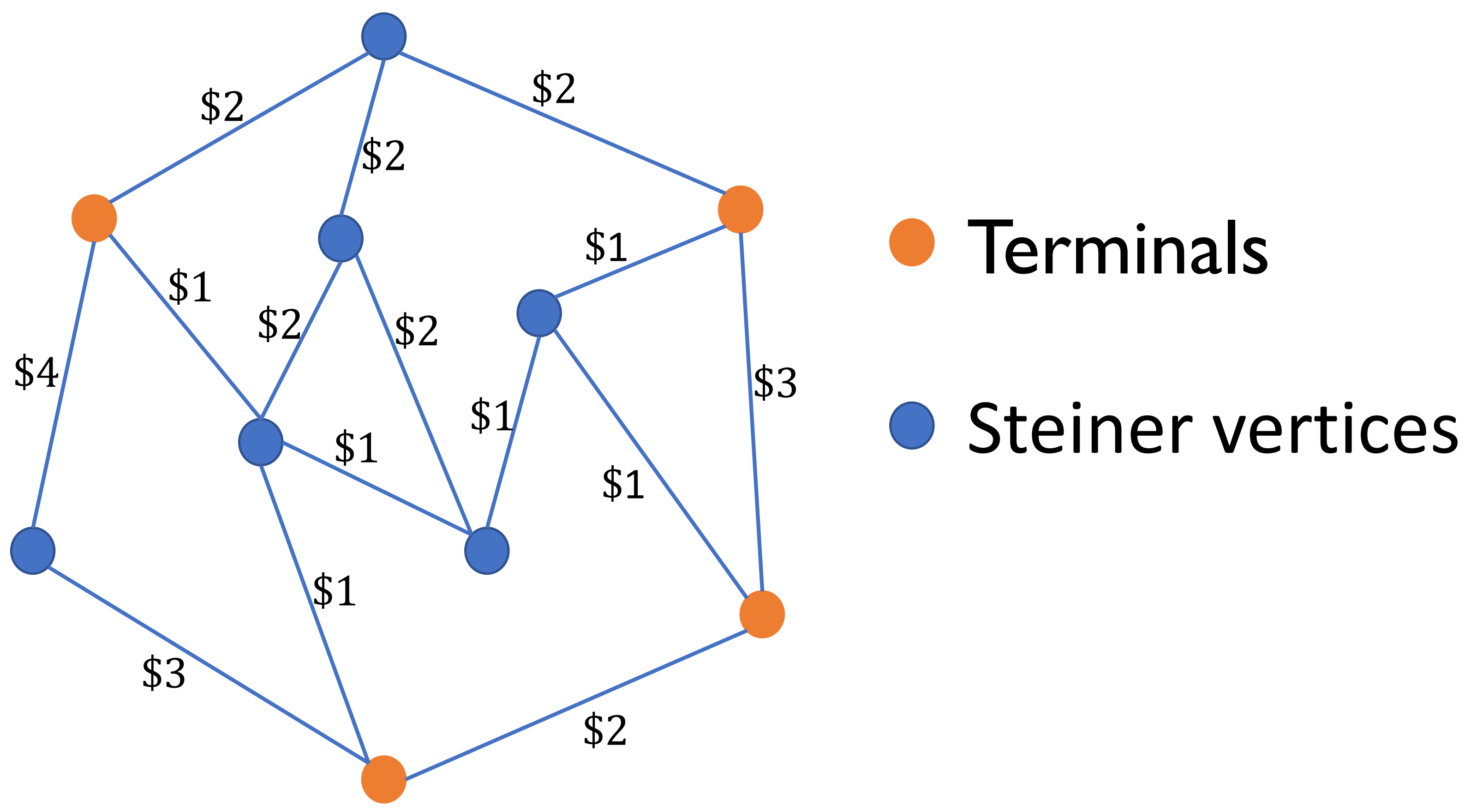
Bundit Laekhanukit
ITCS,
Shanghai,China

Daniel Vaz
MPI-INF/Uni-Saar
Germany

Steiner Tree Problem

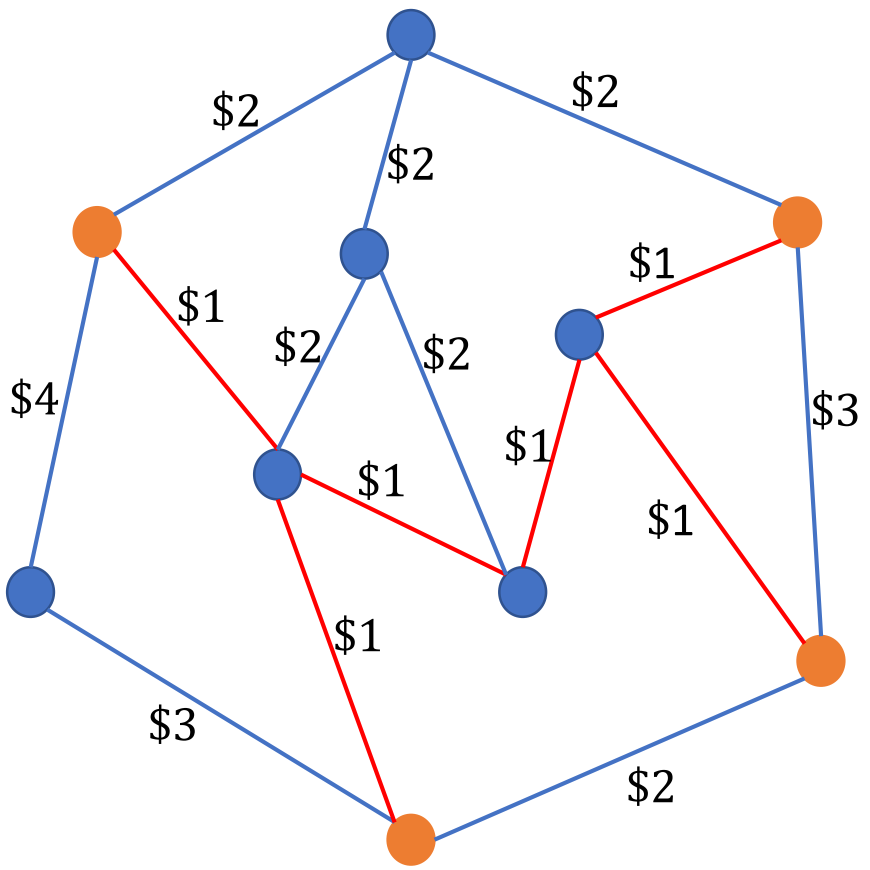


Steiner Tree Problem



Connect the Terminals at the Cheapest Cost

Steiner Tree Problem

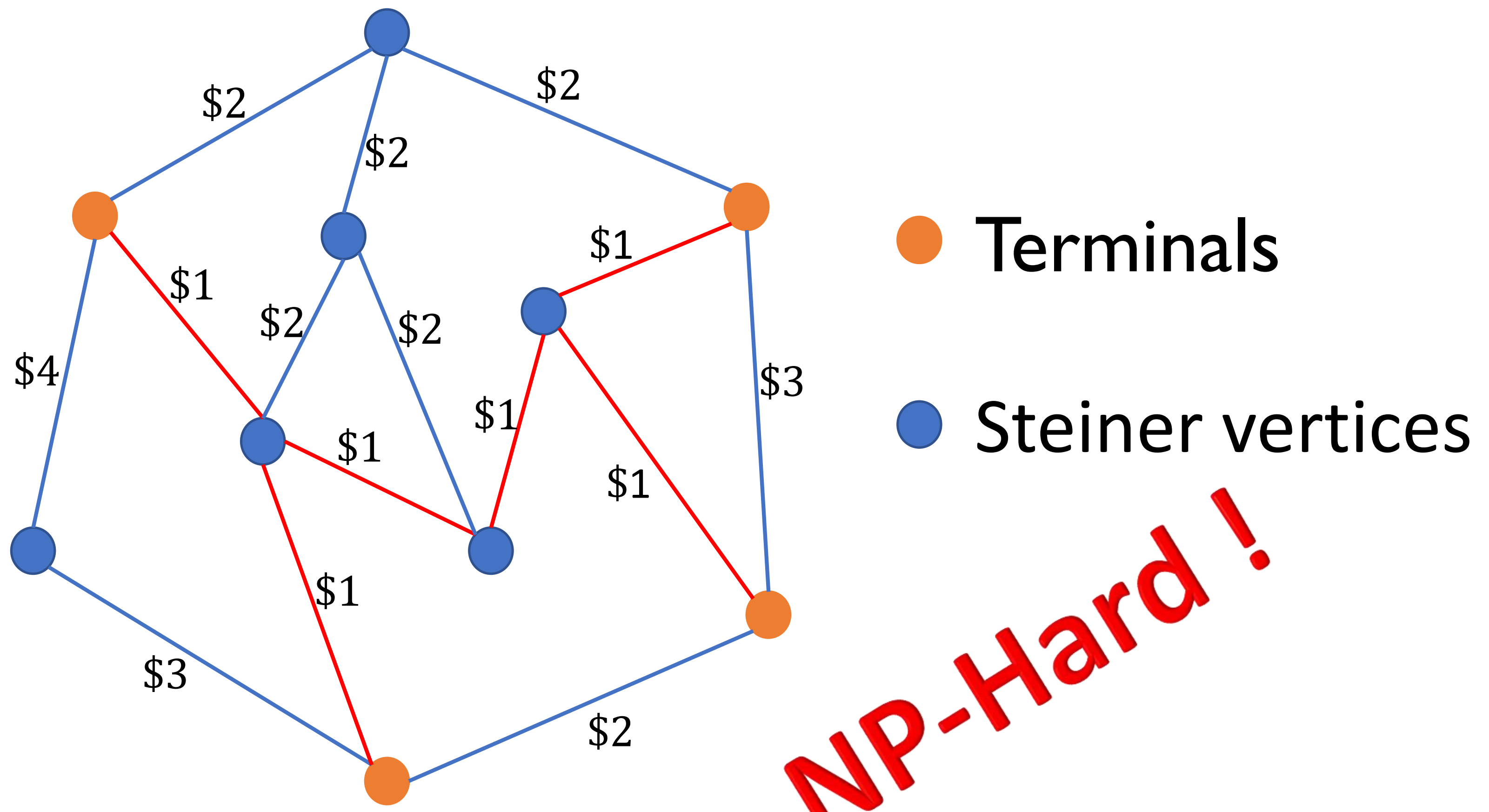


Terminals

Steiner vertices

Connect the Terminals at the Cheapest Cost

Steiner Tree Problem

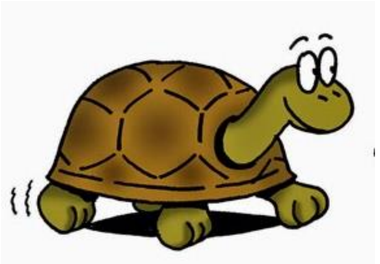


NP-Hard !

Connect the Terminals at the Cheapest Cost

Approximation Algorithm

OPT

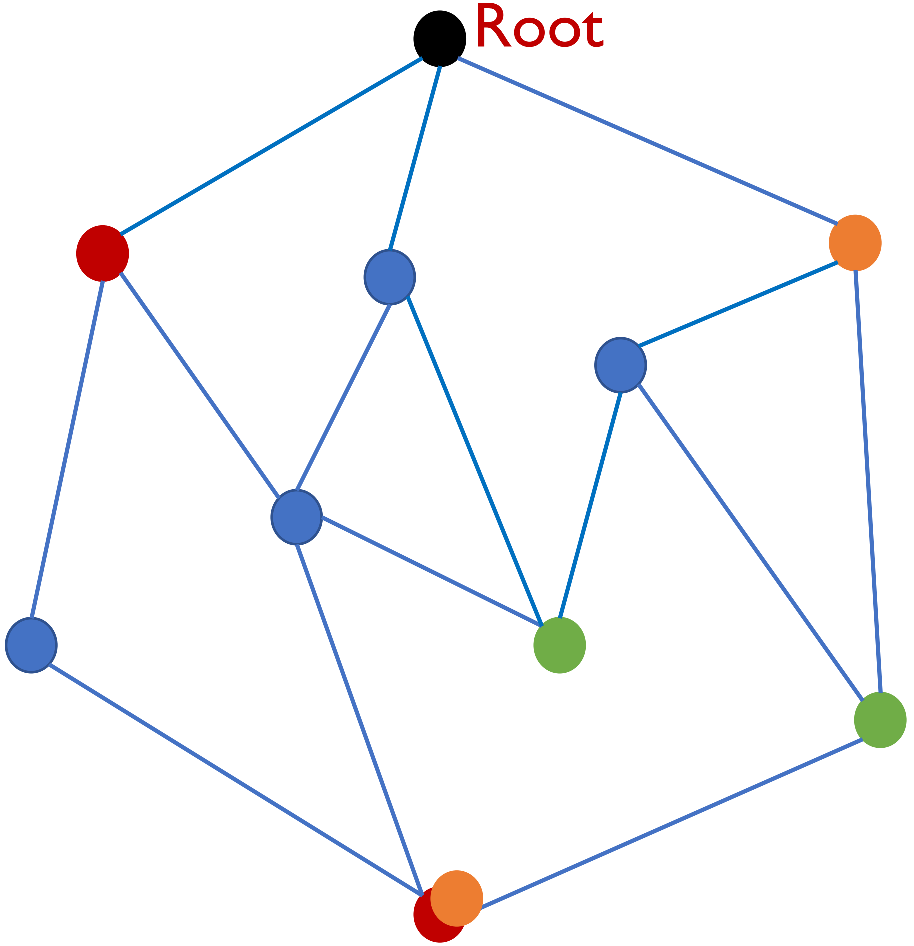


Super-polynomial runtime

$\leq \alpha \cdot OPT$

Polynomial runtime

Group Steiner Tree Problem

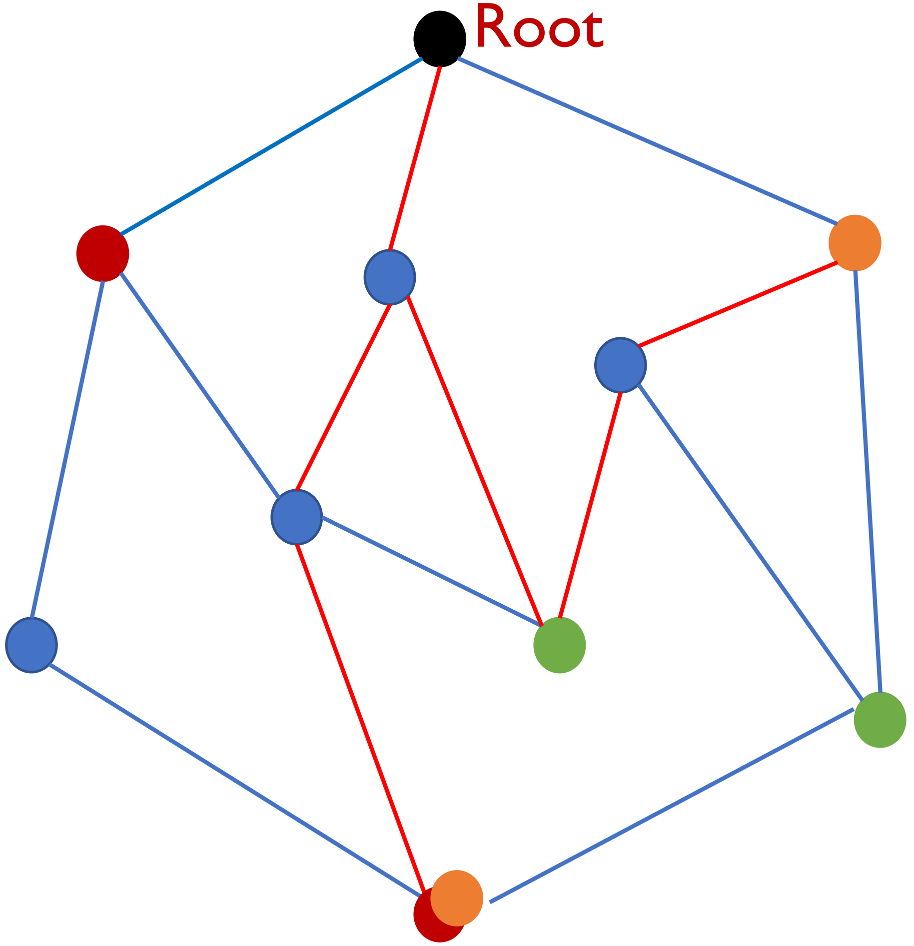


Each color ⇒ Group

● Steiner vertices

Connect at least one Terminal from each Group to the Root at Cheapest Cost

Group Steiner Tree Problem

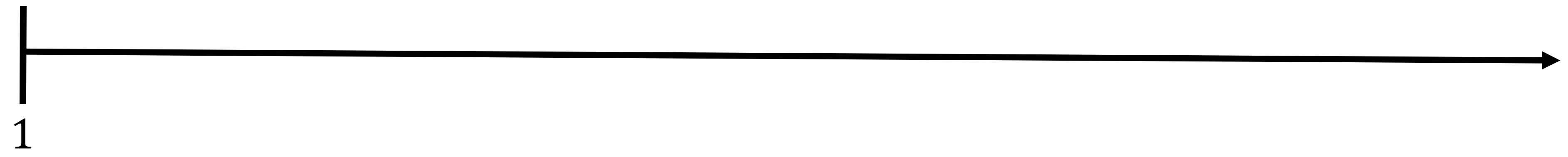


Each color ⇨ Group

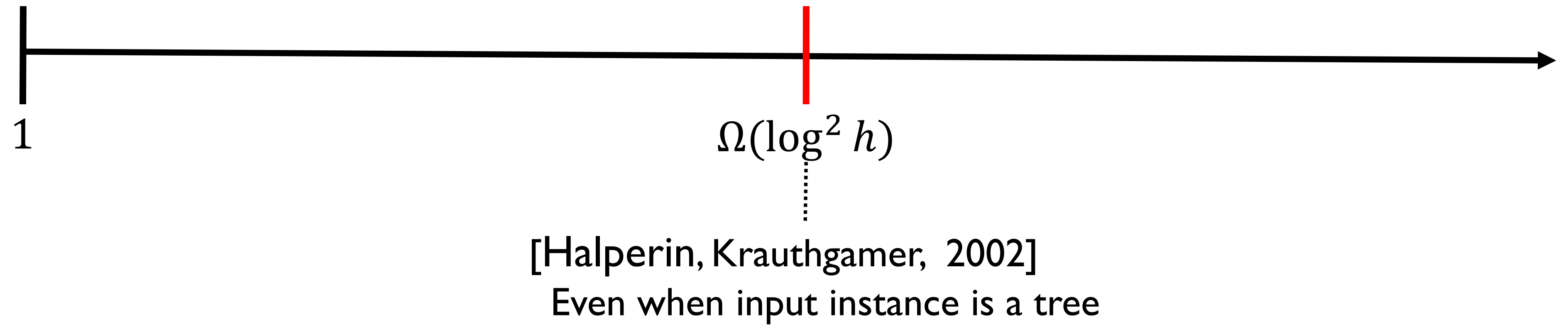
● Steiner vertices

Connect at least one Terminal from each Group to the Root at Cheapest Cost

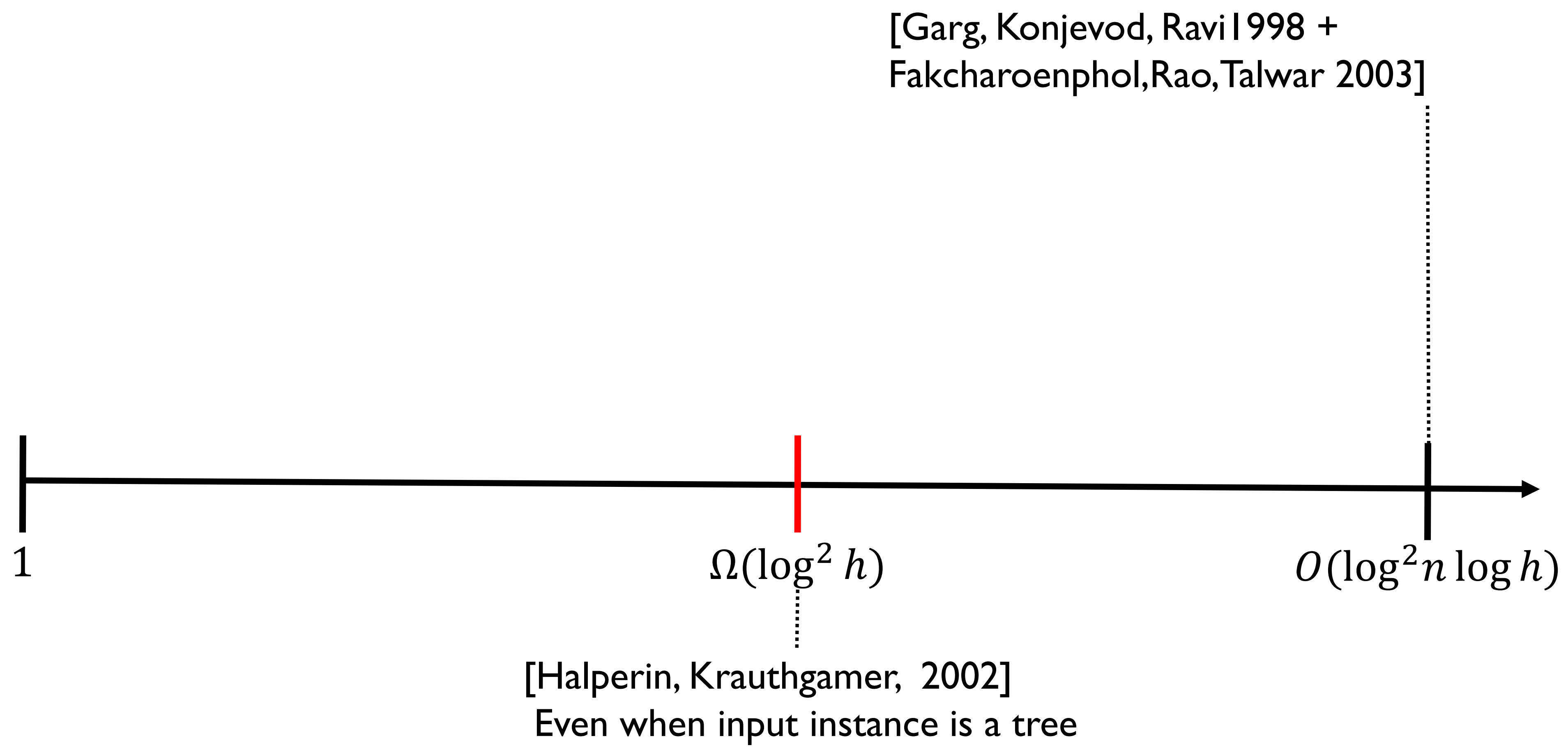
Approximability of Group Steiner Tree Problem



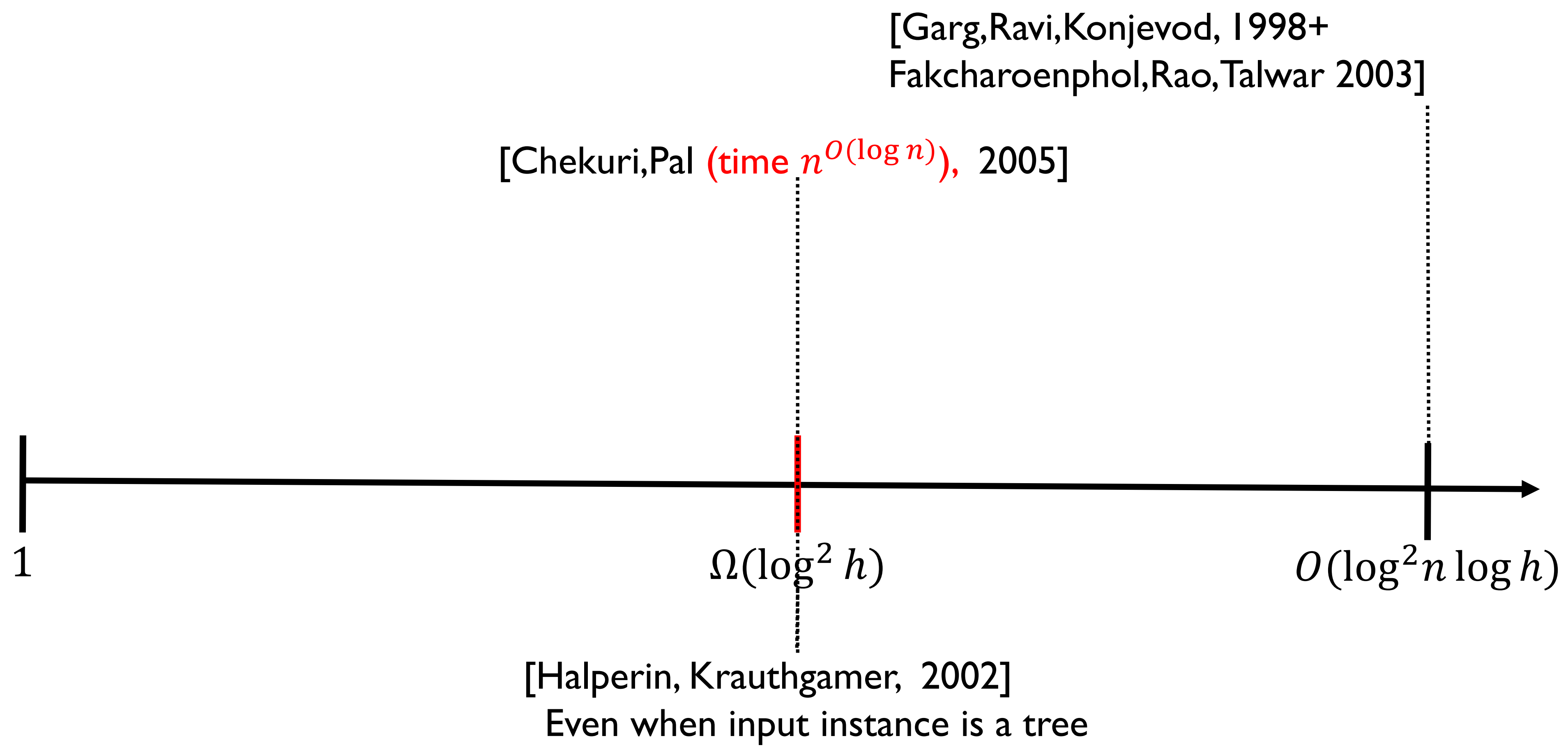
Approximability of Group Steiner Tree Problem



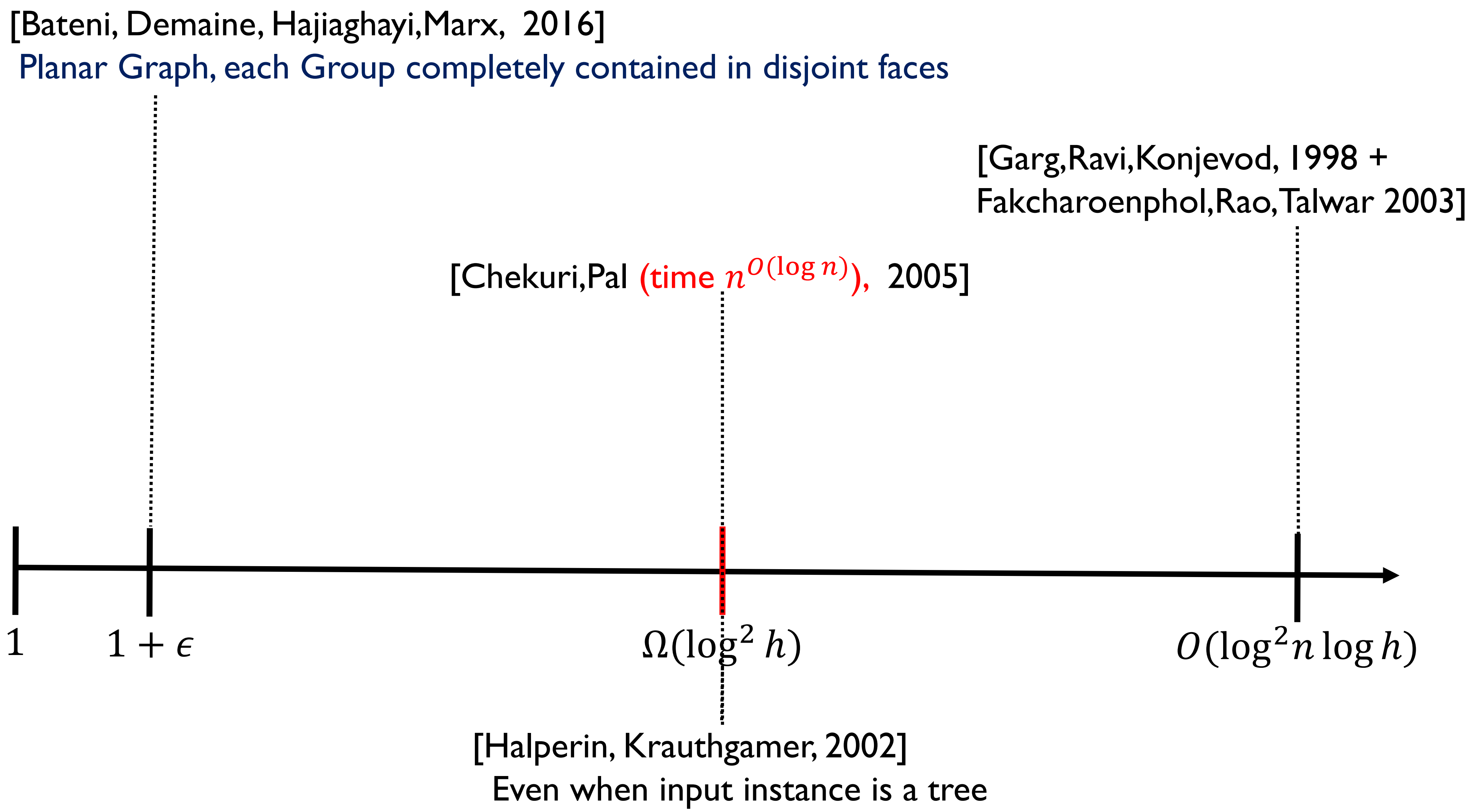
Approximability of Group Steiner Tree Problem



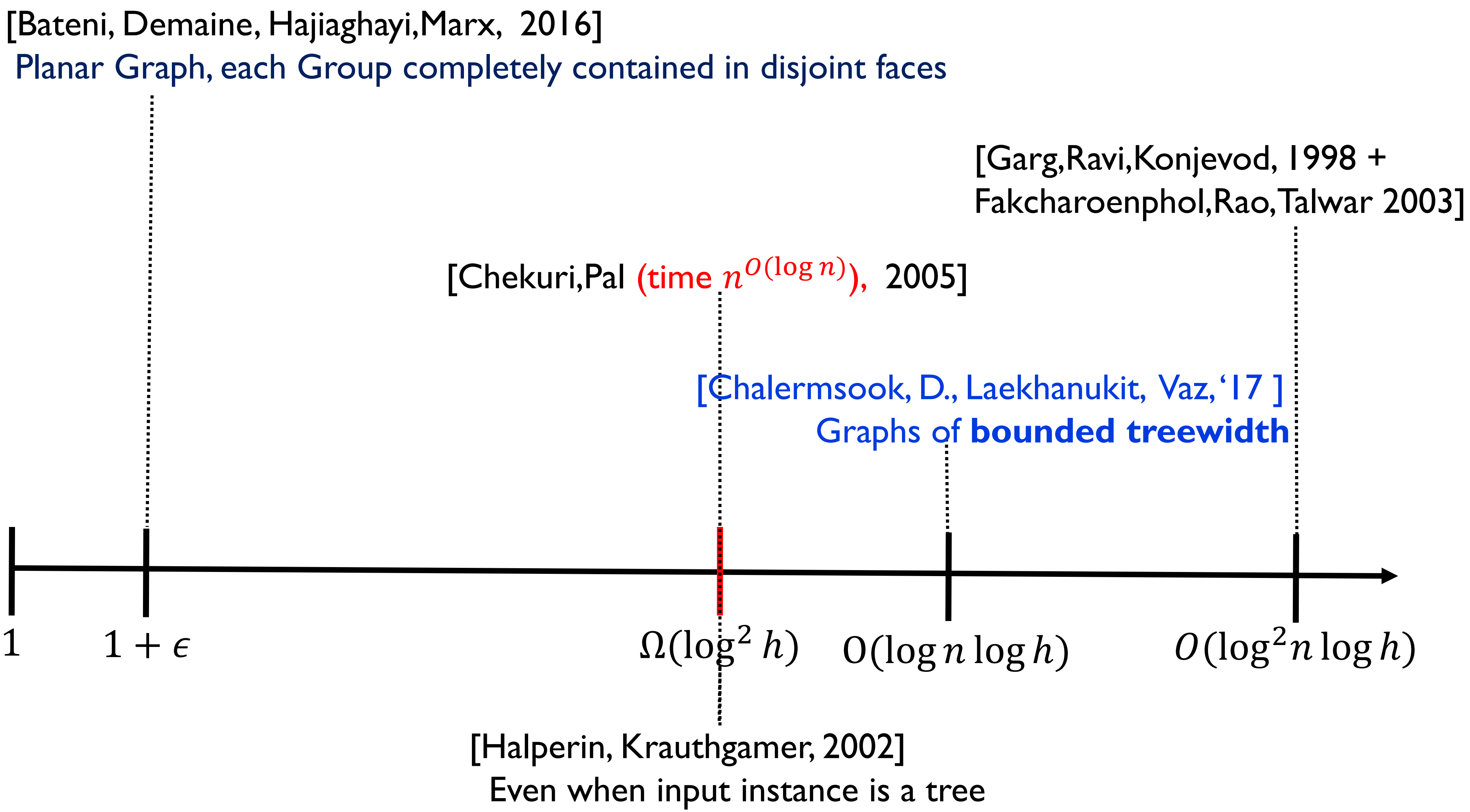
Approximability of Group Steiner Tree Problem



Approximability of Group Steiner Tree Problem



Approximability of Group Steiner Tree Problem

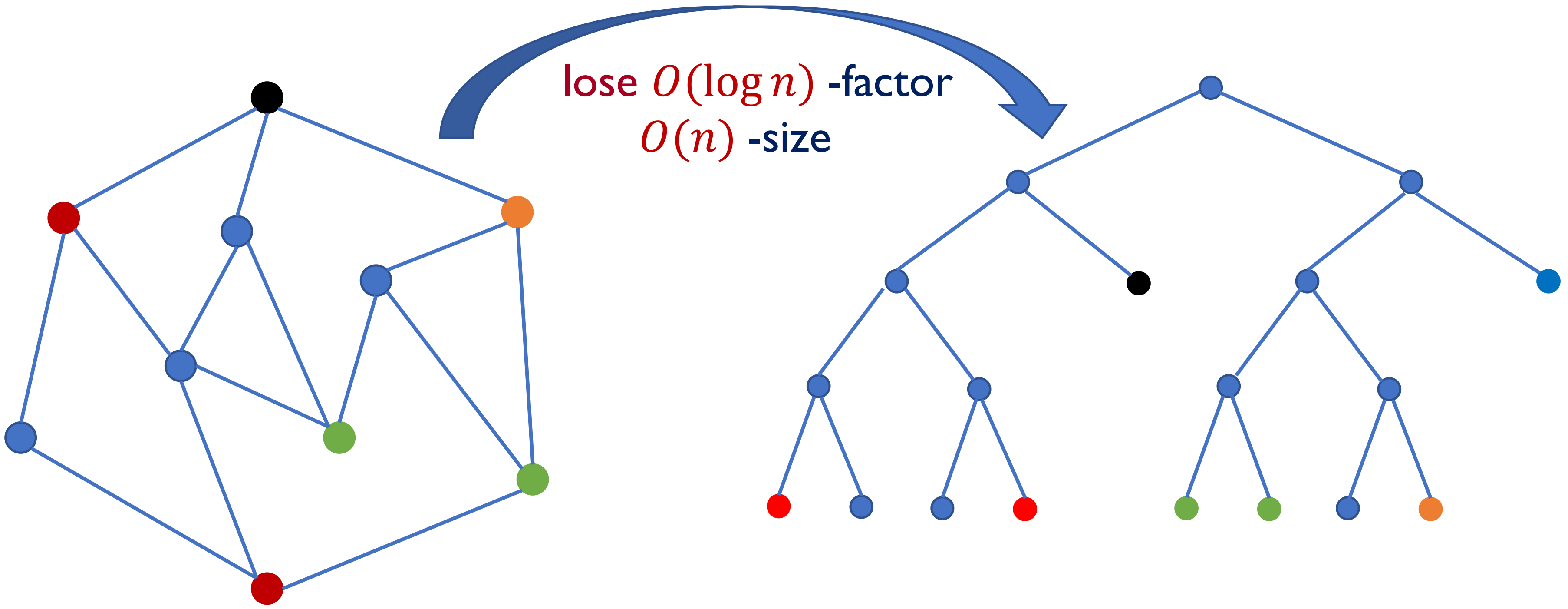


Key Technique : Probabilistic Embedding in to a Tree Metric

[Bartal `99, Fakcharoenphol, Rao, Talwar, `03]

lose $O(\log n)$ -factor

$O(n)$ -size

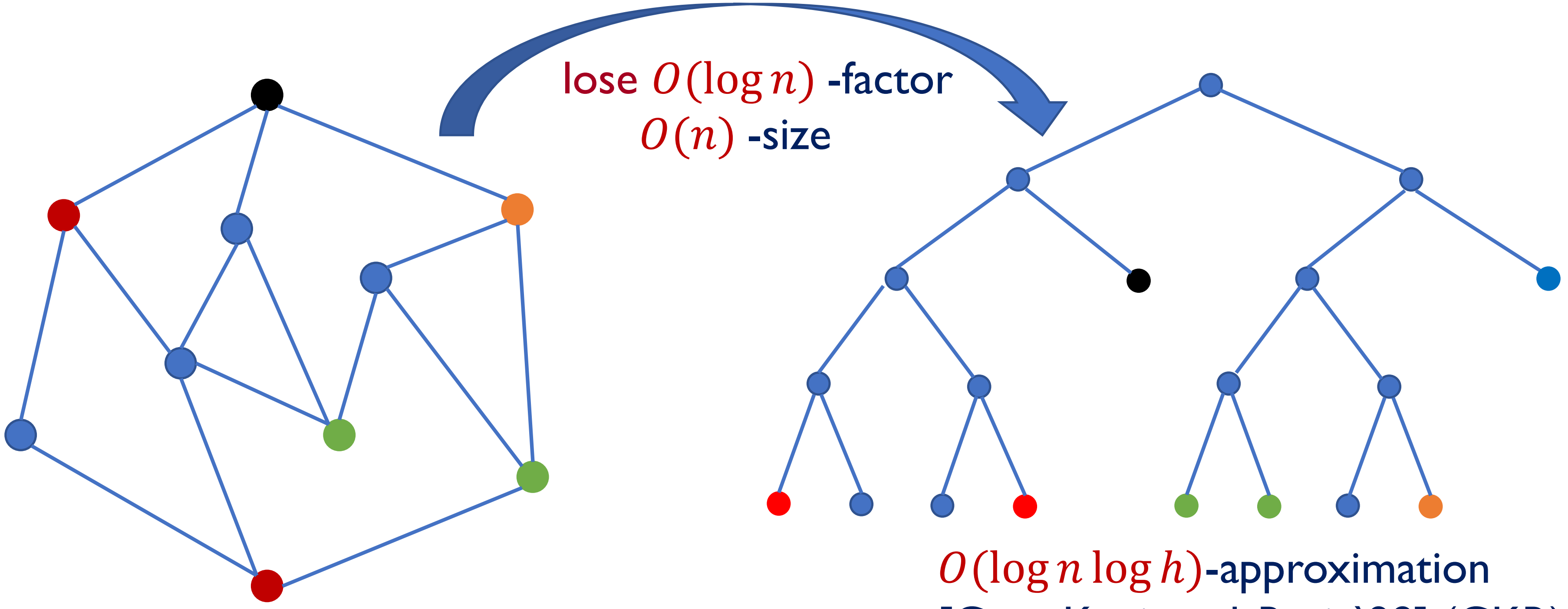


Key Technique : Probabilistic Embedding in to a Tree Metric

[Bartal `99, Fakcharoenphol, Rao, Talwar, `03]

lose $O(\log n)$ -factor

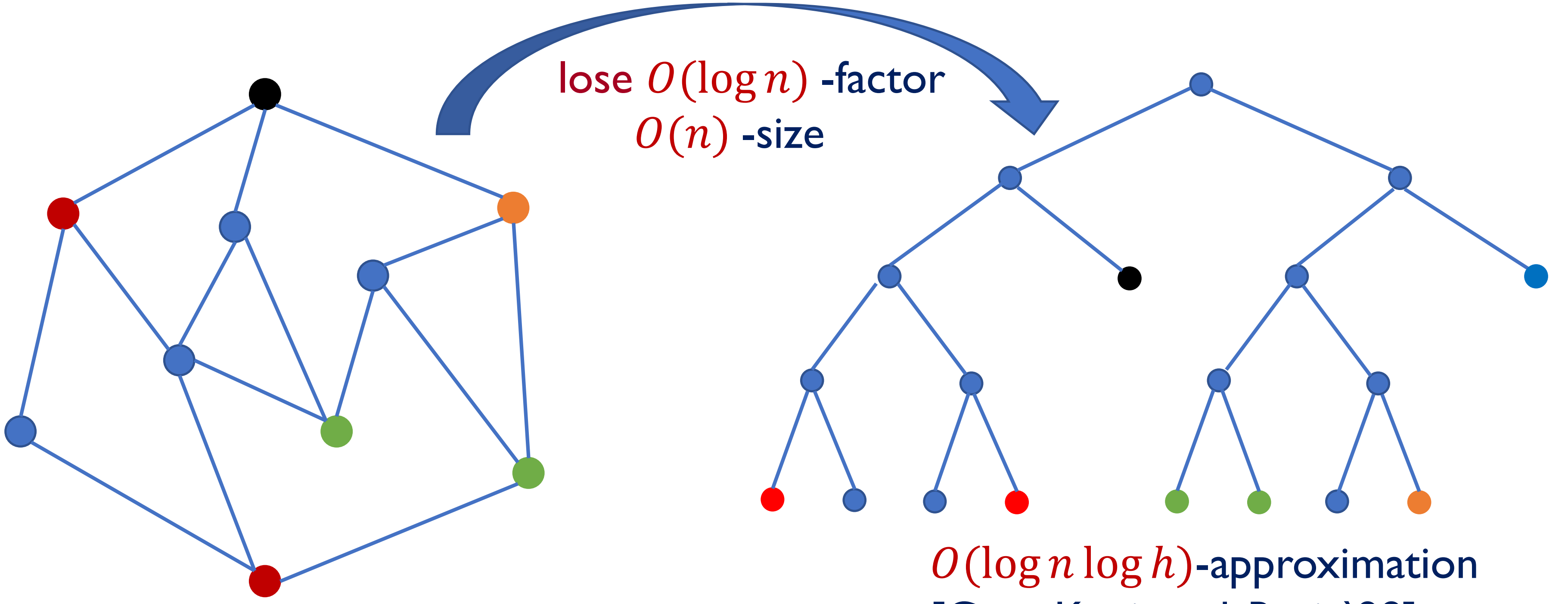
$O(n)$ -size



$O(\log n \log h)$-approximation
[Garg, Konjevod, Ravi, `98] (GKR)
(Via Rounding a suitable Linear Program)

Key Technique : Probabilistic Embedding in to a Tree Metric

[Bartal `99, Fakcharoenphol, Rao, Talwar, `03]

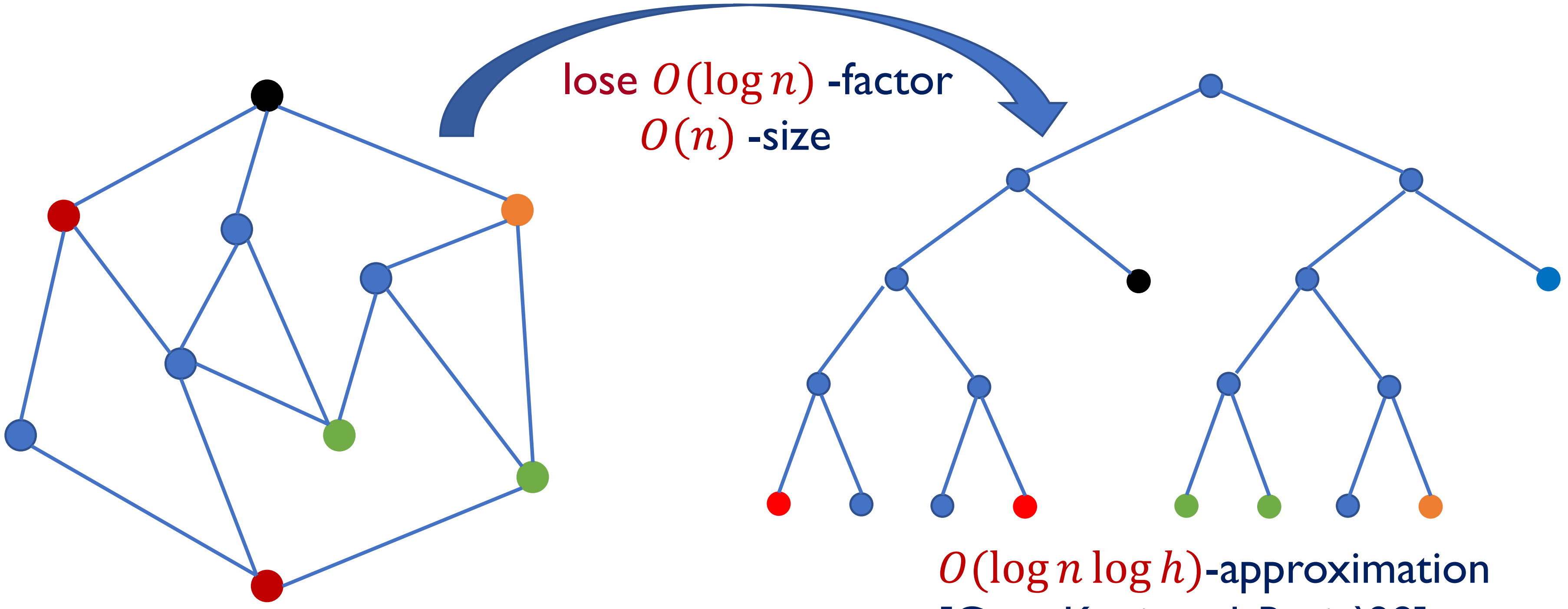


$O(\log n \log h)$-approximation
[Garg, Konjevod, Ravi, `98]

Bottleneck 1: $\Omega(\log^2 h)$ -hard on Trees

Key Technique : Probabilistic Embedding in to a Tree Metric

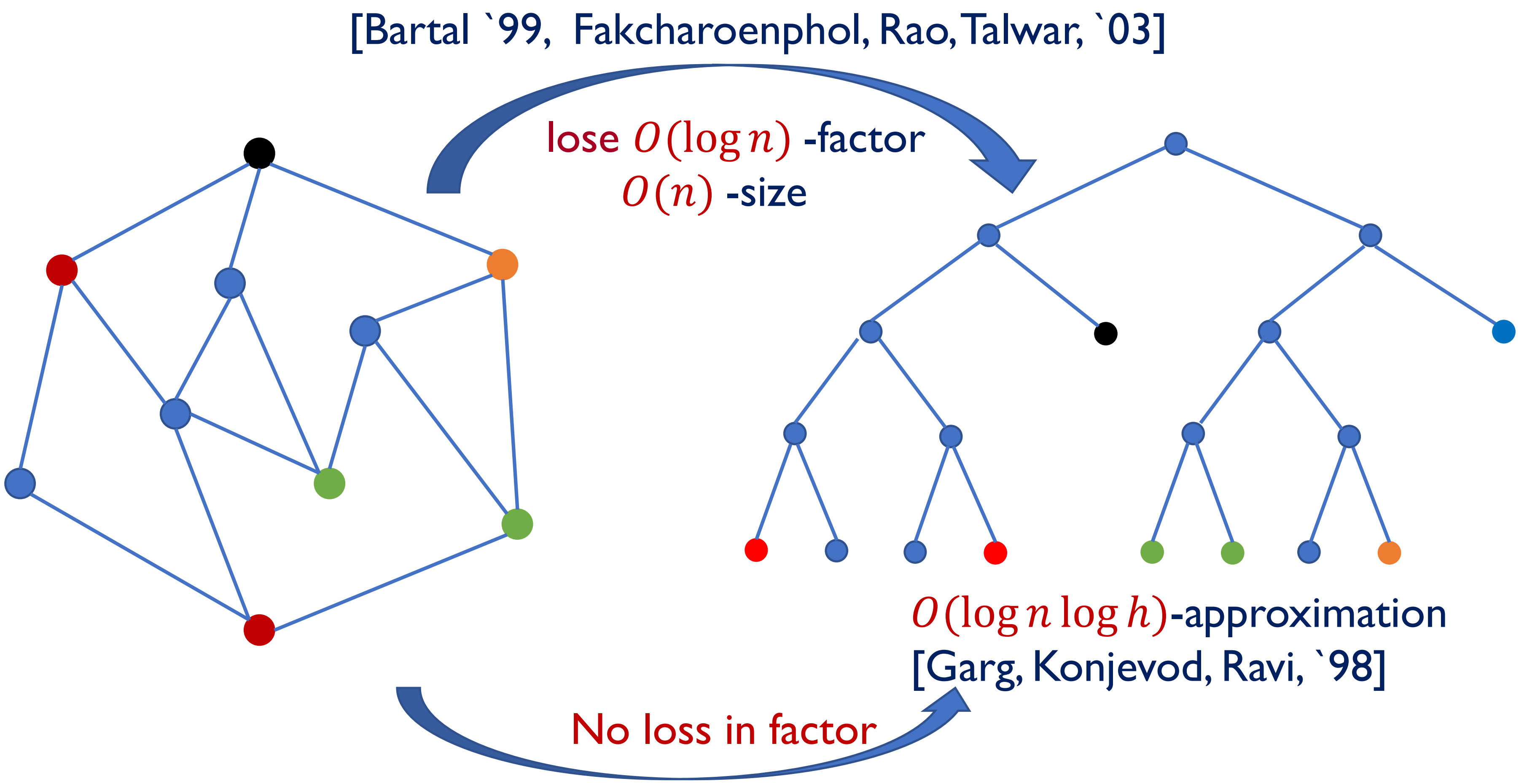
[Bartal `99, Fakcharoenphol, Rao, Talwar, `03]



$O(\log n \log h)$-approximation
[Garg, Konjevod, Ravi, `98]

Bottleneck 1: $\Omega(\log^2 h)$ -hard on Trees
Bottleneck 2: $\Omega(\log n)$ for Bartal embedding (even for treewidth 2)

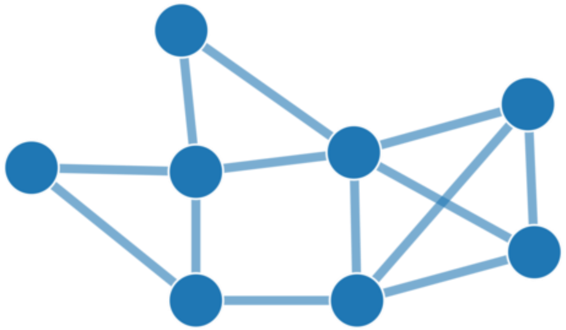
Key Contribution : An Alternate notion of "Embedding"

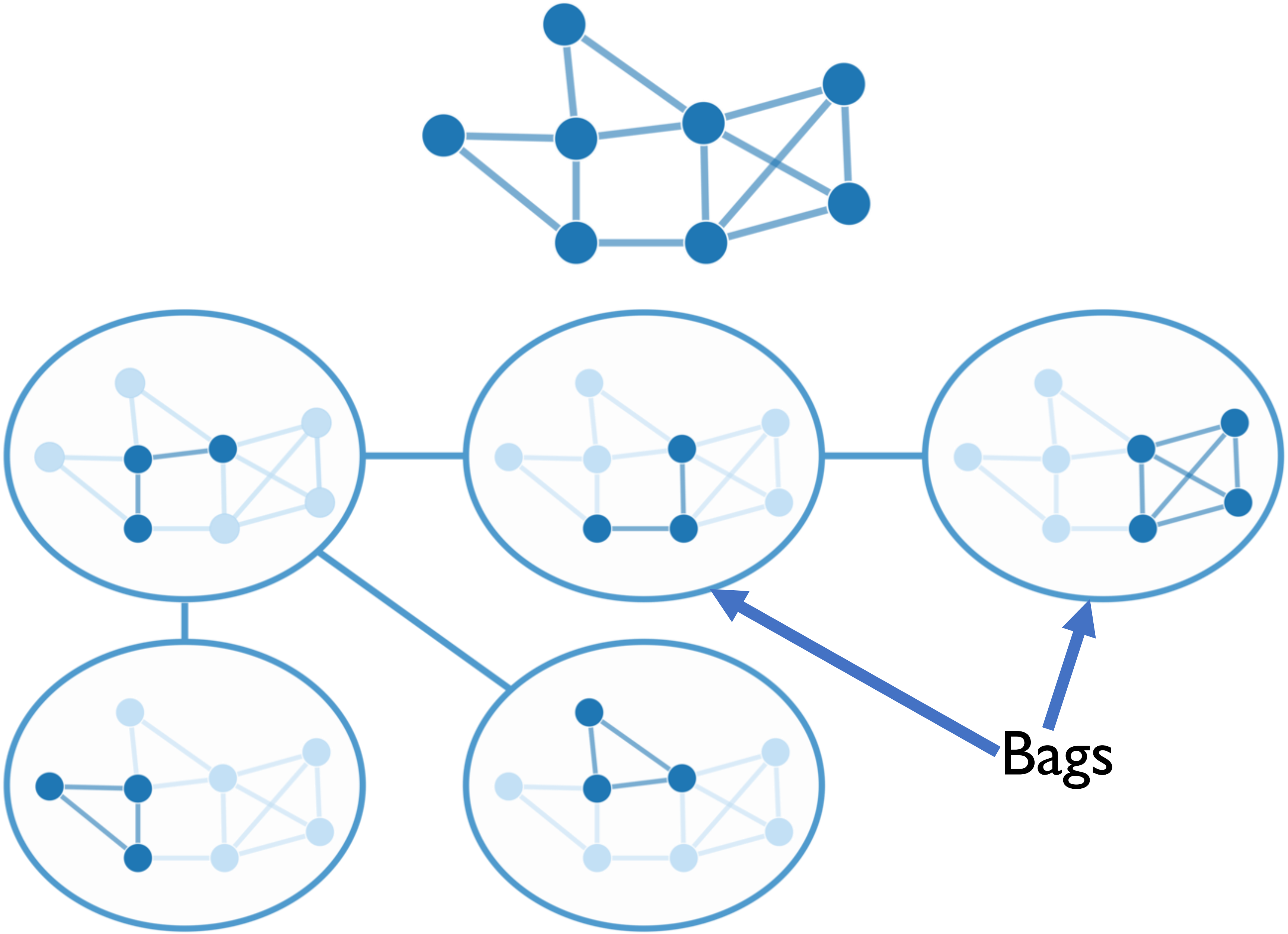


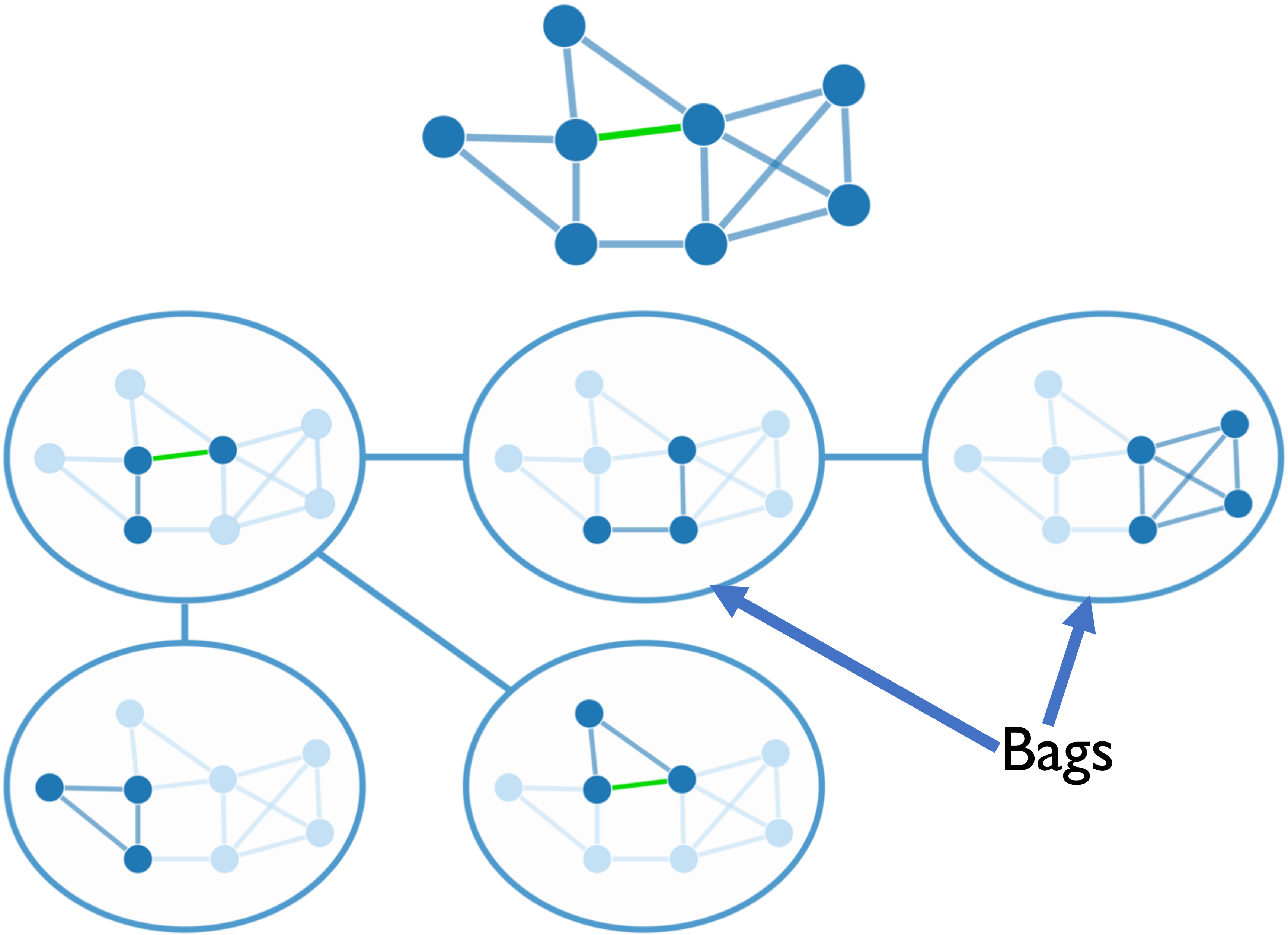
Size = $O(n^{w \log w})$, w = treewidth of the graph

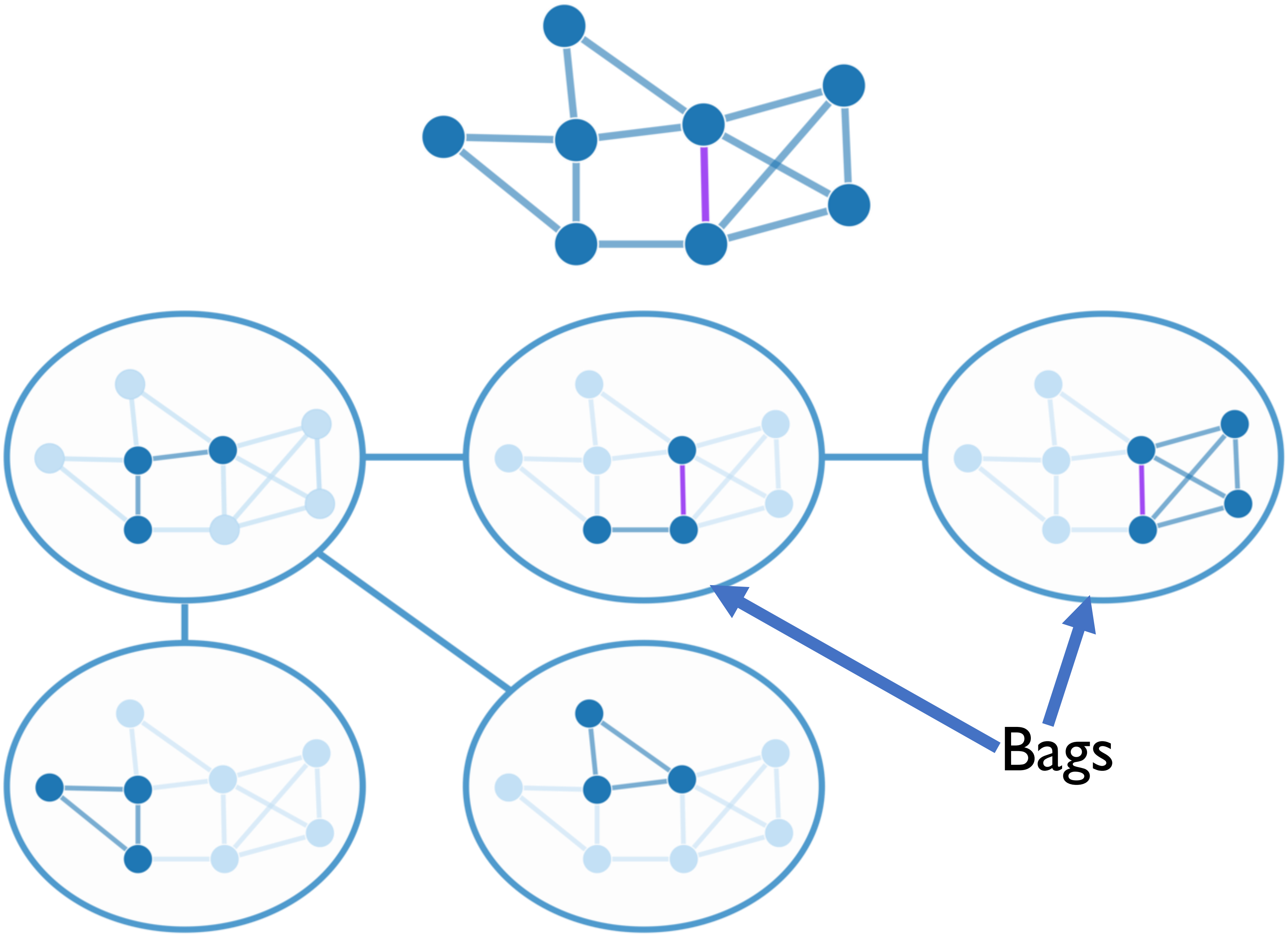
[Chalermsook, D., Laekhanukit, Vaz, 2017]

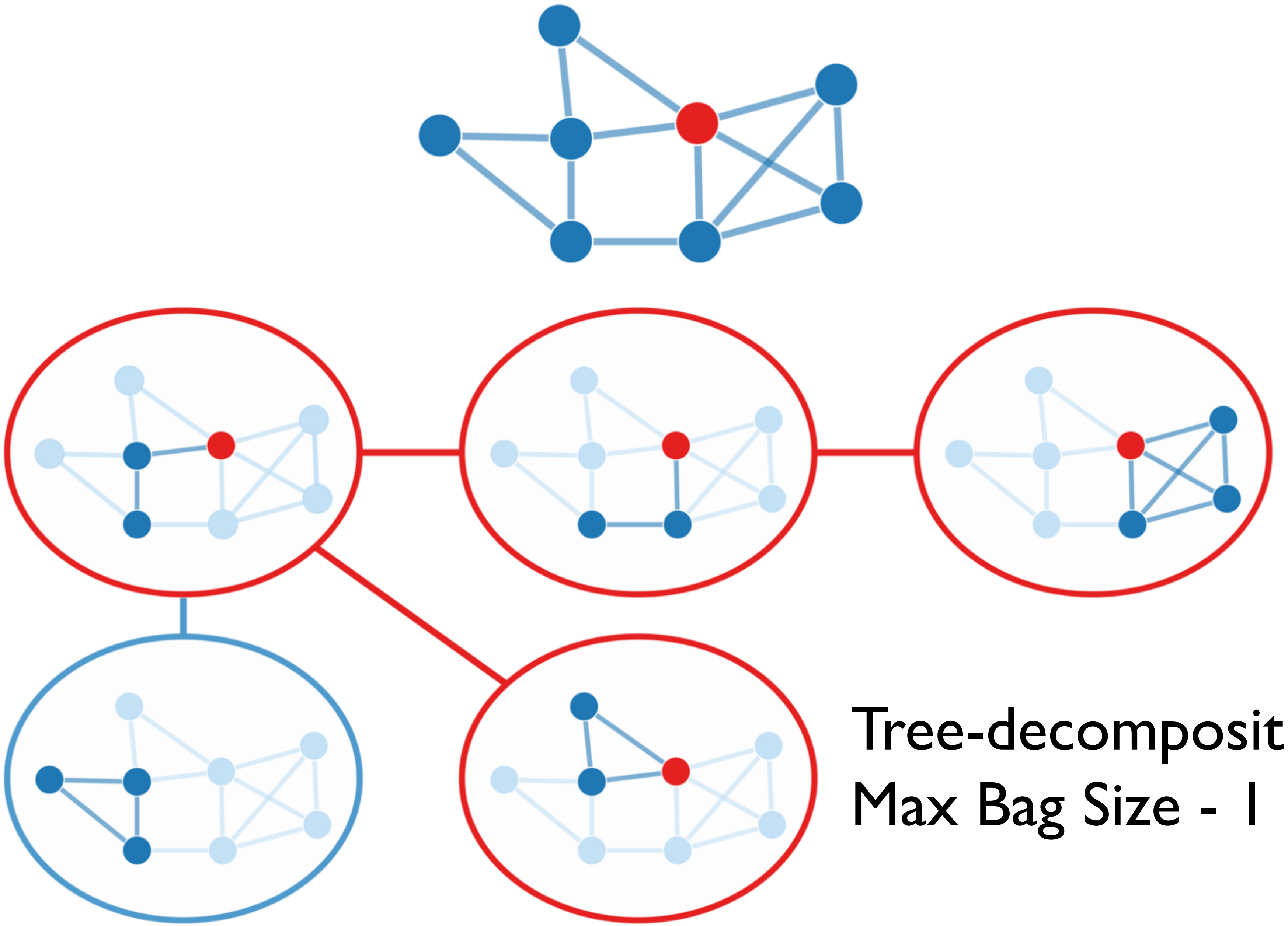
Tree decomposition and Treewidth : A Primer



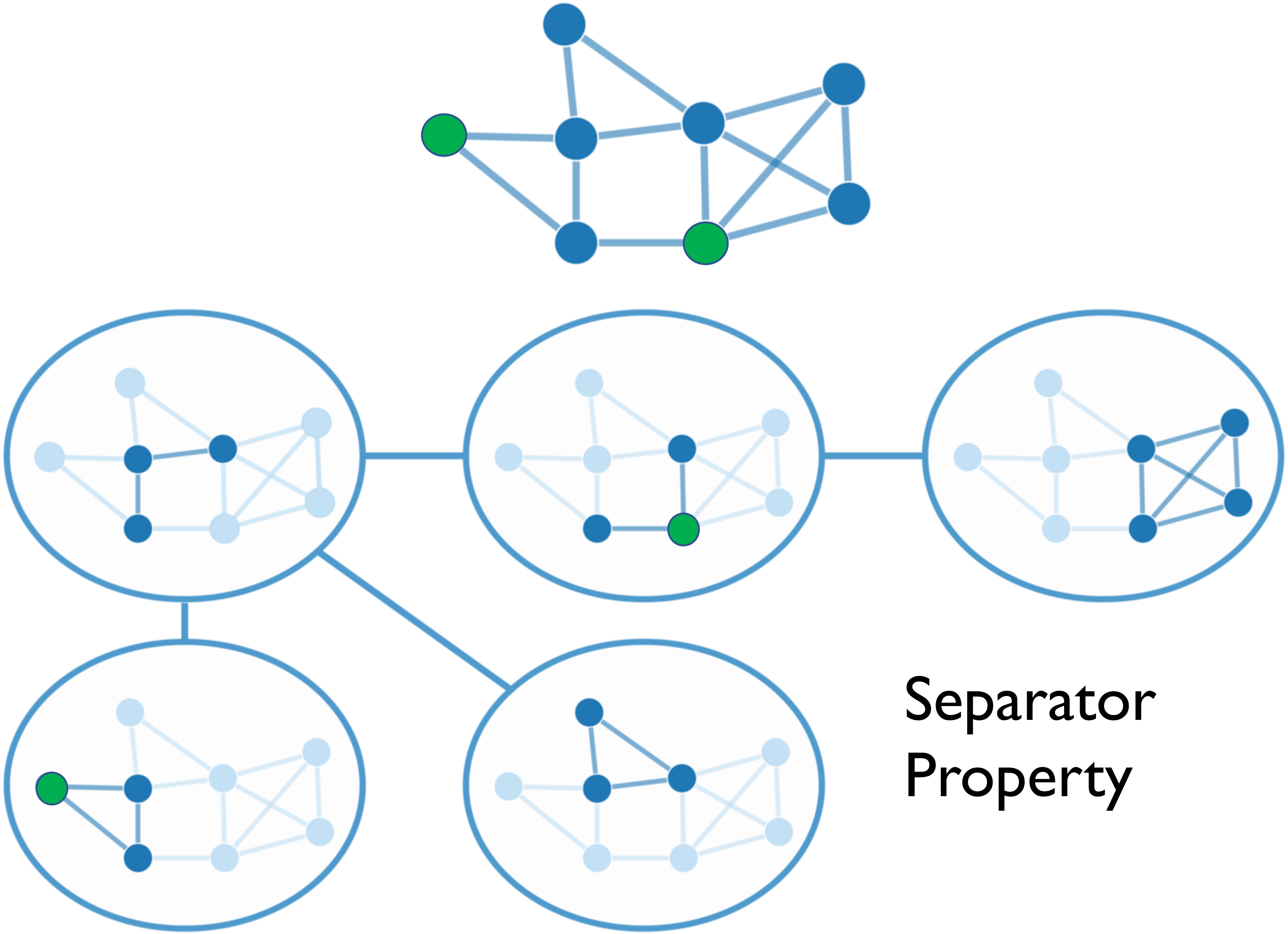

Bags


Bags


Bags

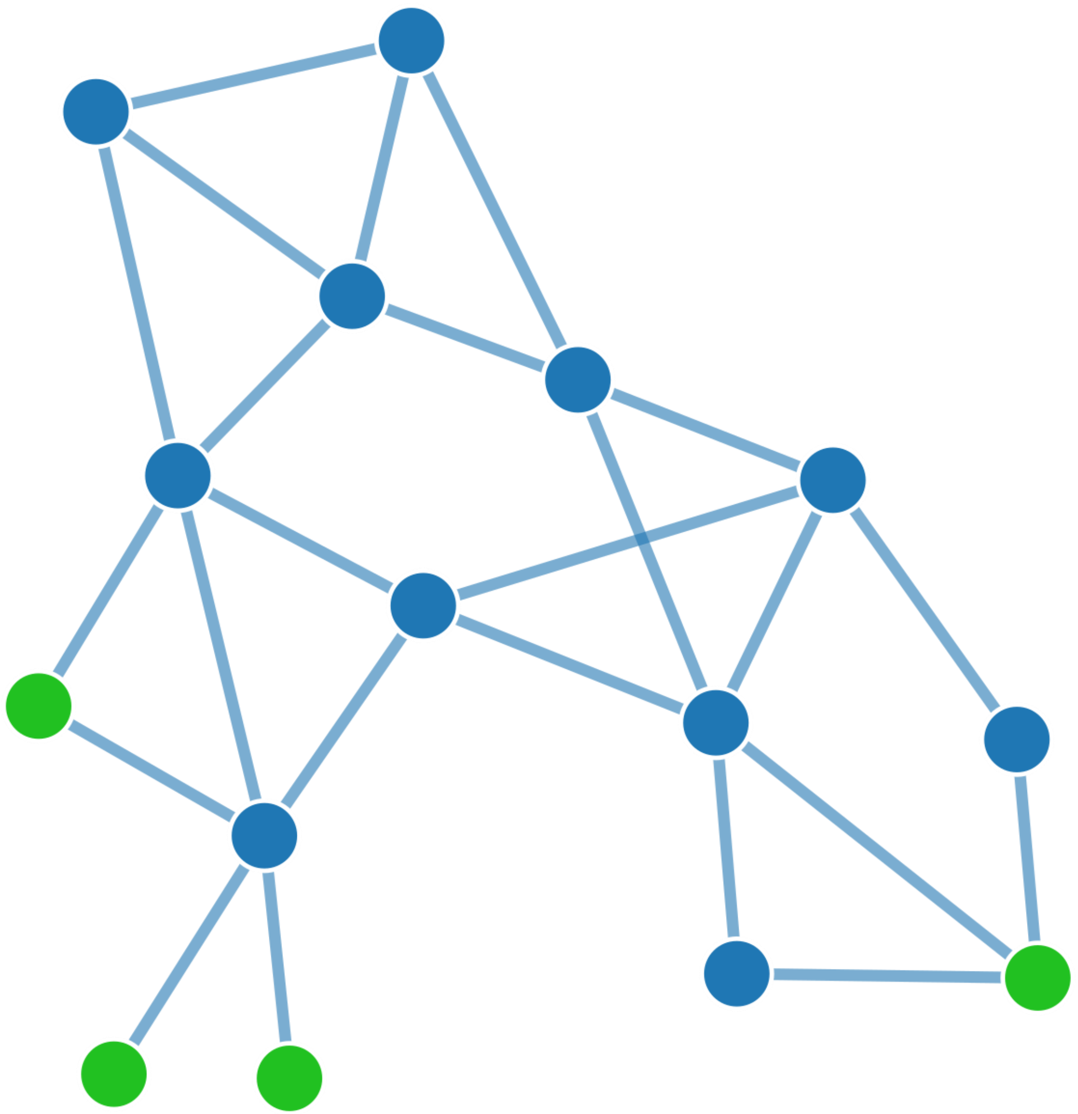


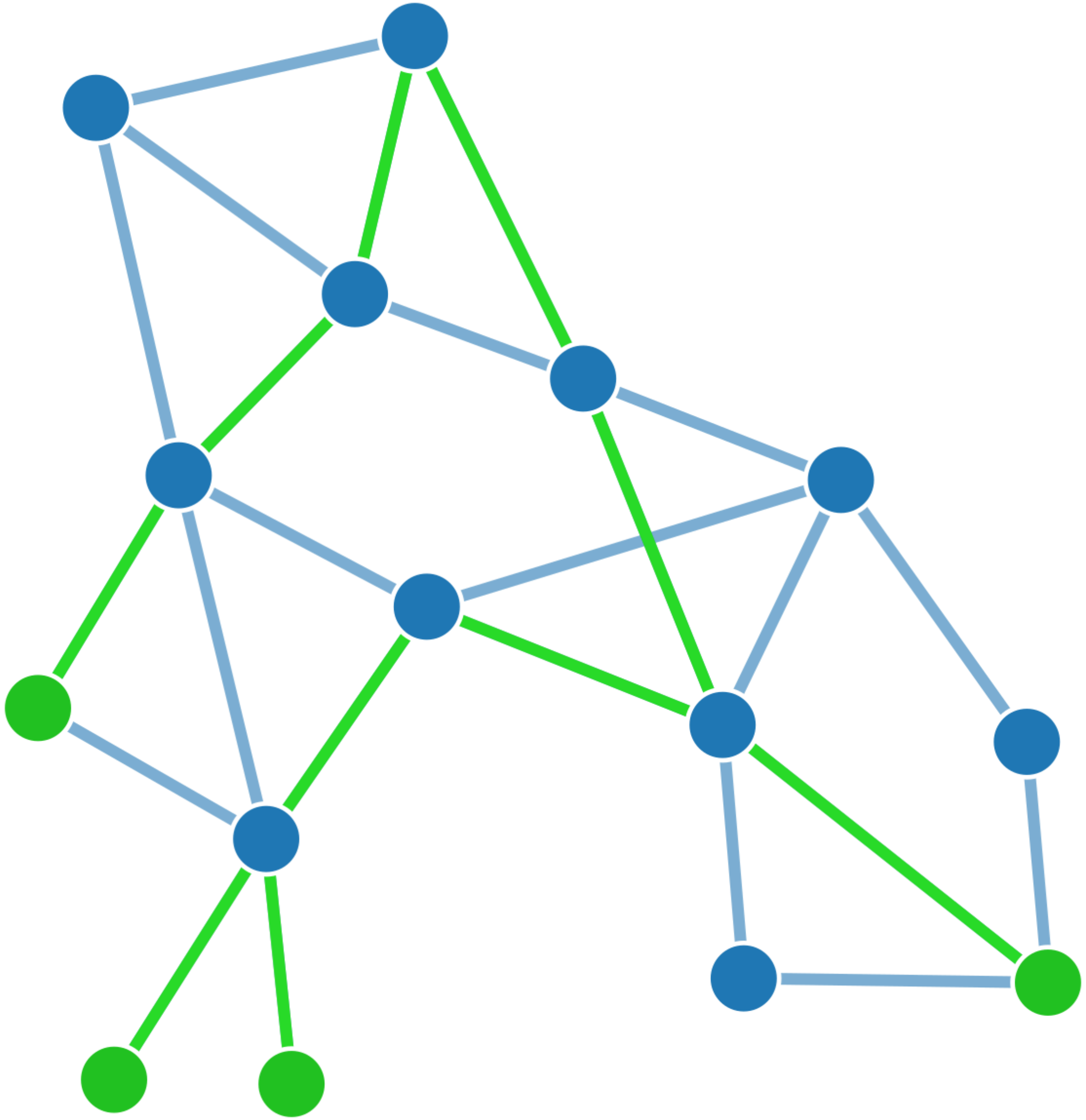
Tree-decomposition width =
Max Bag Size - 1 = 3



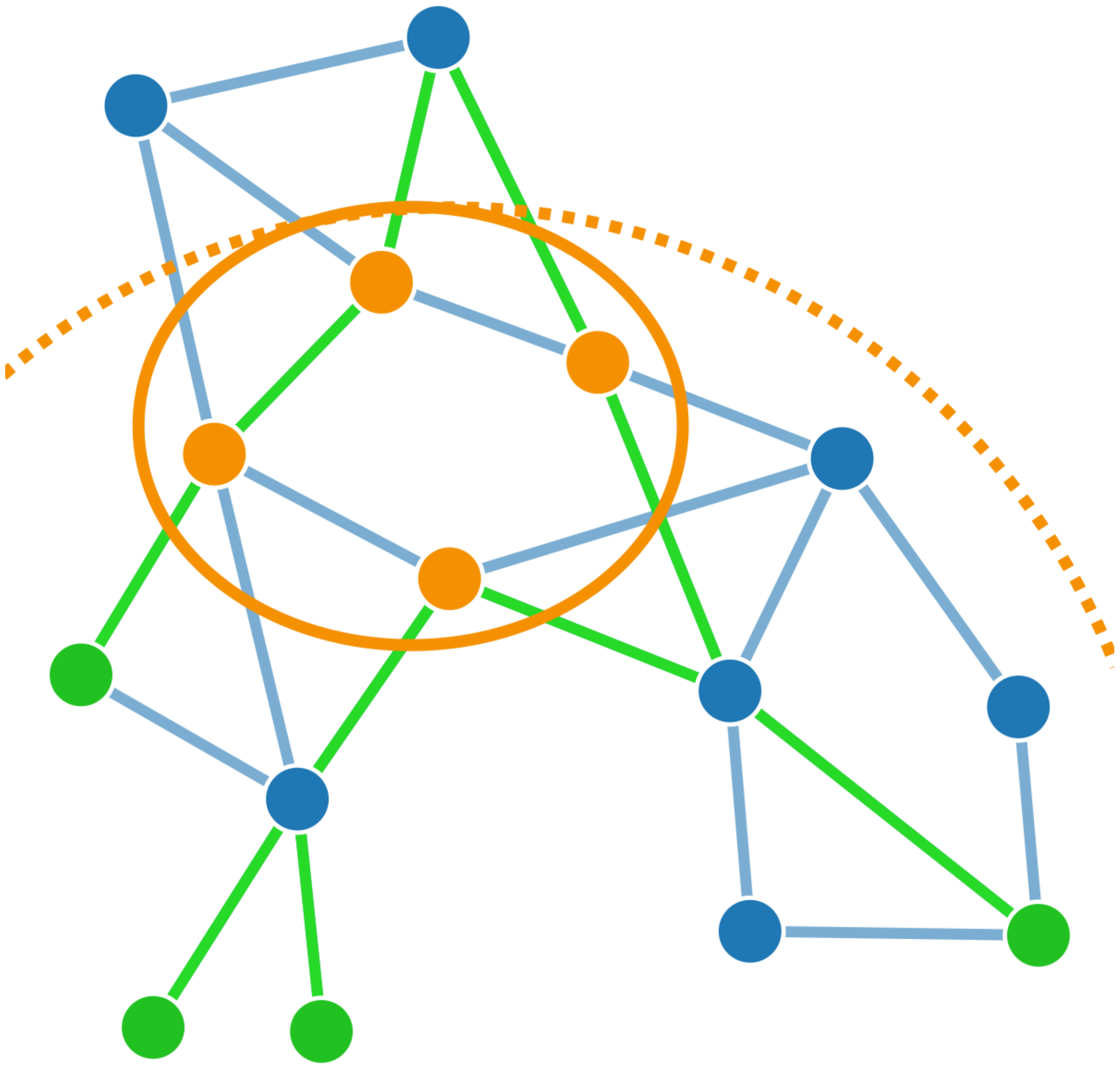
Separator
Property

Warmup : A Dynamic Program Idea for Steiner Trees

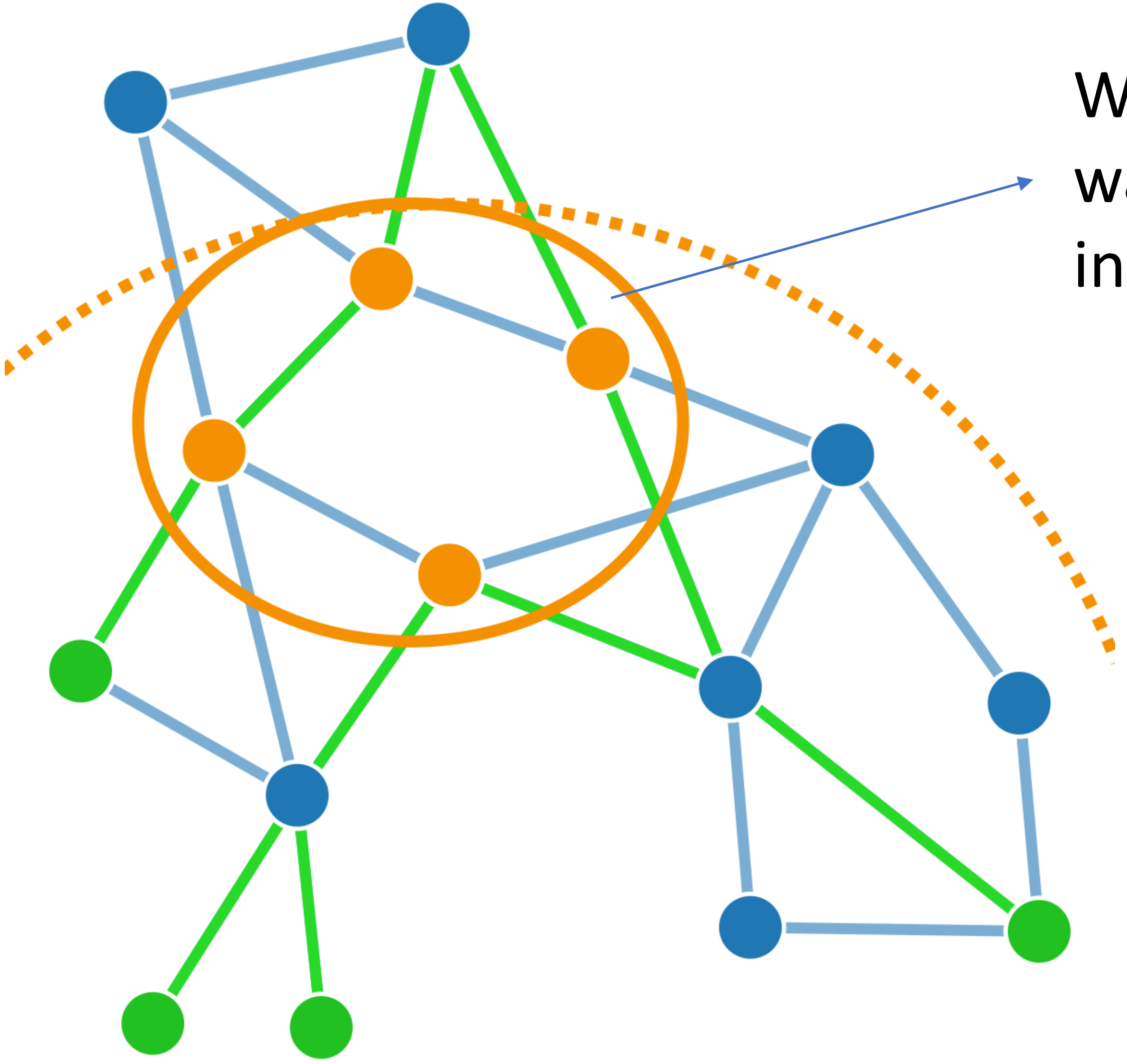




What is a subproblem that we want to solve at this bag?

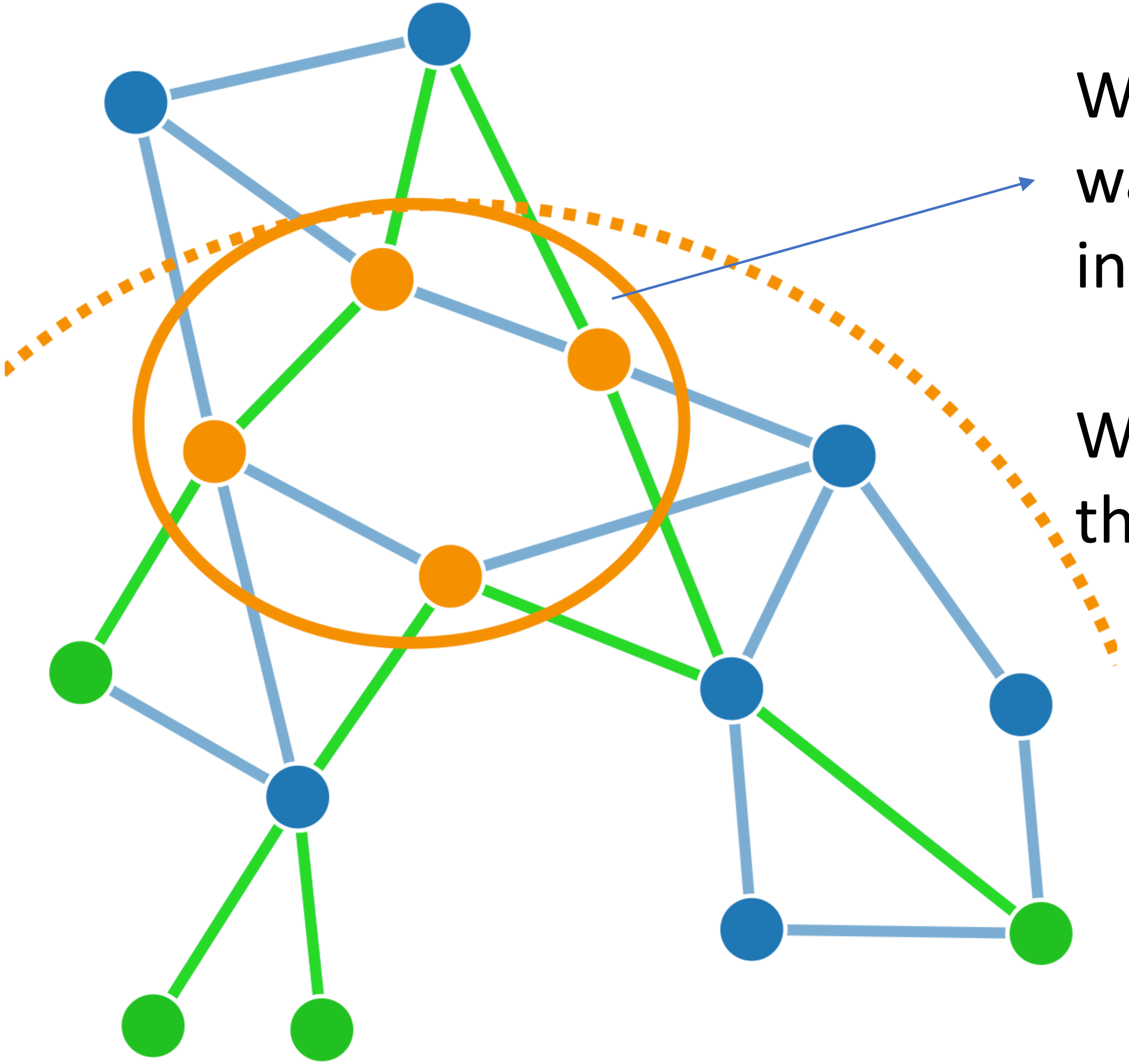


What is a subproblem that we want to solve at this bag?



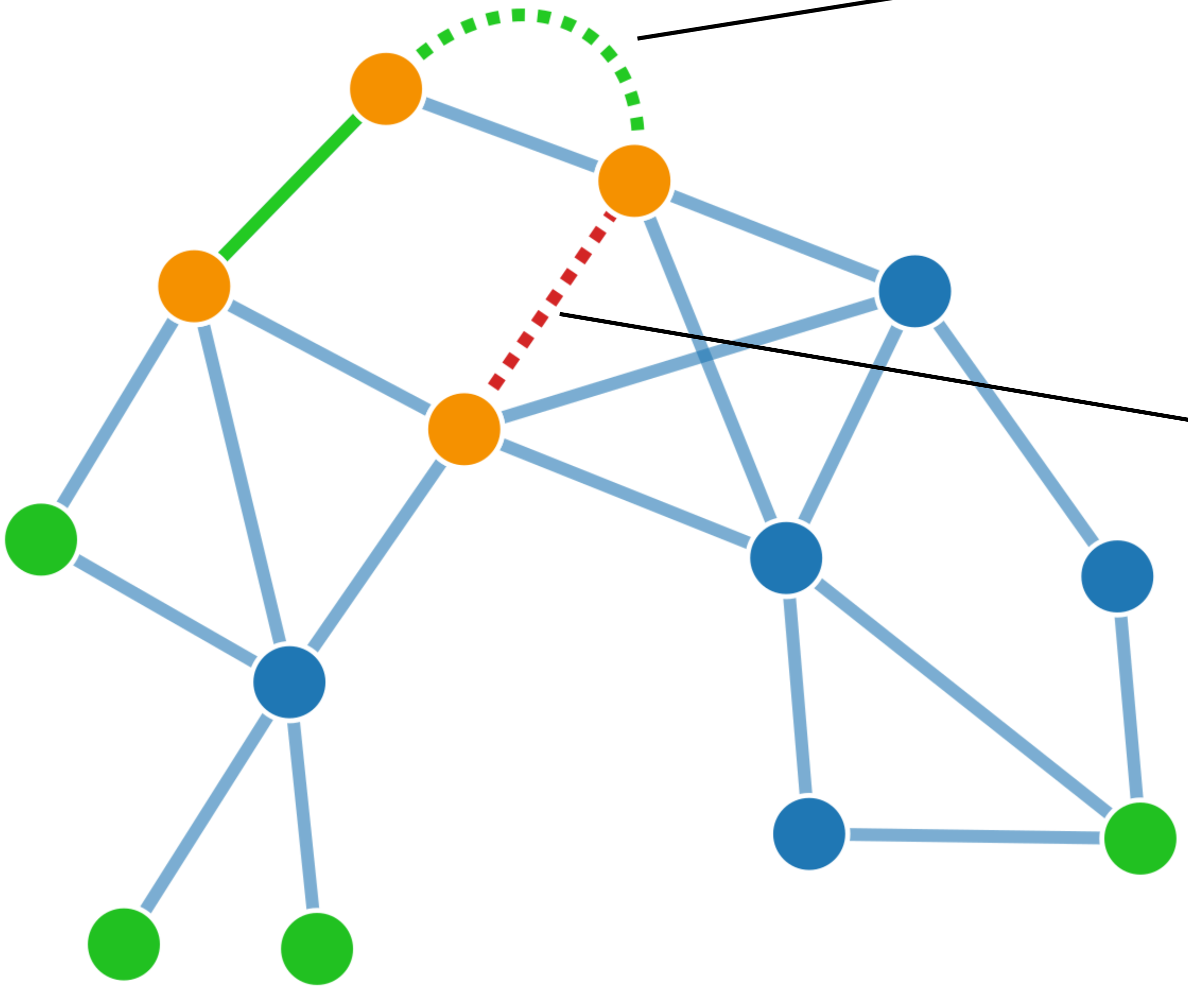
Want to find the cheapest possible way of connecting certain nodes inside the bag with each other

What is a subproblem that we want to solve at this bag?



Want to find the cheapest possible way of connecting certain nodes inside the bag with each other….

With the additional information that…



This connection will be implemented at the ancestors

This connection HAS BEEN already implemented at the children subproblem

Parameters of the Dynamic Programming Table

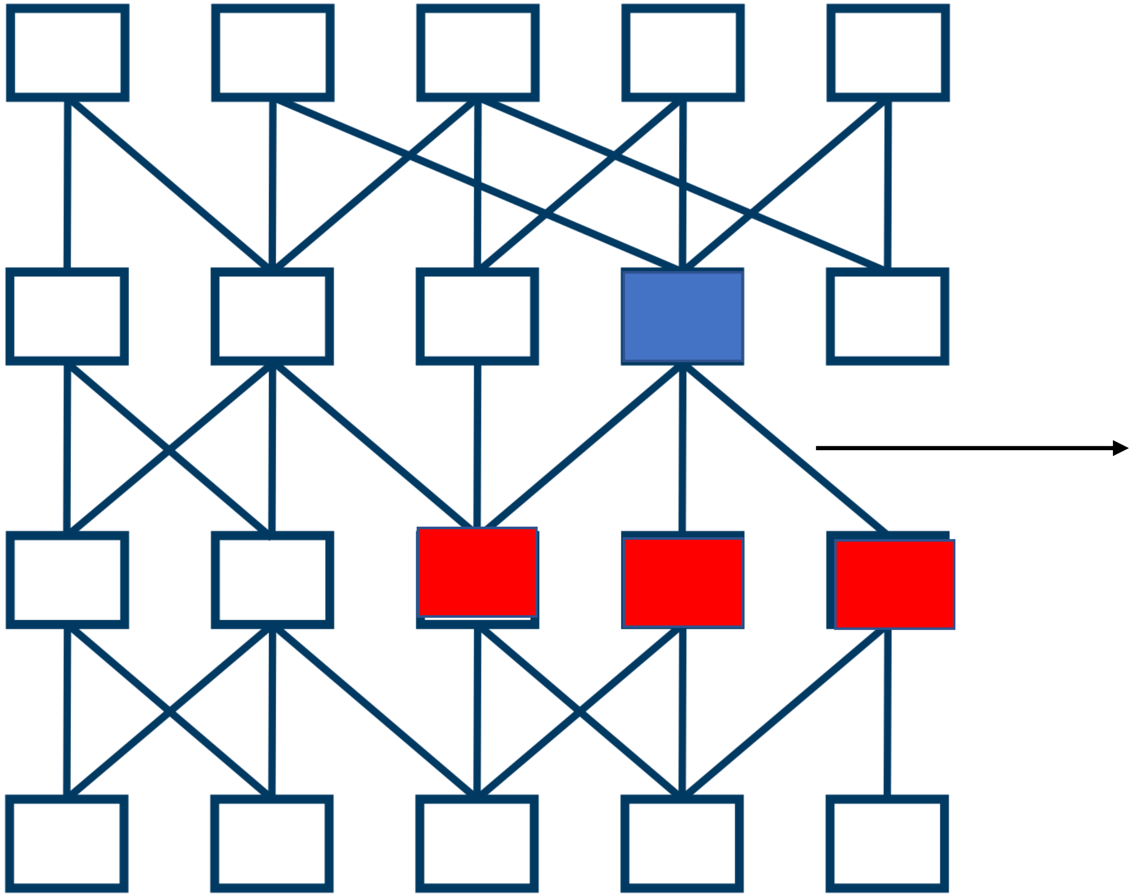
- X_t : A particular bag in the tree decomposition
- Connections implemented at ancestor nodes
- Connections that are required to be implemented at this bag (by buying some edges) and at children nodes

Parameters of the Dynamic Programming Table

- X_t : A particular bag in the tree decomposition $O(n)$
- Connections implemented at parent node $O(2^{w \log w})$
- Connections that are required to be implemented at this bag (by buying some edges) and at children nodes $O(2^{w \log w})$

Total size of DP Table ~ $n \cdot 2^{O(w \log w)}$

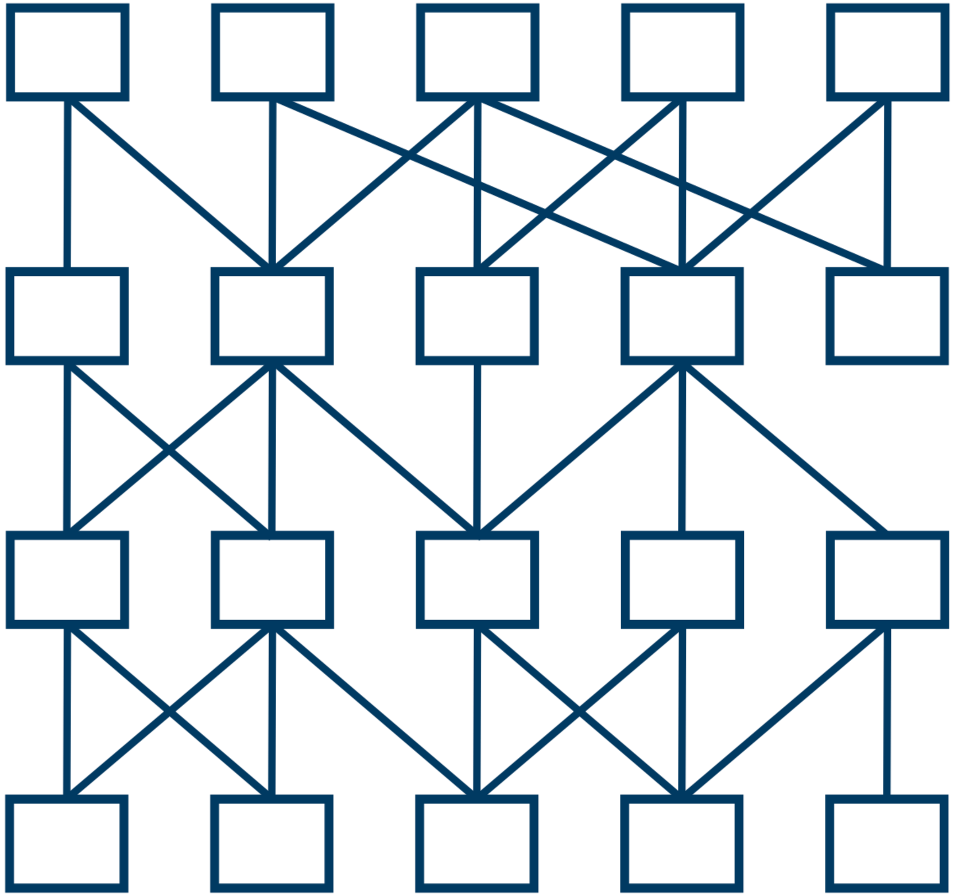
Dynamic Programming Table Represented as a Graph



An edge represents ‘consistency’ : Connections assumed at the child subproblem are correctly implemented at the parent subproblem and vice versa

Local Rule !!

Dynamic Programming Table Represented as a Graph



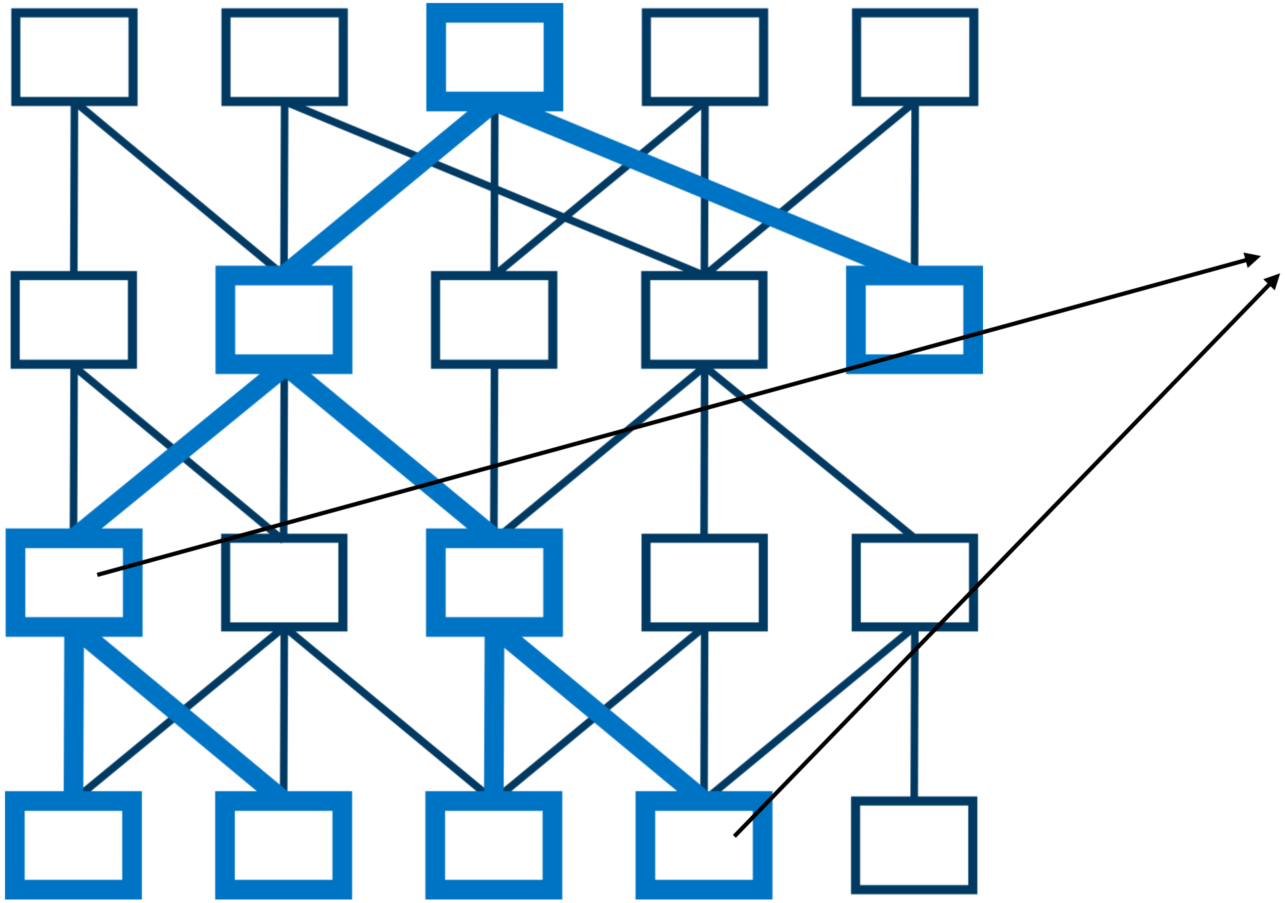
Local Rule !!

The Main Lemma :

Local Rules ⇔ Global Connectivity

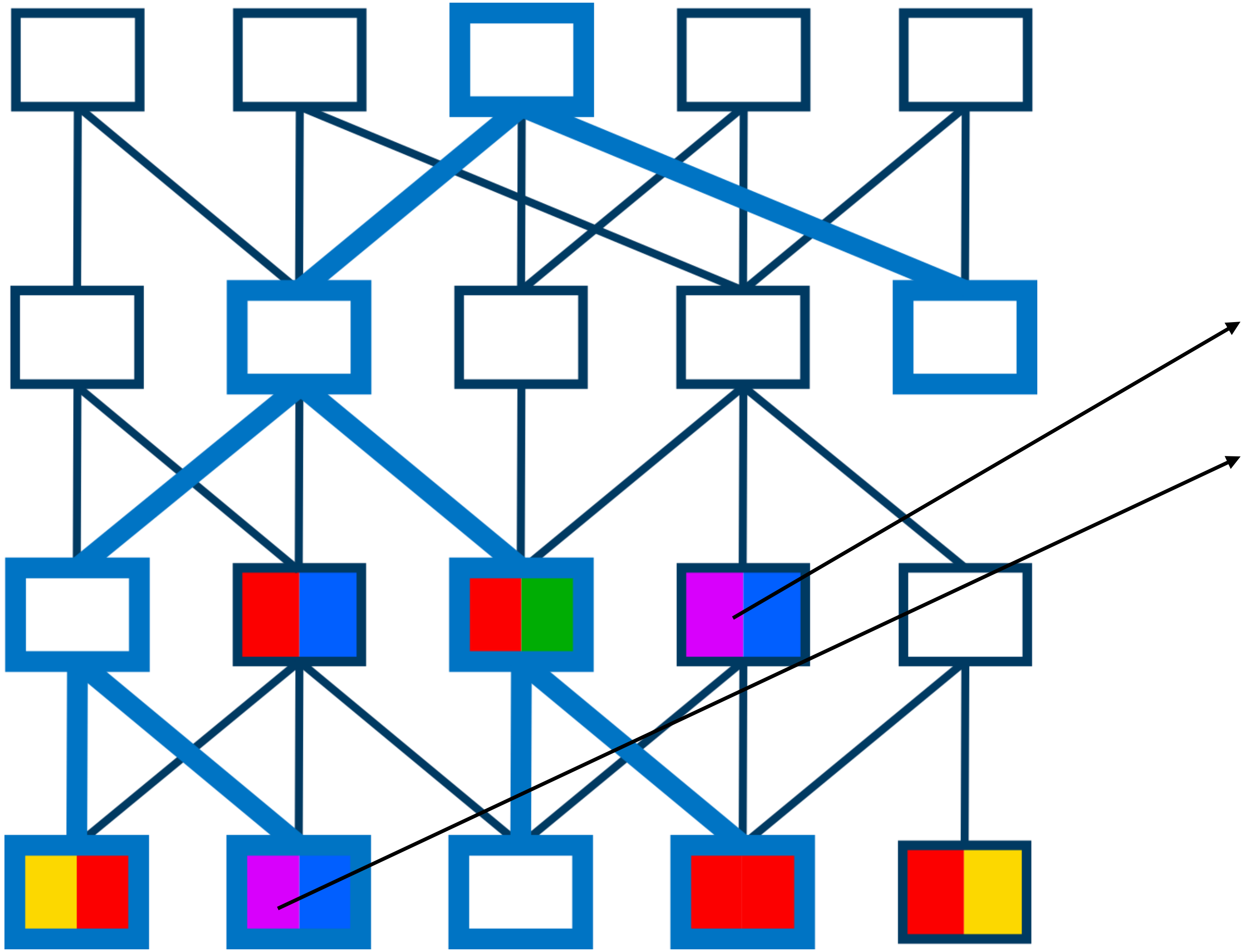
Proof : Using simple properties of tree decomposition

In Simpler Words…..



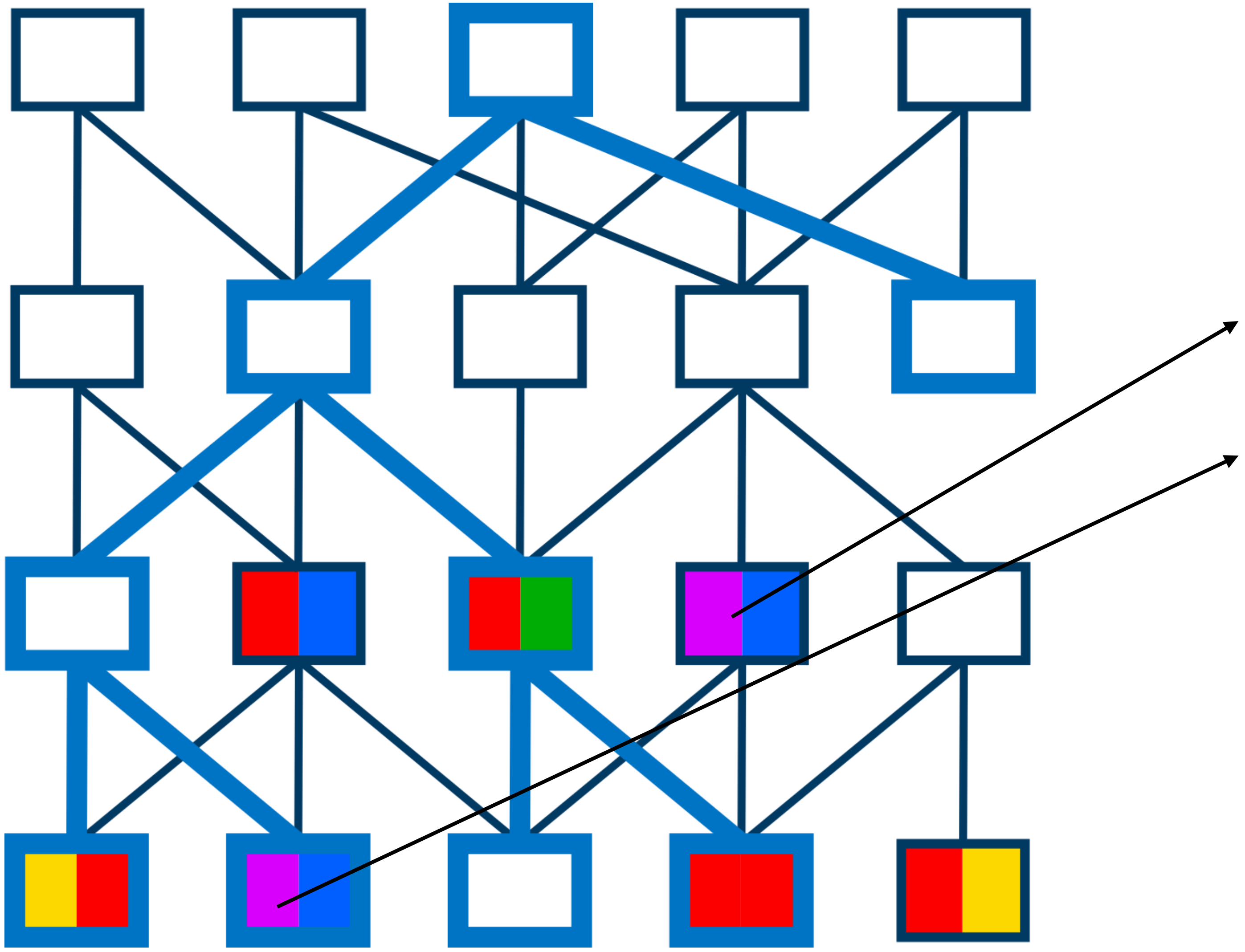
All connections that were promised here will be correctly implemented by buying edges along its path to the root subproblem

But….How do we ensure that terminals are connected ??



Subproblems that promised to connect the purple terminal with the root

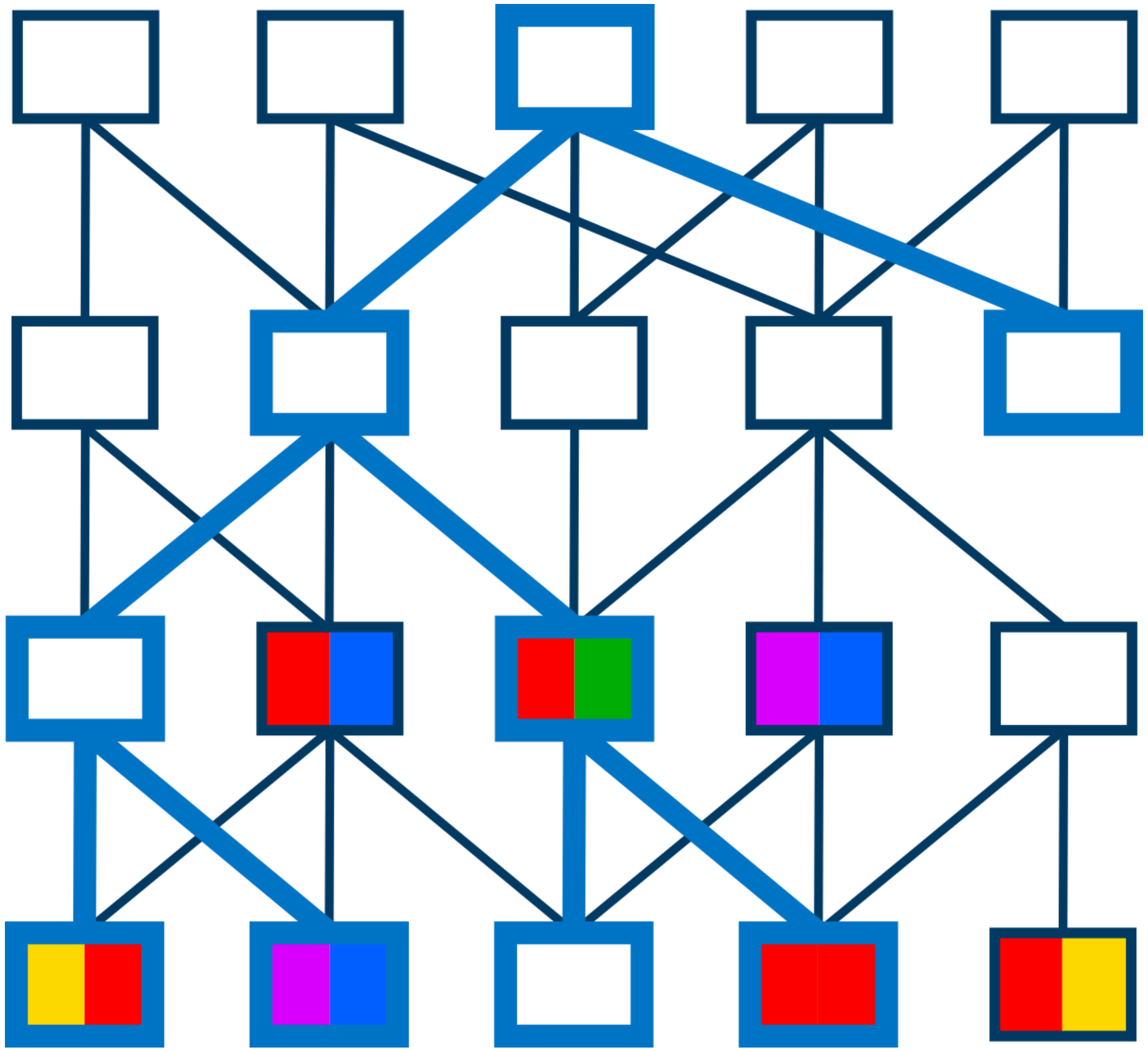
But….How do we ensure that terminals are connected ??



Subproblems that promised to connect the purple terminal with the root

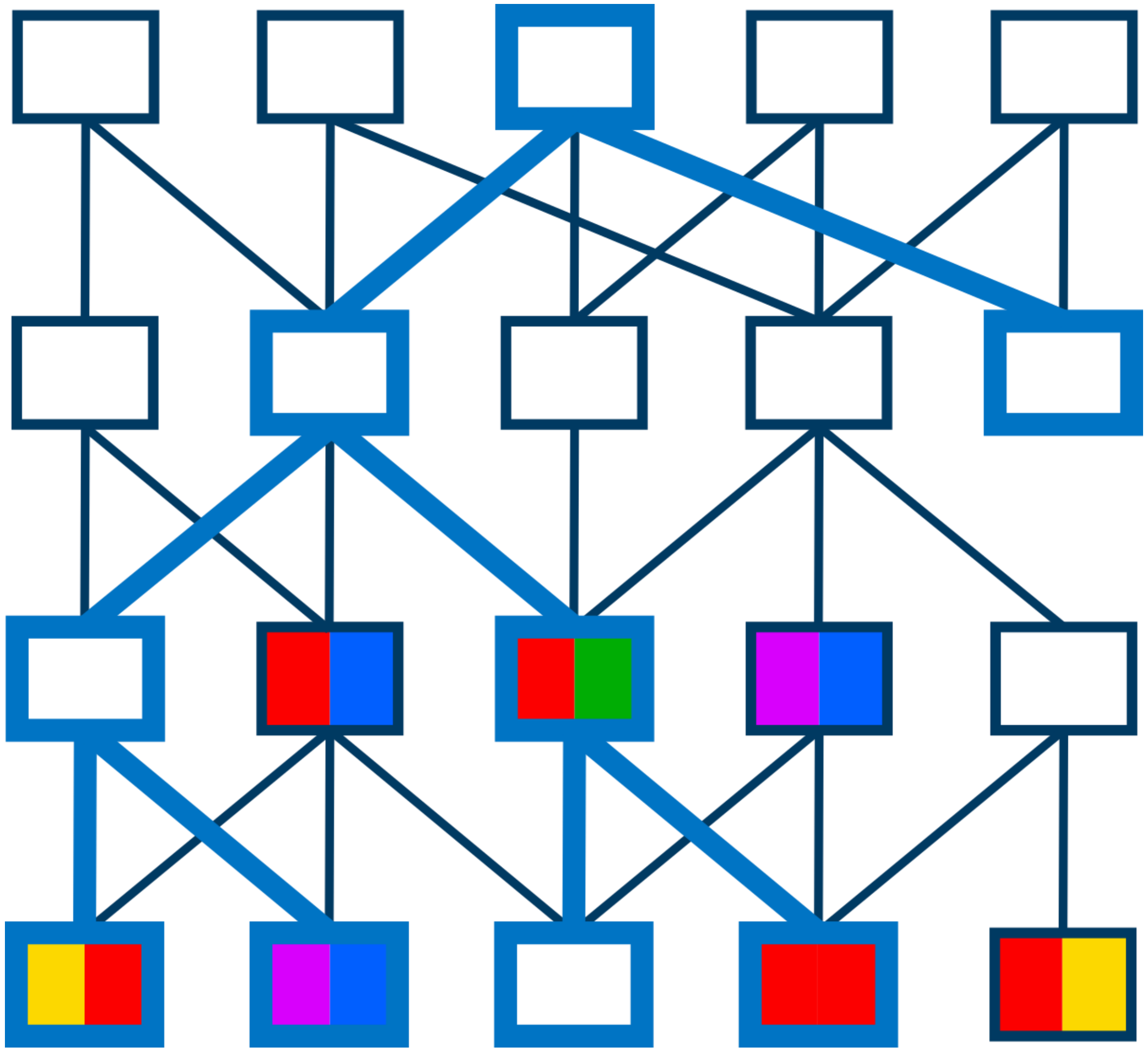
Make sure we hit at least one such node in the DP graph for each terminal

But how do we find this tree ??



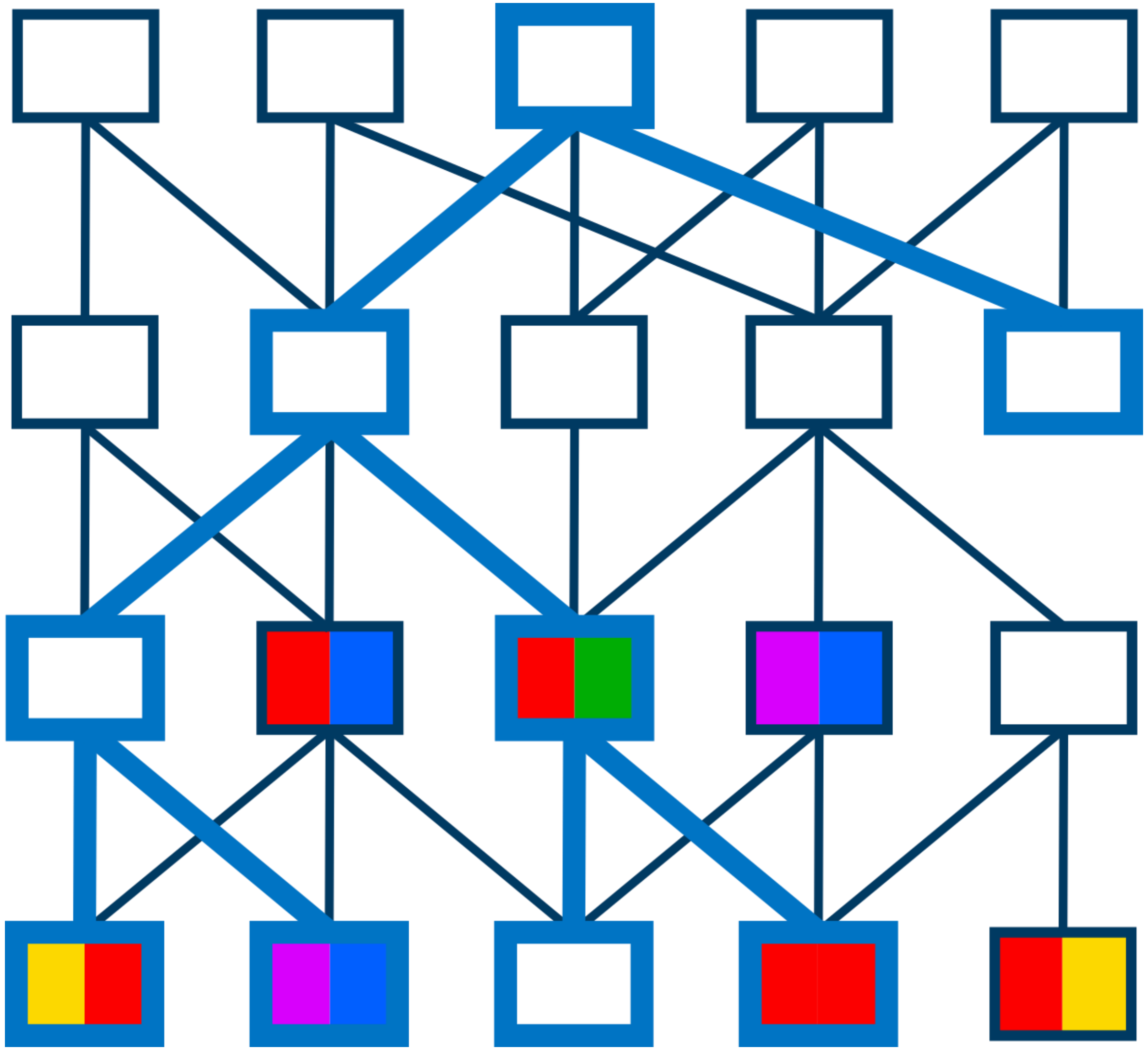
- Bad news : This boils down to solving another Group Steiner Tree Problem in the DP Graph

But how do we find this tree ??



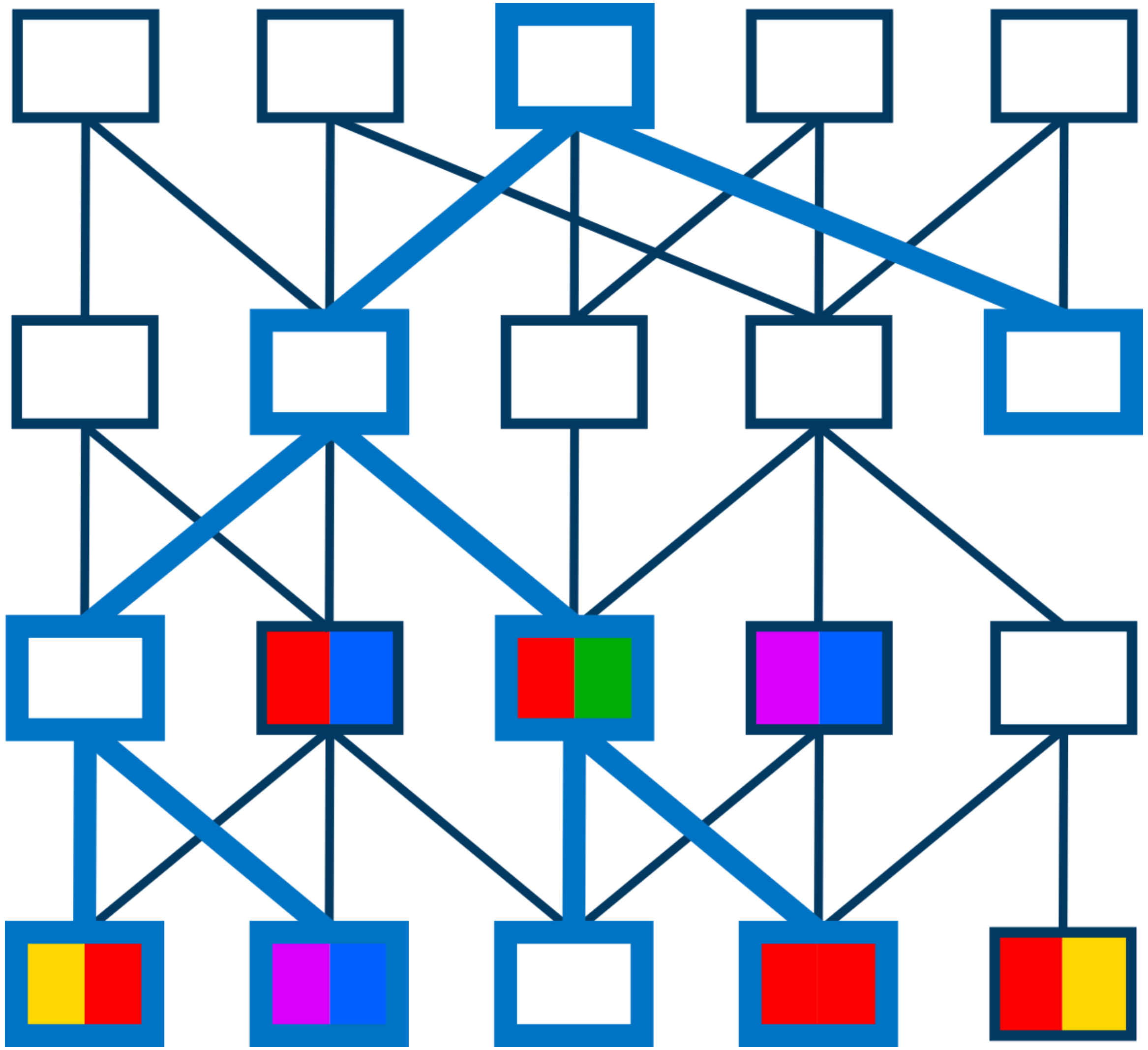
- Bad news : This boils down to solving another Group Steiner Tree Problem in the DP Graph…But…..

But how do we find this tree ??



- Bad news : This boils down to solving another Group Steiner Tree Problem in the DP Graph…But…..
- Good News : The DP Graph itself is actually a tree of size $n^{\tilde{O}(w)}$!!

But how do we find this tree ??



- Good News : The DP Graph itself is actually a tree of size $n^{\tilde{O}(w)}$!!
- Apply GKR rounding (with some additional constraints) , lose $O(\log n \log h)$-factor

Story thus far ….

- Theorem (CDLV '17)

There exists a $O(\log n \log h)$ -approximation algorithm for Group Steiner Tree running in time $n^{\widetilde{O}(w)}$

First $O(\log^2 n)$-approximation for bounded treewidth graphs

Extensions….

Theorem (CDLV '17)

There exists a $O(\log n \log^2 h)$ -approximation algorithm for Group Steiner Trees on node-weighted graphs running in time $n^{\widetilde{O}(w)}$

First polylog approximation for any non-tree graph (Metric Tree embedding does not work with node weights !)

Extensions….

Theorem (CDELV '18)

There exists a $O(\log n \log h)$-approximation algorithm for k-connected Group Steiner Problem on edge/node-weighted graphs running in time $n^{f(k,w)}$

First polylog approx for any class of graphs other than trees and k > 2

Open Problems

- FPT $O(\log^2 n)$-approximation for Group Steiner Trees (Running time $O(f(w), poly(n))$)

Open Problems

- FPT $O(\log^2 n)$-approximation for Group Steiner Trees (Running time $O(f(w), poly(n))$)

- What is the right answer for general graphs ??

$$\Omega(\log^3 h) \text{ or } O(\log^2 h)$$

Thanks !!