Julius-Maximilians-UNIVERSITÄT WÜRZBURG

Lehrstuhl für INFORMATIK I Algorithmen & Komplexität

Institut für Informatik

Exact Algorithms

Sommer Term 2020

Lecture 8. Finding Trees and Partitioning Numbers

Based on: [Exact Exp. Algos: §9.1, Param. Algos: §10.1.2]

Trees: see [J. Nederlof, Algorithmica (2013) 868–884. https://doi.org/10.1007/s00453-012-9630-x]

(slides by J. Spoerhase, Th. van Dijk, S. Chaplick, and A. Wolff)

Alexander Wolff

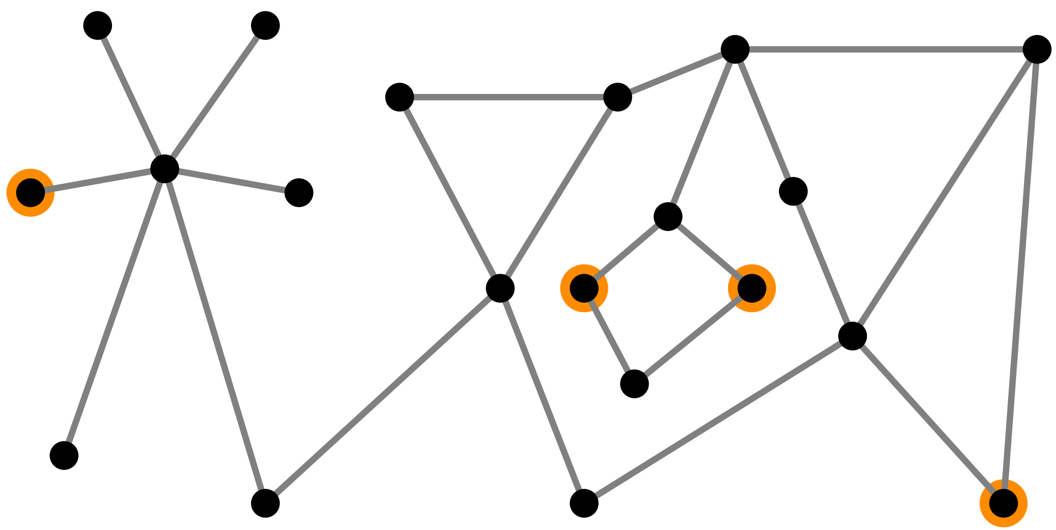
Lehrstuhl für Informatik I

Steiner Tree Problem

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?

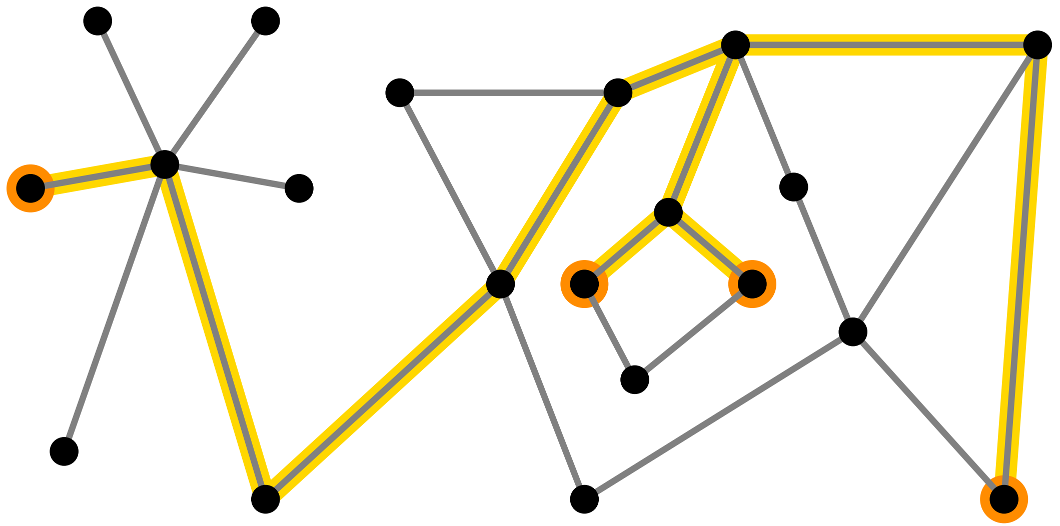


Steiner Tree Problem

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?

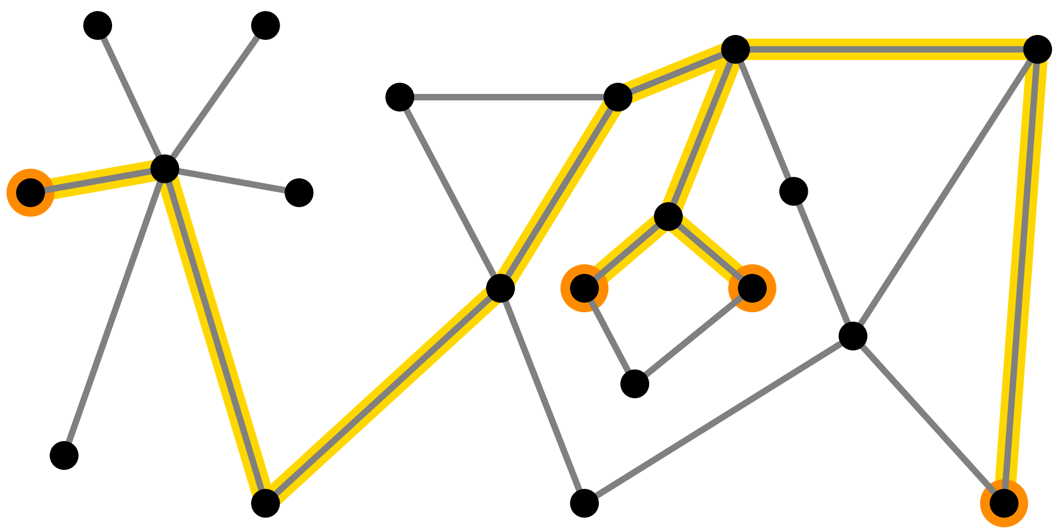


Steiner Tree Problem

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?



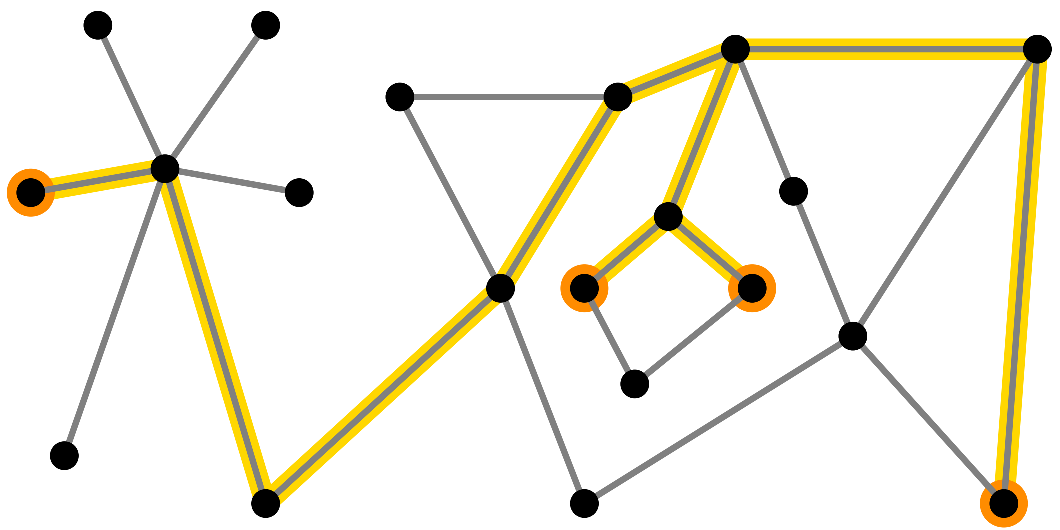
IE-Formulation?

Steiner Tree Problem

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?



IE-Formulation?

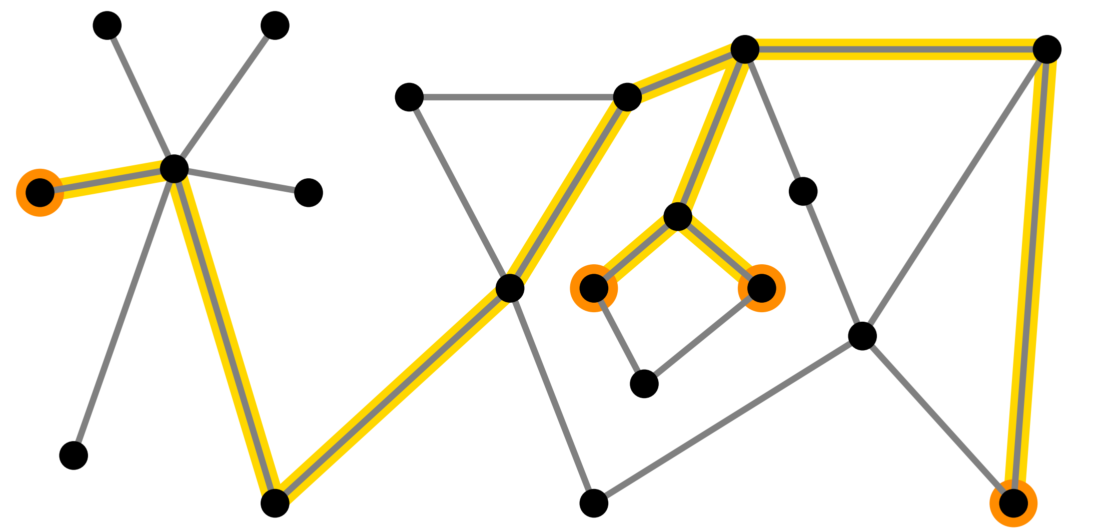
$\mathcal{U} = \{$branching walks of weight $\leq c$ in $G\}$

Steiner Tree Problem

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?



IE-Formulation?

$\mathcal{U} = \{$branching walks of weight $\leq c$ in $G\}$

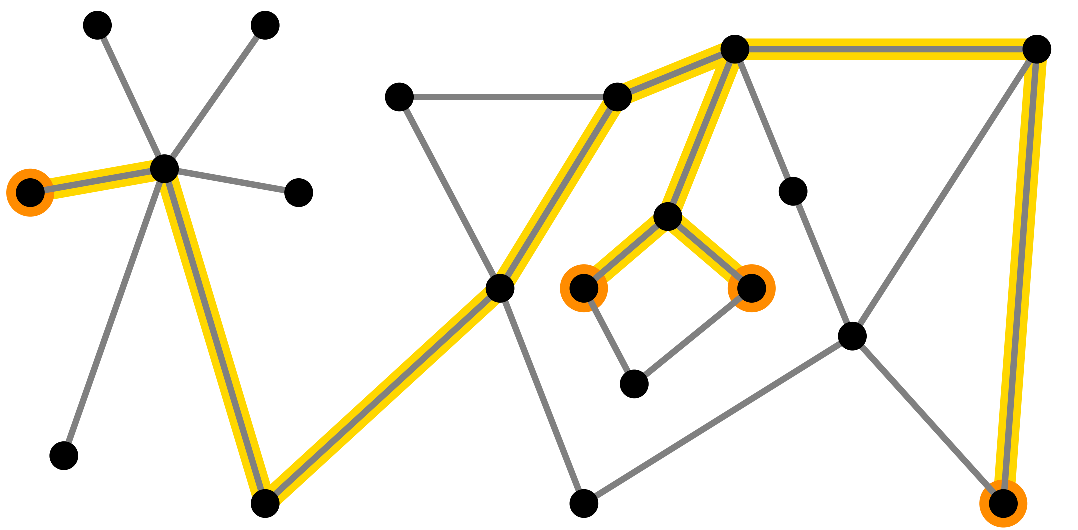
$\mathcal{P} = \{P_v \mid v \in K,$

Steiner Tree Problem

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?



IE-Formulation?

$\mathcal{U} = \{$branching walks of weight $\leq c$ in $G\}$

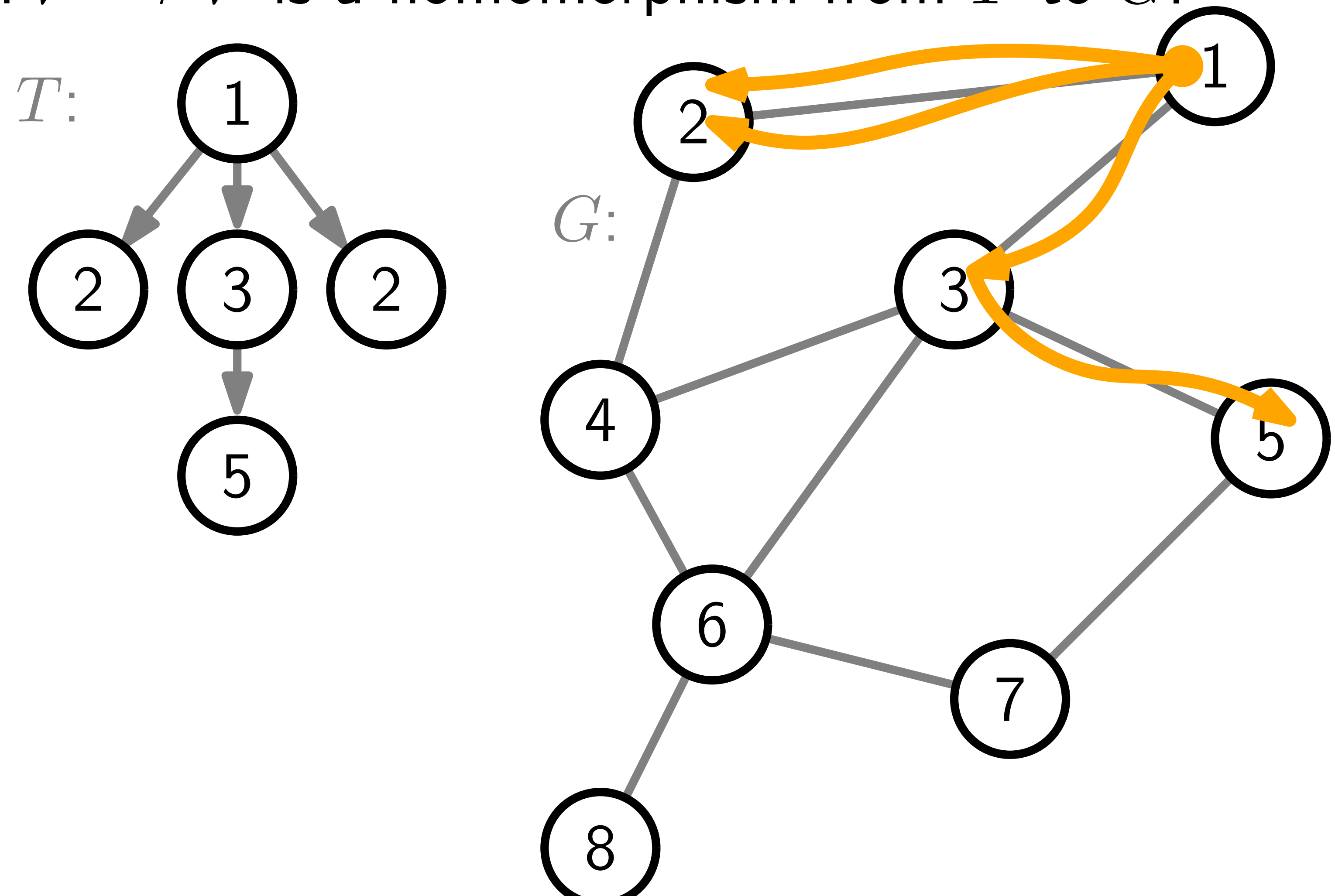
$\mathcal{P} = \{P_v \mid v \in K,$

where $P_v =$ `"branching walk contains` v`"` $\}$

Branching Walks

Def. A *branching walk* in G is a tuple (T, φ) where:

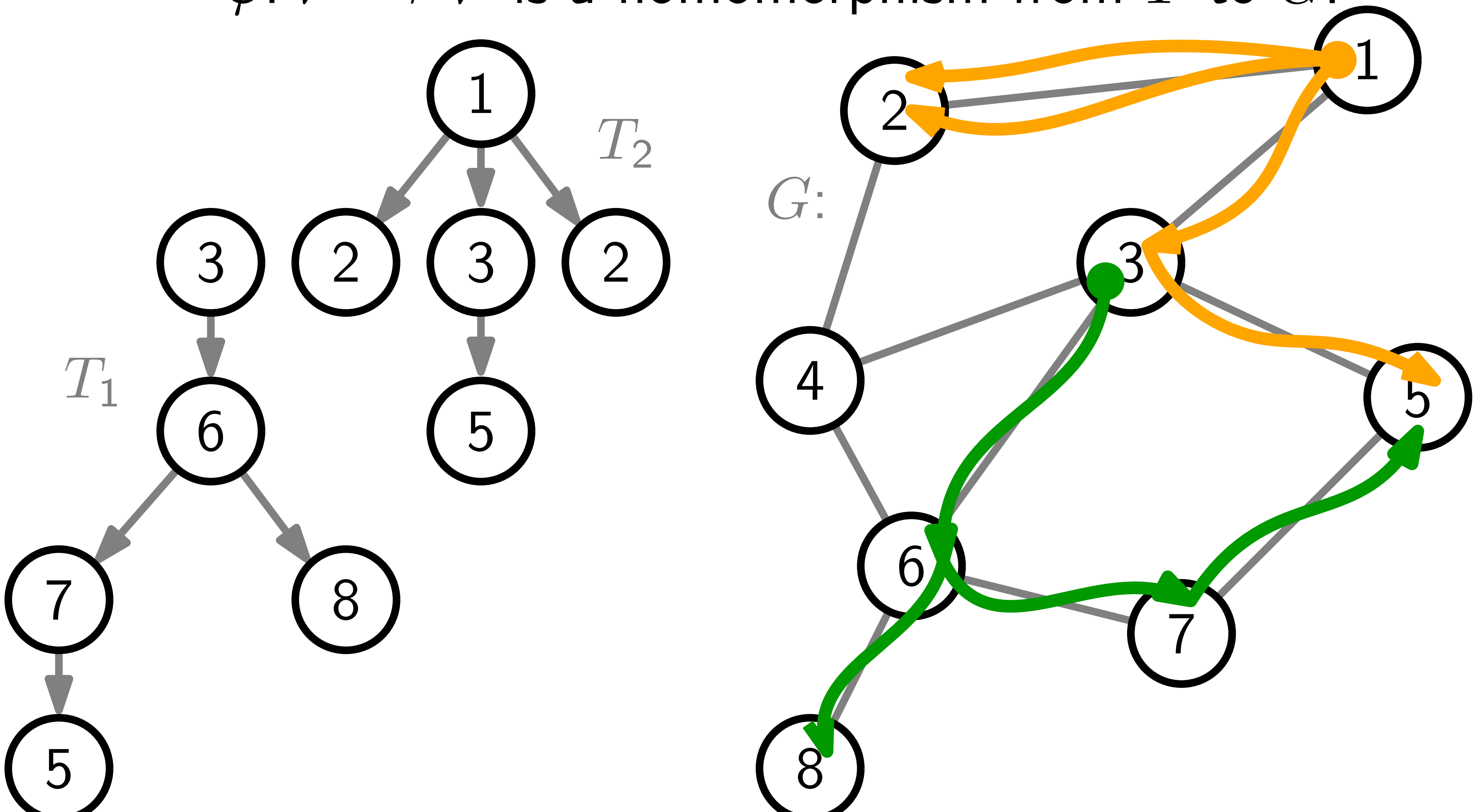
- $T = (V', E')$ is an ordered rooted tree, and
- $\varphi\colon V' \to V$ is a homomorphism from T to G.



Branching Walks

Def. A *branching walk* in G is a tuple (T, φ) where:

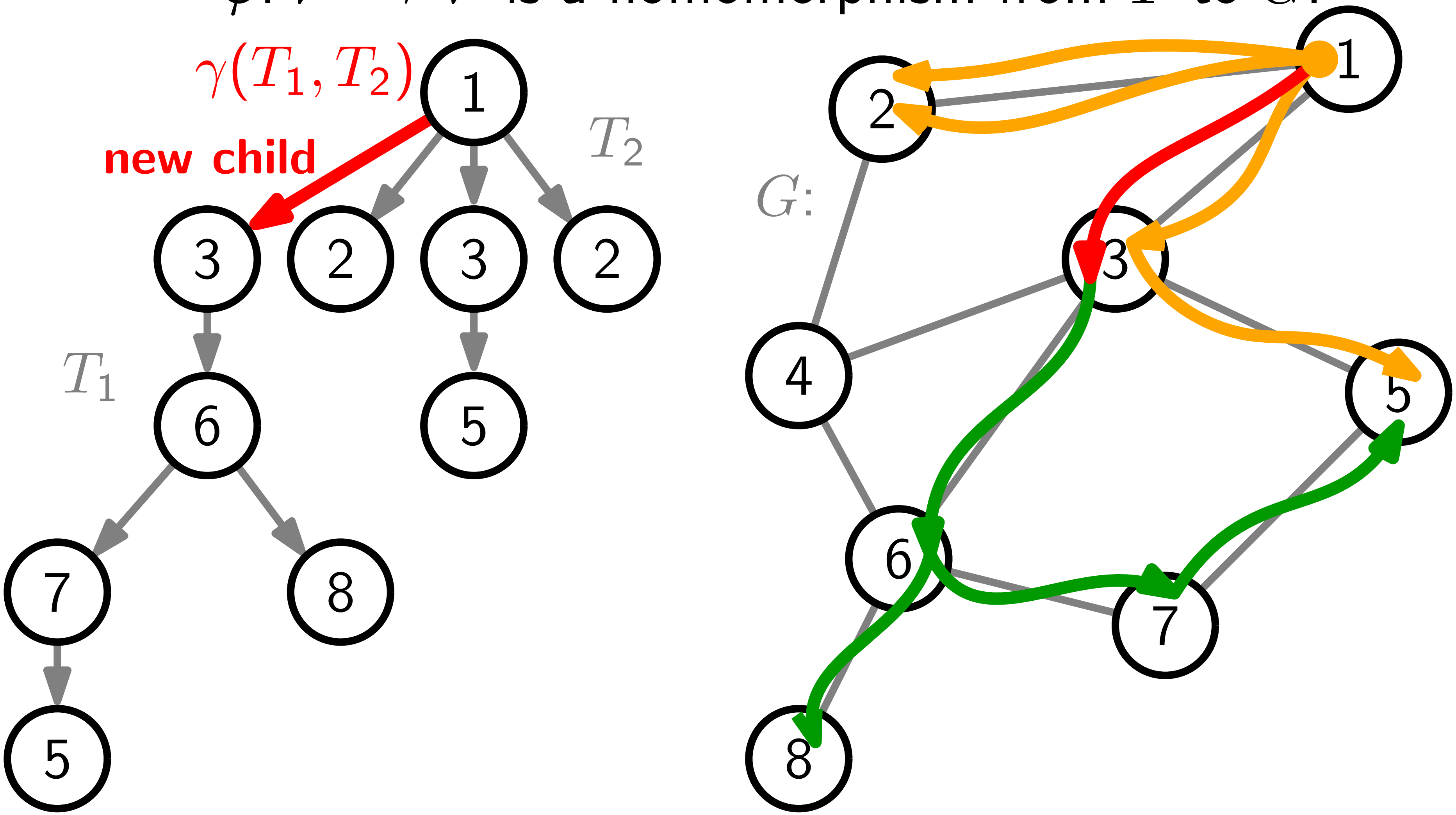
- $T = (V', E')$ is an ordered rooted tree, and
- $\varphi\colon V' \to V$ is a homomorphism from T to G.



Branching Walks

Def. A *branching walk* in G is a tuple (T, φ) where:

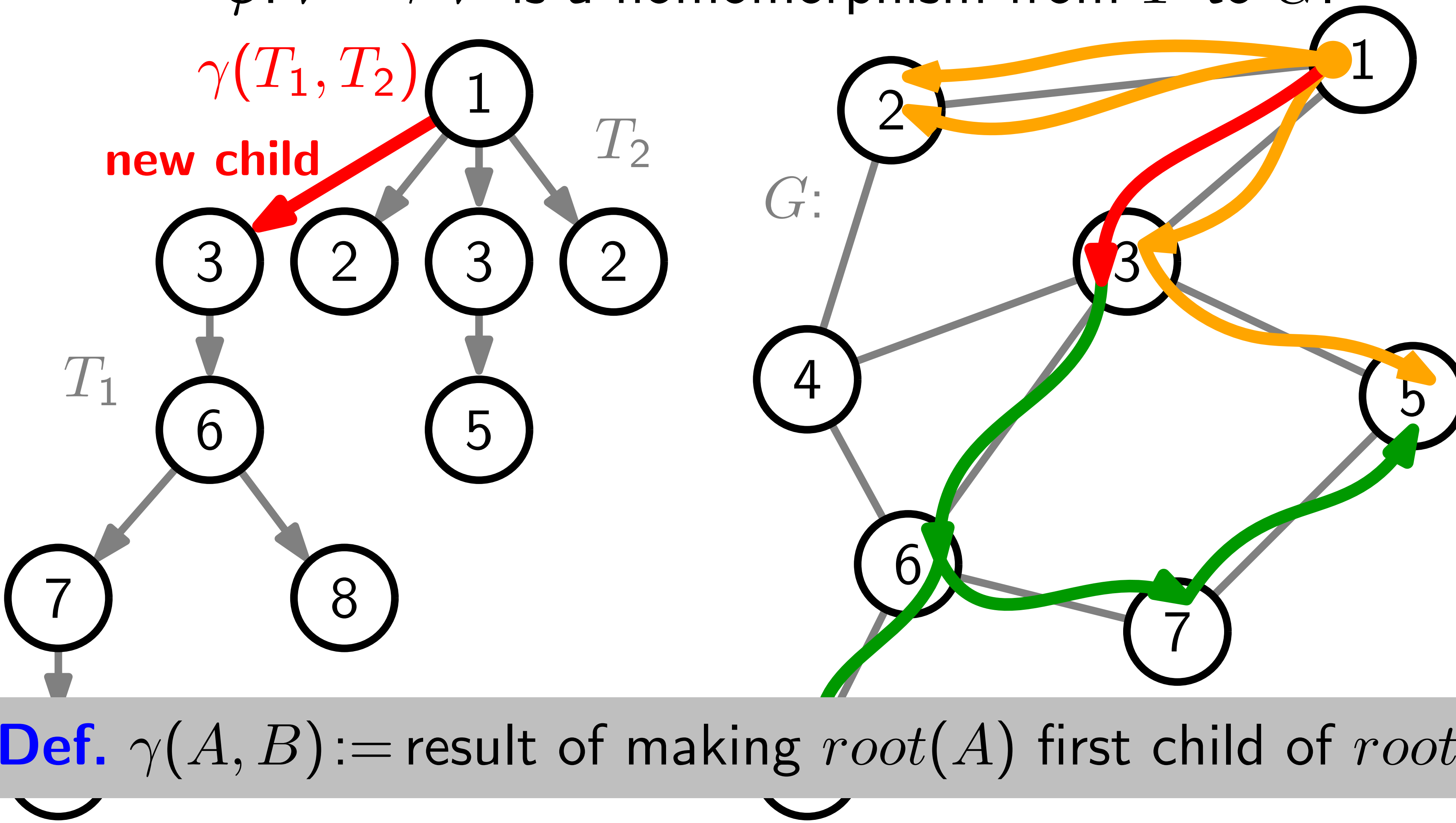
- $T = (V', E')$ is an ordered rooted tree, and
- $\varphi\colon V' \to V$ is a homomorphism from T to G.



Branching Walks

Def. A *branching walk* in G is a tuple (T, φ) where:

- $T = (V', E')$ is an ordered rooted tree, and
- $\varphi\colon V' \to V$ is a homomorphism from T to G.



Def. $\gamma(A, B) :=$ result of making $root(A)$ first child of $root(B)$

Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n$: $\exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j)$

Tree Counting

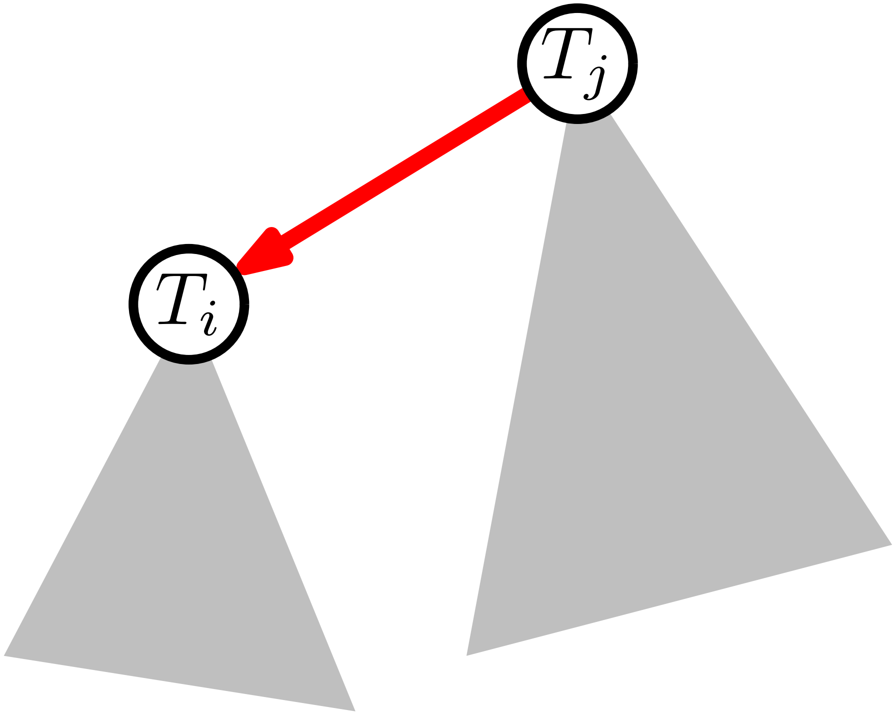
Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n$: $\exists!$ $i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j) \ldots i + j =$?

Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

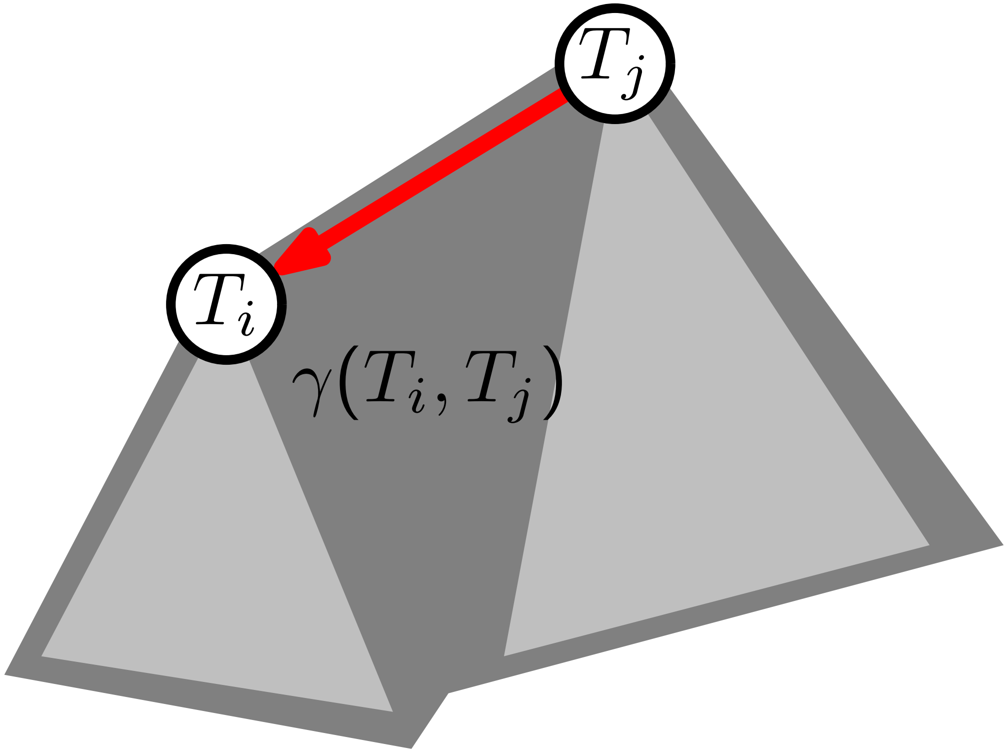
Obs. $\forall T \in \mathcal{T}_n \colon \exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j) \ldots i + j =$?



Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n \colon \exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j) \ldots i + j =$?



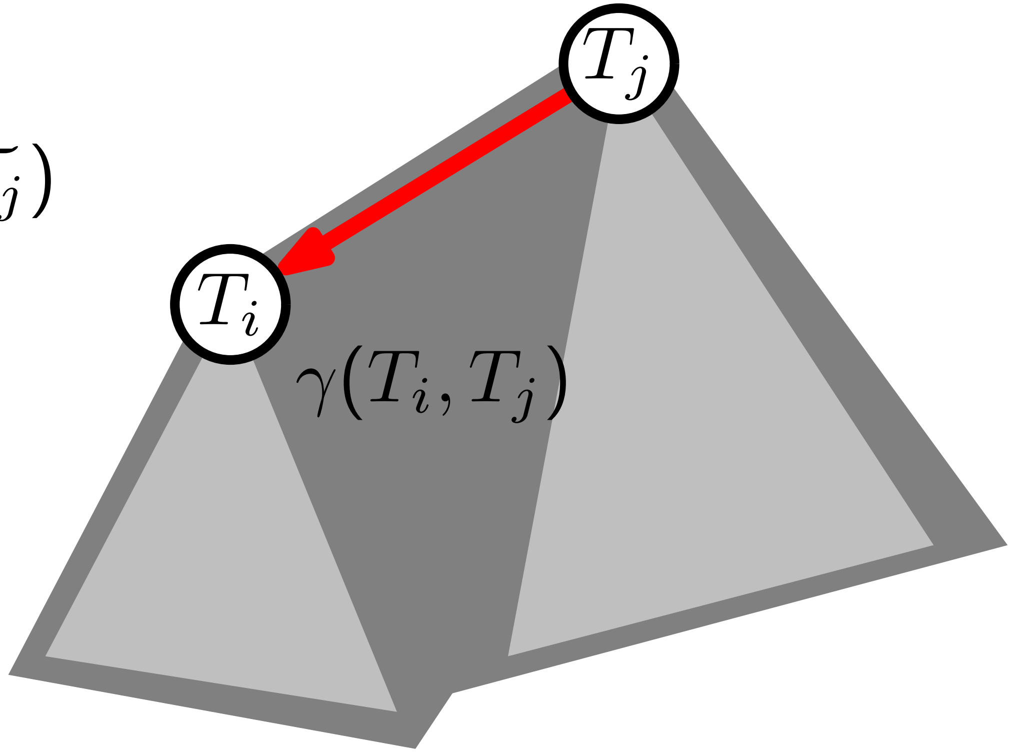
Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n \colon \exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j) \ldots i + j =$?

Bijective mapping:

$$\mathcal{T}_n \leftrightarrow \bigcup_{i+j=n-1} (\mathcal{T}_i \times \mathcal{T}_j)$$



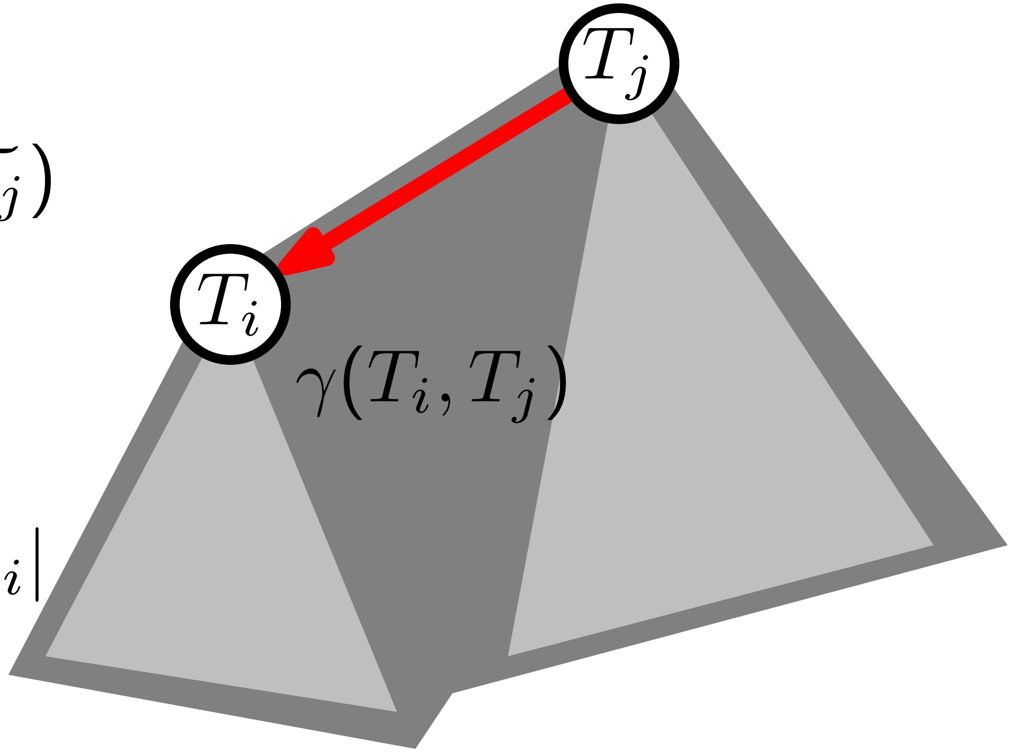
Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n \colon \exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j) \ldots i + j = ?$

Bijective mapping:

$$\mathcal{T}_n \leftrightarrow \bigcup_{i+j=n-1} (\mathcal{T}_i \times \mathcal{T}_j)$$



Theorem.

$$|\mathcal{T}_n| = \sum_{i=0}^{n-1} |\mathcal{T}_i| \cdot |\mathcal{T}_{n-1-i}|$$

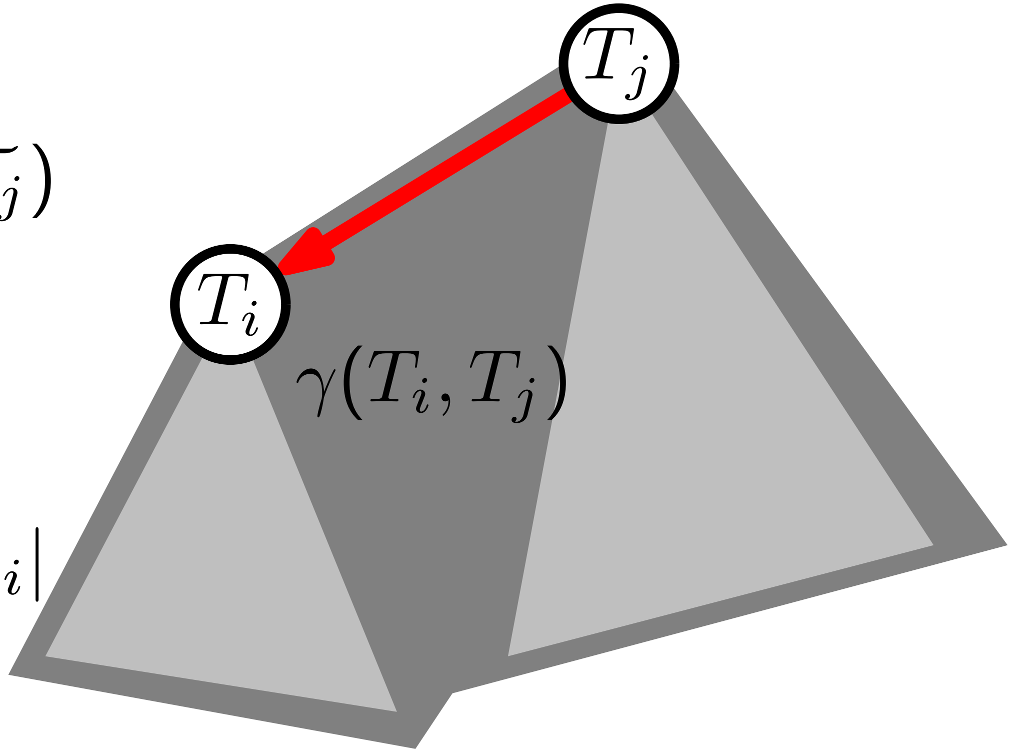
Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n \colon \exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j) \ldots i + j = ?$

Bijective mapping:

$$\mathcal{T}_n \leftrightarrow \bigcup_{i+j=n-1} (\mathcal{T}_i \times \mathcal{T}_j)$$



Theorem.

$$|\mathcal{T}_n| = \sum_{i=0}^{n-1} |\mathcal{T}_i| \cdot |\mathcal{T}_{n-1-i}|$$

$$|\mathcal{T}_0| = ?$$

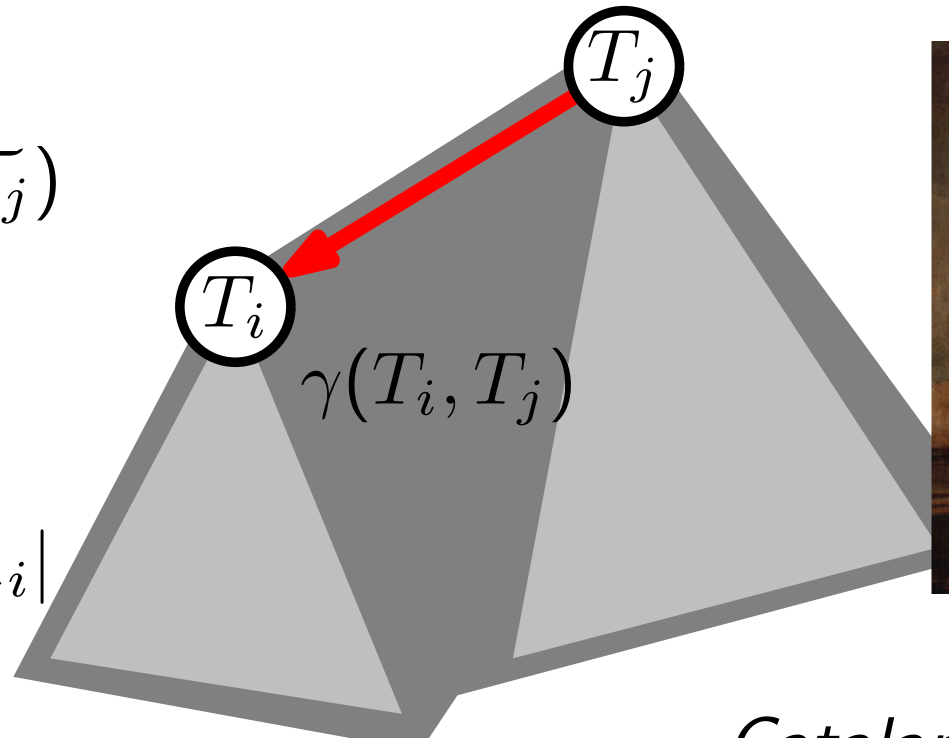
Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n$: $\exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j)$ … $i + j =$?

Bijective mapping:

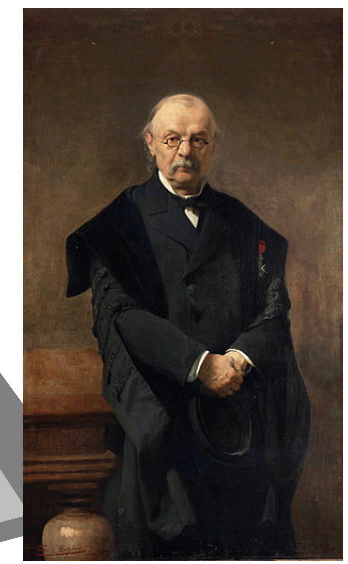
$$\mathcal{T}_n \ \leftrightarrow \bigcup_{i+j=n-1} (\mathcal{T}_i \times \mathcal{T}_j)$$



Theorem.

$$|\mathcal{T}_n| = \sum_{i=0}^{n-1} |\mathcal{T}_i| \cdot |\mathcal{T}_{n-1-i}|$$

$|\mathcal{T}_0| =$?



Recall:
Catalan numbers

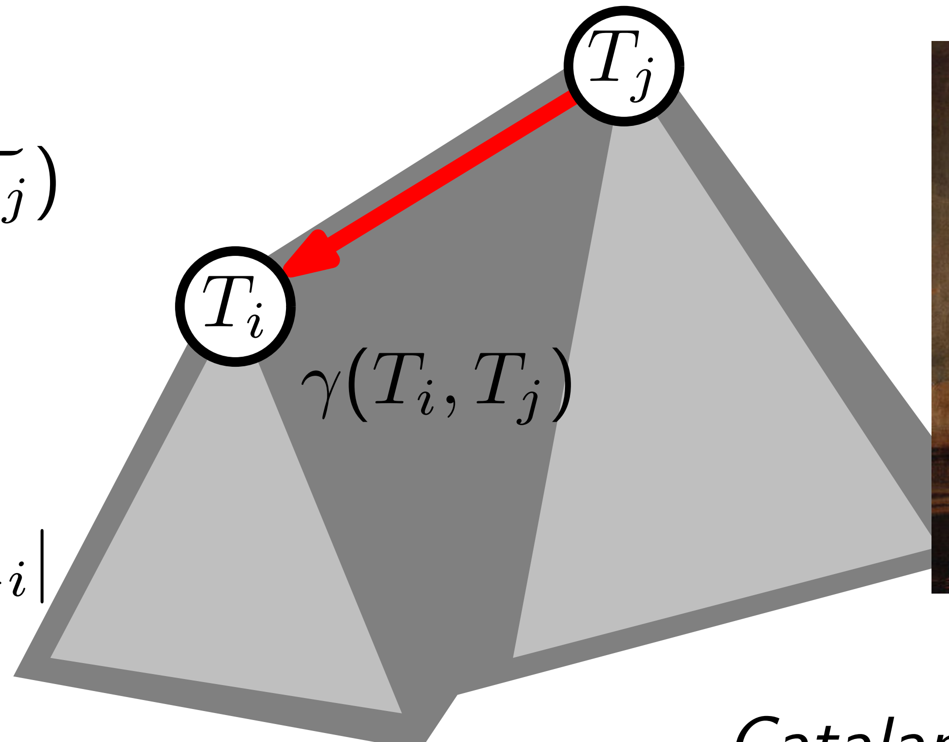
Tree Counting

Def. $\mathcal{T}_n :=$ set of different ordered rooted trees with n edges.

Obs. $\forall T \in \mathcal{T}_n \colon \exists!\ i, j \in \mathbb{N},\ T_i \in \mathcal{T}_i,\ T_j \in \mathcal{T}_j$
such that $T = \gamma(T_i, T_j) \ldots i + j = ?$

Bijective mapping:

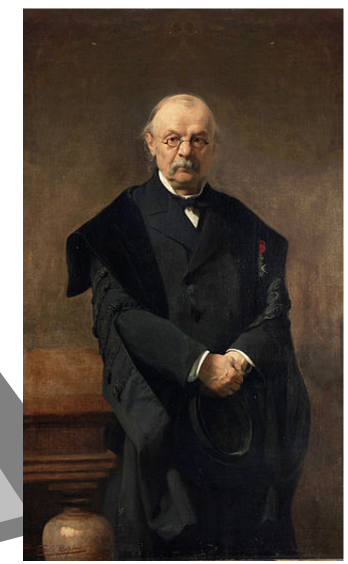
$$\mathcal{T}_n \leftrightarrow \bigcup_{i+j=n-1} (\mathcal{T}_i \times \mathcal{T}_j)$$



Theorem.

$$|\mathcal{T}_n| = \sum_{i=0}^{n-1} |\mathcal{T}_i| \cdot |\mathcal{T}_{n-1-i}|$$

$$|\mathcal{T}_0| = ?$$



Recall:
Catalan numbers

$C_0 = 1; C_n = \sum_{i=0}^{n-1} C_i C_{n-1-i} \sim \frac{4^n}{n^{3/2}\sqrt{\pi}}$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees. . . still need φ.

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c)$:= all branching walks of type (T, φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c) :=$ all branching walks of type (T, φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

Let $b_F(x, c) := |\mathcal{B}_F(x, c)|$.

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x,c)$:= all branching walks of type (T,φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

Let $b_F(x,c) := |\mathcal{B}_F(x,c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x,c) = \begin{cases} ? & \text{if } c = 0, \\ & \\ & \end{cases}$$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c)$:= all branching walks of type (T, φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

Let $b_F(x, c) := |\mathcal{B}_F(x, c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x, c) = \begin{cases} 1 & \text{if } c = 0, \\ & \\ & \end{cases}$$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c)$:= all branching walks of type (T, φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

Let $b_F(x, c) := |\mathcal{B}_F(x, c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x, c) = \begin{cases} 1 & \text{if } c = 0, \\ \displaystyle\sum_{t \in N(x) \setminus F} \sum_{c_1 + c_2 \leq c - 1} b_F(t, c_1) \cdot b_F(x, c_2) & \text{otherwise.} \end{cases}$$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c)$:= all branching walks of type (T, φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

Let $b_F(x, c) := |\mathcal{B}_F(x, c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x, c) = \begin{cases} 1 & \text{if } c = 0, \\ \displaystyle\sum_{t \in N(x) \setminus F} \; \sum_{c_1 + c_2 \leq c - 1} b_F(t, c_1) \cdot b_F(x, c_2) & \text{otherwise.} \end{cases}$$

Runtime:

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c) :=$ all branching walks of type (T, φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

Let $b_F(x, c) := |\mathcal{B}_F(x, c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x, c) = \begin{cases} 1 & \text{if } c = 0, \\ \displaystyle\sum_{t \in N(x) \setminus F} \; \sum_{c_1 + c_2 \leq c - 1} b_F(t, c_1) \cdot b_F(x, c_2) & \text{otherwise.} \end{cases}$$

Runtime: $O(n^2 \cdot n^3) = O(n^5)$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c)$:= all branching walks of type (T, φ)
- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- with $\leq c$ edges

[and with edge weights $\omega\colon E \to \mathbb{Z}_{\geq 0}$?]

Let $b_F(x, c) := |\mathcal{B}_F(x, c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x,c) = \begin{cases} 1 & \text{if } c = 0, \\ \displaystyle\sum_{t \in N(x) \setminus F} \; \sum_{c_1 + c_2 \leq c-1} b_F(t, c_1) \cdot b_F(x, c_2) & \text{otherwise.} \end{cases}$$

Runtime: $O(n^2 \cdot n^3) = O(n^5)$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x, c) :=$ all branching walks of type (T, φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- ~~with $\leq c$ edges~~ total edge weight $\leq c$

[and with edge weights $\omega \colon E \to \mathbb{Z}_{\geq 0}$?]

Let $b_F(x, c) := |\mathcal{B}_F(x, c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x, c) = \begin{cases} 1 & \text{if } c = 0, \\ \displaystyle\sum_{t \in N(x) \setminus F} \sum_{c_1 + c_2 \leq c - 1} b_F(t, c_1) \cdot b_F(x, c_2) & \text{otherwise.} \end{cases}$$

Runtime: $O(n^2 \cdot n^3) = O(n^5)$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees... still need φ.

Def. $\mathcal{B}_F(x,c) :=$ all branching walks of type (T,φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- ~~with $\leq c$ edges~~ total edge weight $\leq c$

[and with edge weights $\omega\colon E \to \mathbb{Z}_{\geq 0}$?]

Let $b_F(x,c) := |\mathcal{B}_F(x,c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x,c) = \begin{cases} 1 & \text{if } c = 0, \\ \displaystyle\sum_{t \in N(x)\setminus F} \ \sum_{c_1+c_2 \leq \cancel{c-1}\ c-\omega(xt)} b_F(t,c_1)\cdot b_F(x,c_2) & \text{otherwise.} \end{cases}$$

Runtime: $O(n^2 \cdot n^3) = O(n^5)$

Counting Branching Walks

$|\mathcal{T}_n|$ works for counting rooted ordered trees. . . still need φ.

Def. $\mathcal{B}_F(x,c) :=$ all branching walks of type (T,φ)

- contained in $G[V \setminus F]$
- starting at $x \in V \setminus F$
- ~~with $\leq c$ edges~~ total edge weight $\leq c$

[and with edge weights $\omega\colon E \to \mathbb{Z}_{\geq 0}$?]

Let $b_F(x,c) := |\mathcal{B}_F(x,c)|$.

Thm. For $F \subseteq K$ and $x \in V \setminus F$:

$$b_F(x,c) = \begin{cases} 1 & \text{if } c = 0, \\ \displaystyle\sum_{t \in N(x) \setminus F} \; \sum_{c_1 + c_2 \leq \cancel{c-1}\; c - \omega(xt)} b_F(t,c_1) \cdot b_F(x,c_2) & \text{otherwise.} \end{cases}$$

Runtime: $O(n^2 \cdot n^3) = O(n^5)$ – unweighted case

$O(nc \cdot nc^2) = O(n^2c^3)$ – weighted case

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?

IE-Formulation:

$s_0 \in K$

$\mathcal{U} =$ {branching walks with root s_0 and weight $\leq c$ in G}

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\},$

where $P_v =$ `"branching walk contains` v`"` $\}$

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that
- $K \subseteq V'$ and
- $|E'| \leq c$?

IE-Formulation:

$s_0 \in K$

$\mathcal{U} = \{$branching walks with root s_0 and weight $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\}$,
where $P_v =$ `"branching walk contains` v`"` $\}$

Recall: $N(\mathcal{P}) =$

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- $|E'| \leq c$?

IE-Formulation:

$s_0 \in K$

$\mathcal{U} = \{$branching walks with root s_0 and weight $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\},$

where $P_v =$ "`branching walk contains` v" $\}$

Recall: $N(\mathcal{P}) = \sum_{F \subseteq \mathcal{P}} (-1)^{|F|} N(\emptyset, F, \mathcal{P} \setminus F)$

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that
- $K \subseteq V'$ and
- $|E'| \leq c$?

IE-Formulation:

$s_0 \in K$

$\mathcal{U} = \{$branching walks with root s_0 and weight $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\}$,

where $P_v =$ `"branching walk contains` v`"` $\}$

Recall: $N(\mathcal{P}) = \sum_{F \subseteq \mathcal{P}} (-1)^{|F|} \underbrace{N(\emptyset, F, \mathcal{P} \setminus F)}_{\beta_F(s_0, c)}$

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that
- $K \subseteq V'$ and
- $|E'| \leq c$?

IE-Formulation:

$s_0 \in K$

$\mathcal{U} = \{\text{branching walks with root } s_0 \text{ and weight} \leq c \text{ in } G\}$

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\},$
where $P_v =$ "branching walk contains v" $\}$

Runtime:

Recall: $N(\mathcal{P}) = \sum_{F \subseteq \mathcal{P}} (-1)^{|F|} \underbrace{N(\emptyset, F, \mathcal{P} \setminus F)}_{\beta_F(s_0, c)}$

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c

Question: Does there exist a subtree (V', E') of G such that
- $K \subseteq V'$ and
- $|E'| \leq c$?

IE-Formulation:

$s_0 \in K$

$\mathcal{U} = \{$branching walks with root s_0 and weight $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\},$
where $P_v =$ "branching walk contains v" $\}$

Runtime: $O(2^k \cdot \text{poly}(n))$ unweighted

Recall: $N(\mathcal{P}) = \sum_{F \subseteq \mathcal{P}} (-1)^{|F|} \underbrace{N(\emptyset, F, \mathcal{P} \setminus F)}_{\beta_F(s_0, c)}$

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c and edge weights $\omega\colon E \to \mathbb{Z}_{\geq 0}$,

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- ~~$|E'| \leq c$?~~ $\omega(E') \leq c$

IE-Formulation:

$\mathcal{U} =$ {branching walks with root s_0 and weight $\leq c$ in G}   ($\swarrow$ $s_0 \in K$)

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\}$,
where $P_v =$ "branching walk contains v" }

Runtime: $O(2^k \cdot \text{poly}(n))$ unweighted

Recall: $N(\mathcal{P}) = \sum_{F \subseteq \mathcal{P}} (-1)^{|F|} \underbrace{N(\emptyset, F, \mathcal{P} \setminus F)}_{\beta_F(s_0, c)}$

Steiner Tree: Summary

Given: Graph $G = (V, E)$, terminals $K \subseteq V$, number c
and edge weights $\omega \colon E \to \mathbb{Z}_{\geq 0}$,

Question: Does there exist a subtree (V', E') of G such that

- $K \subseteq V'$ and
- ~~$|E'| \leq c$?~~ $\omega(E') \leq c$

IE-Formulation:

$s_0 \in K$

$\mathcal{U} = \{$branching walks with root s_0 and weight $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in K \setminus \{s_0\},$
where $P_v =$ "branching walk contains v" $\}$

Runtime: $O(2^k \cdot \text{poly}(n))$ unweighted
$O(2^k \cdot \text{poly}(n, c))$ weighted

Recall: $N(\mathcal{P}) = \sum_{F \subseteq \mathcal{P}} (-1)^{|F|} \underbrace{N(\emptyset, F, \mathcal{P} \setminus F)}_{\beta_F(s_0, c)}$

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

IE-Formulation:

$\mathcal{U} =$

$\mathcal{P} =$

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ and of degree $\leq c$ in $G\}$

$\mathcal{P} =$

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \le c \le n$

Question: $\exists$ spanning tree of G of maximum degree $\le c$?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n - 1$ and of degree $\le c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in V,$

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n - 1$ and of degree $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ `"branching walk contains` v`"` $\}$

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ and of degree $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ `"branching walk contains` v`"` $\}$

Easier Problem:

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n - 1$ and of degree $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ `"branching walk contains` v`"` $\}$

Easier Problem: (find and solve it yourself!)

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n - 1$ and of degree $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ "`branching walk contains` v" $\}$

Easier Problem: (find and solve it yourself!)

Runtime: $O^*(2^n)$

Degree-Constrained Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G of maximum degree $\leq c$?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ and of degree $\leq c$ in $G\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ "`branching walk contains` v" $\}$

Easier Problem: (find and solve it yourself!)

Runtime: $O^*(2^n)$, improving over $O^*(5.92^n)$ [Amini et al., ICALP'09]

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} =$

$\mathcal{P} =$

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ with $\geq c$ internal vtc.$\}$

$\mathcal{P} =$

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ with $\geq c$ internal vtc.$\}$

$\mathcal{P} = \{P_v \mid v \in V,$

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ with $\geq c$ internal vtc.$\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ "`branching walk contains` v" $\}$

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ with $\geq c$ internal vtc.$\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ "`branching walk contains` v" $\}$

Easier Problem:

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ with $\geq c$ internal vtc.$\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ "`branching walk contains` v" $\}$

Easier Problem: (find and solve it yourself!)

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ with $\geq c$ internal vtc.$\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ "`branching walk contains` v" $\}$

Easier Problem: (find and solve it yourself!)

Runtime: $O^*(2^n)$

Maximum Internal Spanning Tree

Given: Graph $G = (V, E)$, number $1 \leq c \leq n$

Question: $\exists$ spanning tree of G with $\geq c$ internal vertices?

IE-Formulation:

$\mathcal{U} = \{$branching walks of length $n-1$ with $\geq c$ internal vtc.$\}$

$\mathcal{P} = \{P_v \mid v \in V,$

where $P_v =$ "`branching walk contains` v" $\}$

Easier Problem: (find and solve it yourself!)

Runtime: $O^*(2^n)$, improving over $O^*(3^n)$

[Fernau et al., WG'09]

Partitioning Numbers

PARTITION

Given: Set S of integers.

Question: $\exists$ partition of S into two sets with the same sum?

Partitioning Numbers

PARTITION

Given: Set S of integers.

Question: $\exists$ partition of S into two sets with the same sum?

SUBSETSUM

Given: Set S of integers and an integer t.

Question: $\exists$ subset of S that sums to t?

Partitioning Numbers

PARTITION

Given: Set S of integers.

Question: $\exists$ partition of S into two sets with the same sum?

SUBSETSUM

Given: Set S of integers and an integer t.

Question: $\exists$ subset of S that sums to t?

3-PARTITION

Given: Set S of integers

Question: $\exists$ partition of S into 3-tuples with the same sum?

Partitioning Numbers

PARTITION

Given: Set S of integers.

Question: $\exists$ partition of S into two sets with the same sum?

SUBSETSUM

Given: Set S of integers and an integer t.

Question: $\exists$ subset of S that sums to t?

3-PARTITION

Given: Set S of integers

Question: $\exists$ partition of S into 3-tuples with the same sum?

Subset Sum – A Question

Thm. SubSetSum is *weakly* NP-hard.

Subset Sum – A Question

Thm. SubSetSum is *weakly* NP-hard.

Obs. Standard DP needs $O^*(n \cdot t)$ time and space

Subset Sum – A Question

Thm. SubSetSum is *weakly* NP-hard.

Obs. Standard DP needs $O^*(n \cdot t)$ time and space

Question: What is possible without depending on $\sum S$?

Subset Sum – A Question

Thm. SubSetSum is *weakly* NP-hard.

Obs. Standard DP needs $O^*(n \cdot t)$ time and space

Question: What is possible without depending on $\sum S$?

Obs. For S partitioned into $S_1 \mathbin{\dot\cup} S_2$, note:

Subset Sum – A Question

Thm. $\textsc{SubSetSum}$ is *weakly* NP-hard.

Obs. Standard DP needs $O^*(n \cdot t)$ time and space

Question: What is possible without depending on $\sum S$?

Obs. For S partitioned into $S_1 \,\dot\cup\, S_2$, note:

$$S \text{ has a subset of sum } t$$

$$\Updownarrow$$

Subset Sum – A Question

Thm. SubSetSum is *weakly* NP-hard.

Obs. Standard DP needs $O^*(n \cdot t)$ time and space

Question: What is possible without depending on $\sum S$?

Obs. For S partitioned into $S_1 \cup\!\!\!\cdot\; S_2$, note:

$$S \text{ has a subset of sum } t$$

$$\Updownarrow$$

$$\exists t_1, t_2 \text{ such that } t_1 + t_2 = t \text{ and } S_1 \text{ has a subset of sum } t_1 \text{ and } S_2 \text{ has a subset of sum } t_2$$

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \dot\cup S_2$:
$\sum_1$:= all possible subset sums of S_1
$\sum_2$:= all possible subset sums of S_2

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \,\dot\cup\, S_2$:
$\sum_1 :=$ all possible subset sums of S_1
$\sum_2 :=$ all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

Test: $\exists t_1 \in \sum_1,\ t_2 \in \sum_2$ with $t_1 + t_2 = t$

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \dot\cup S_2$:
$\sum_1$:= all possible subset sums of S_1
$\sum_2$:= all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

Test: $\exists t_1 \in \sum_1, t_2 \in \sum_2$ with $t_1 + t_2 = t$

Runtime:

Subset Sum – A Solution

$$S = \{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots\}$$

sum t_1 sum t_2

Def. For S partitioned into $S_1 \cupdot S_2$:
$\sum_1 :=$ all possible subset sums of S_1
$\sum_2 :=$ all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

Test: $\exists t_1 \in \sum_1$, $t_2 \in \sum_2$ with $t_1 + t_2 = t$

Runtime: ... $\sum_1 \times \sum_2$ candidates to check?

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \cup S_2$:
$\sum_1 :=$ all possible subset sums of S_1
$\sum_2 :=$ all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

sort $\sum_1$ and $\sum_2$

Test: $\exists t_1 \in \sum_1,\ t_2 \in \sum_2$ with $t_1 + t_2 = t$

Runtime:

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \dot\cup S_2$:
$\sum_1$:= all possible subset sums of S_1
$\sum_2$:= all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

sort $\sum_1$ and $\sum_2$ $\leftarrow$ How much time does this take?

Test: $\exists t_1 \in \sum_1,\ t_2 \in \sum_2$ with $t_1 + t_2 = t$

Runtime:

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \cup S_2$:
$\sum_1$:= all possible subset sums of S_1
$\sum_2$:= all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

sort $\sum_1$ and $\sum_2$

Test: $\exists t_1 \in \sum_1, t_2 \in \sum_2$ with $t_1 + t_2 = t$

Binary seach for each t_1!

Runtime:

Subset Sum – A Solution

$$S = \{\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_1},\underbrace{\ldots,\ldots,\ldots,\ldots,\ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \cupdot S_2$:
$\sum_1 :=$ all possible subset sums of S_1
$\sum_2 :=$ all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

sort $\sum_1$ and $\sum_2$

Binary seach for each t_1!

Test: $\exists t_1 \in \sum_1, t_2 \in \sum_2$ with $t_1 + t_2 = t$

Runtime: $O(\sqrt{2}^n \cdot \log_2(\sqrt{2}^n))$

Subset Sum – A Solution

$$S = \{\ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots\}$$

sum t_1 sum t_2

Def. For S partitioned into $S_1 \cup\!\!\!\cdot\; S_2$:
$\sum_1 :=$ all possible subset sums of S_1
$\sum_2 :=$ all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

sort $\sum_1$ and $\sum_2$

Binary seach for each t_1!

Test: $\exists t_1 \in \sum_1, t_2 \in \sum_2$ with $t_1 + t_2 = t$

Runtime: $O(\sqrt{2}^n \cdot \log_2(\sqrt{2}^n)) \subseteq O(\sqrt{2}^n \cdot n)$

Subset Sum – A Solution

$$S = \{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots,\ldots\}$$

sum t_1 sum t_2

Def. For S partitioned into $S_1 \cup S_2$:
$\sum_1 :=$ all possible subset sums of S_1
$\sum_2 :=$ all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

sort $\sum_1$ and $\sum_2$

Binary seach for each t_1!

Test: $\exists t_1 \in \sum_1, t_2 \in \sum_2$ with $t_1 + t_2 = t$

Runtime: $O(\sqrt{2}^n \cdot \log_2(\sqrt{2}^n)) \subseteq O(\sqrt{2}^n \cdot n) \subseteq O^*(\sqrt{2}^n)$

Subset Sum – A Solution

$$S = \{\underbrace{\ldots, \ldots, \ldots, \ldots, \ldots, \ldots}_{\text{sum } t_1}, \underbrace{\ldots, \ldots, \ldots, \ldots, \ldots}_{\text{sum } t_2}\}$$

Def. For S partitioned into $S_1 \,\dot\cup\, S_2$:
$\sum_1$:= all possible subset sums of S_1
$\sum_2$:= all possible subset sums of S_2

Algo: Compute $\sum_1$ and $\sum_2$.

sort $\sum_1$ and $\sum_2$

Test: $\exists t_1 \in \sum_1, t_2 \in \sum_2$ with $t_1 + t_2 = t$

Binary seach for each t_1!

Runtime: $O(\sqrt{2}^n \cdot \log_2(\sqrt{2}^n)) \subseteq O(\sqrt{2}^n \cdot n) \subseteq O^*(\sqrt{2}^n)$

Can we get rid of this factor?