Approximation Algorithms

Lecture 10:
Minimum-Degree Spanning Tree
via Local Search

Part I:
Minimum-Degree Spanning Tree

Alexander Wolff

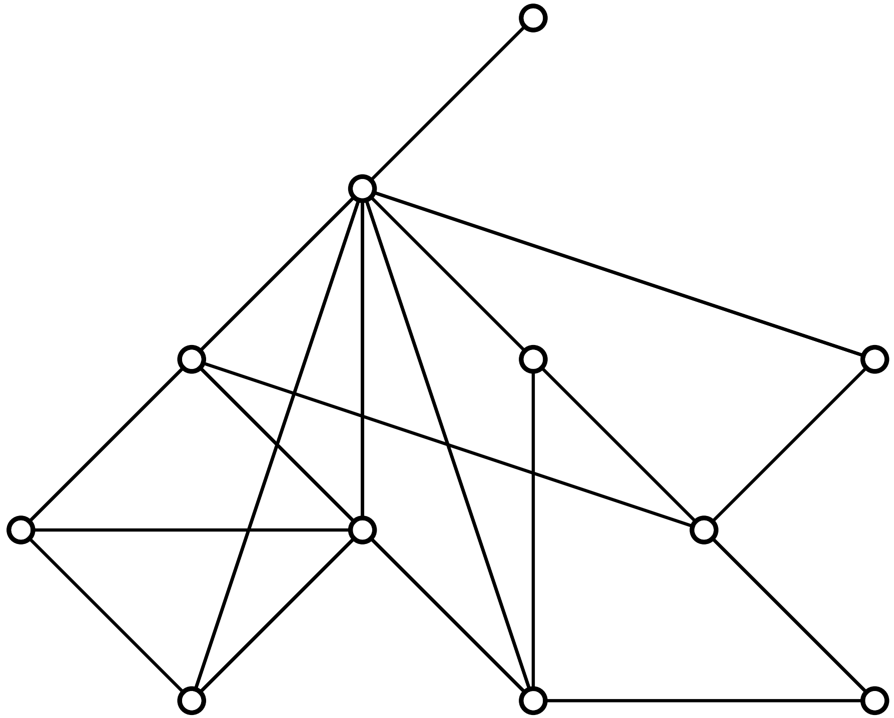
Winter term 2025

Minimum-Degree Spanning Tree

Given: A connected graph G.

Minimum-Degree Spanning Tree

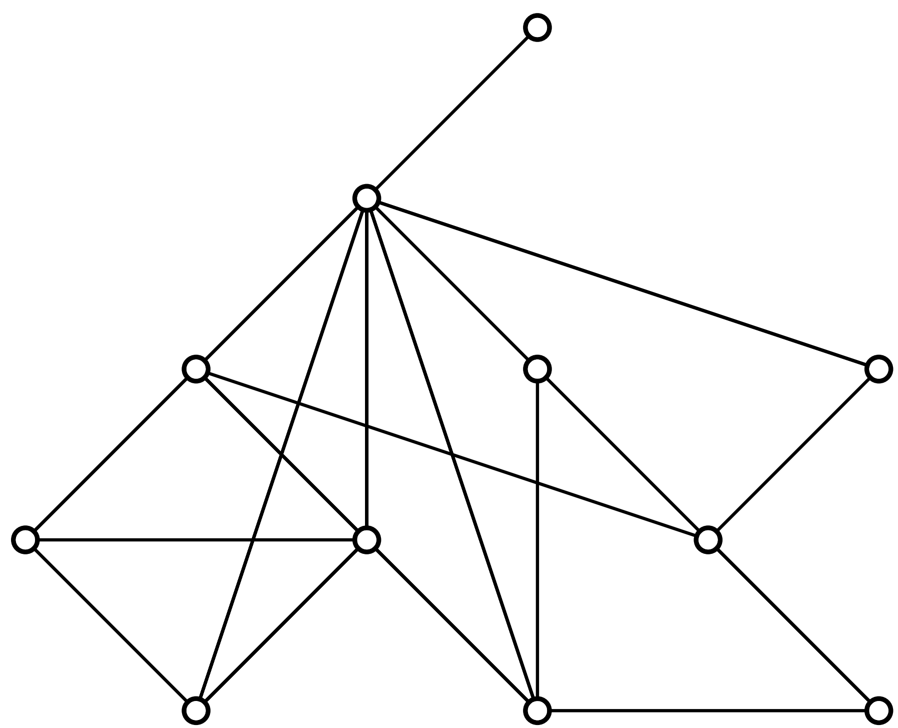
Given: A connected graph G.



Minimum-Degree Spanning Tree

Given: A connected graph G.

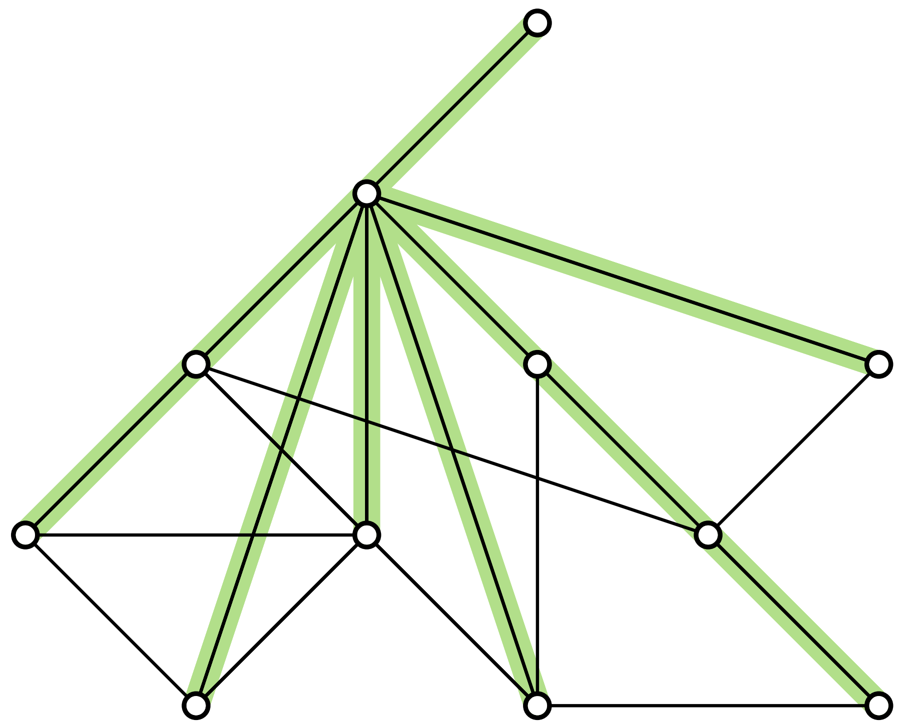
Task: Find a spanning tree T that has the smallest maximum degree $\Delta(T)$ among all spanning trees of G.



Minimum-Degree Spanning Tree

Given: A connected graph G.

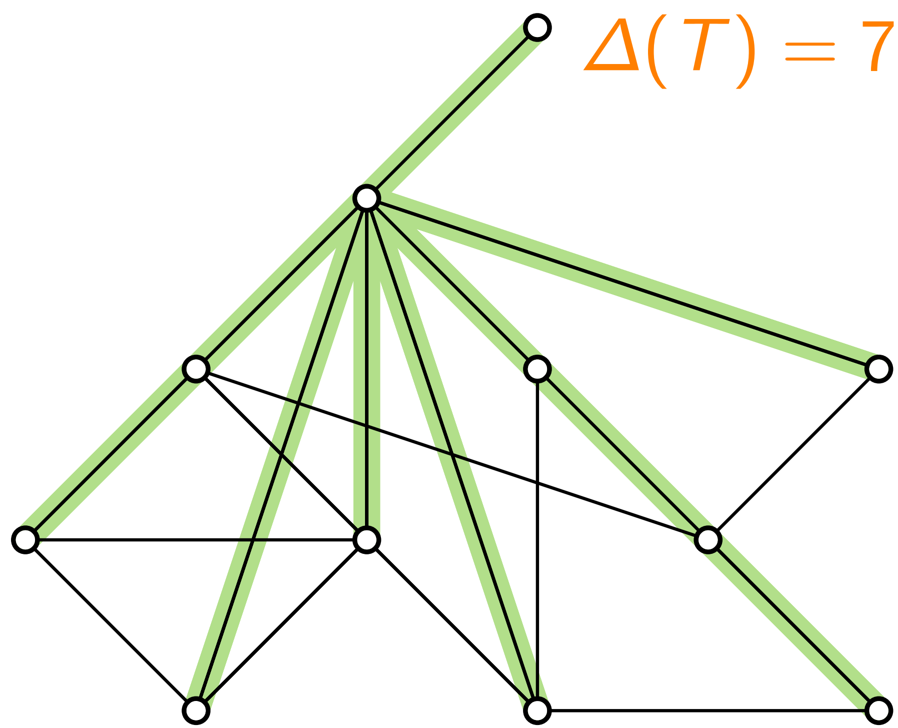
Task: Find a spanning tree T that has the smallest maximum degree $\Delta(T)$ among all spanning trees of G.



Minimum-Degree Spanning Tree

Given: A connected graph G.

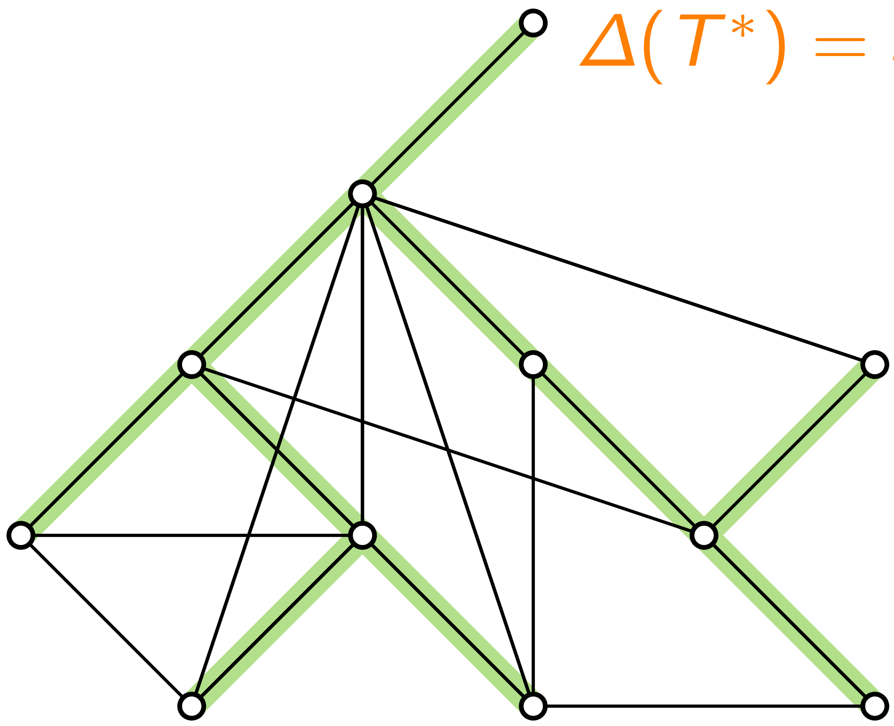
Task: Find a spanning tree T that has the smallest maximum degree $\Delta(T)$ among all spanning trees of G.



Minimum-Degree Spanning Tree

Given: A connected graph G.

Task: Find a spanning tree T that has the smallest maximum degree $\Delta(T)$ among all spanning trees of G.

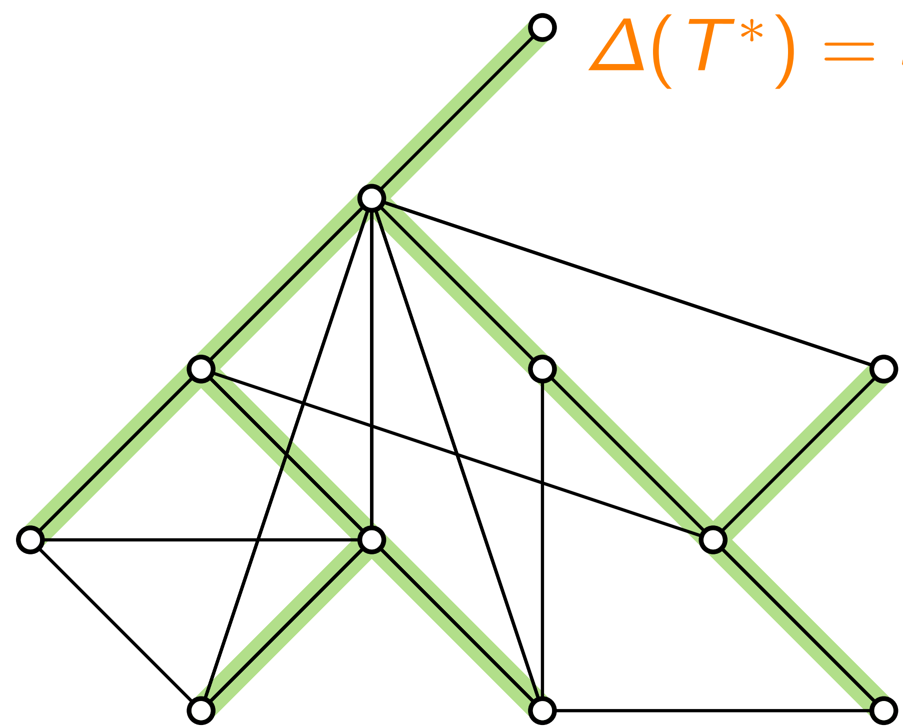


Minimum-Degree Spanning Tree

Given: A connected graph G.

Task: Find a spanning tree T that has the smallest maximum degree $\Delta(T)$ among all spanning trees of G.

NP-hard.



$\Delta(T^*) = 3$

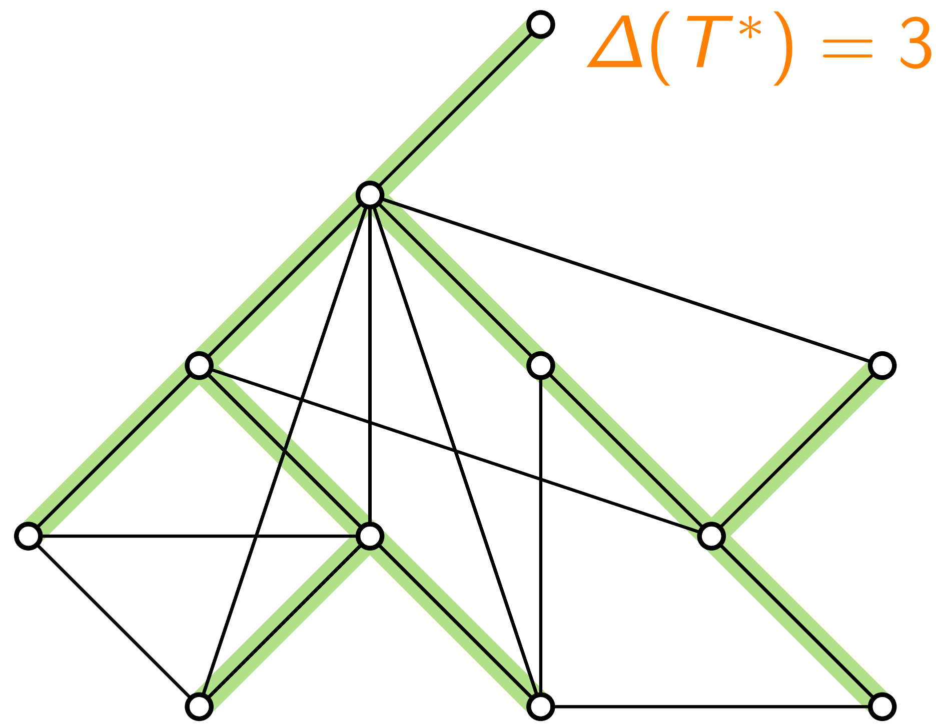
Minimum-Degree Spanning Tree

Given: A connected graph G.

Task: Find a spanning tree T that has the smallest maximum degree $\Delta(T)$ among all spanning trees of G.

NP-hard. 

Why?



Minimum-Degree Spanning Tree

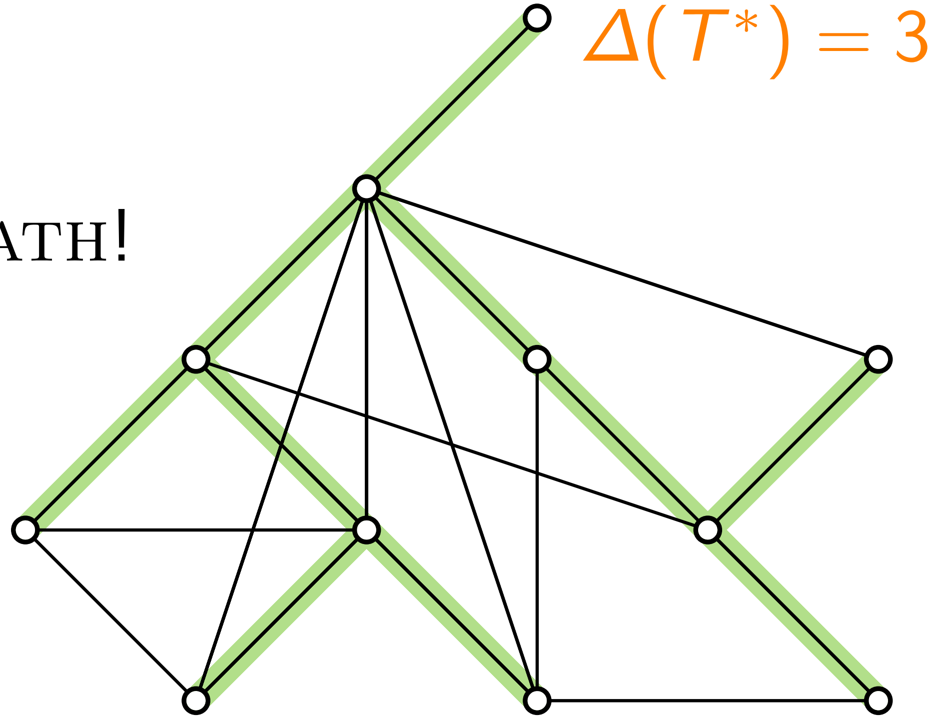
Given: A connected graph G.

Task: Find a spanning tree T that has the smallest maximum degree $\Delta(T)$ among all spanning trees of G.

NP-hard. 

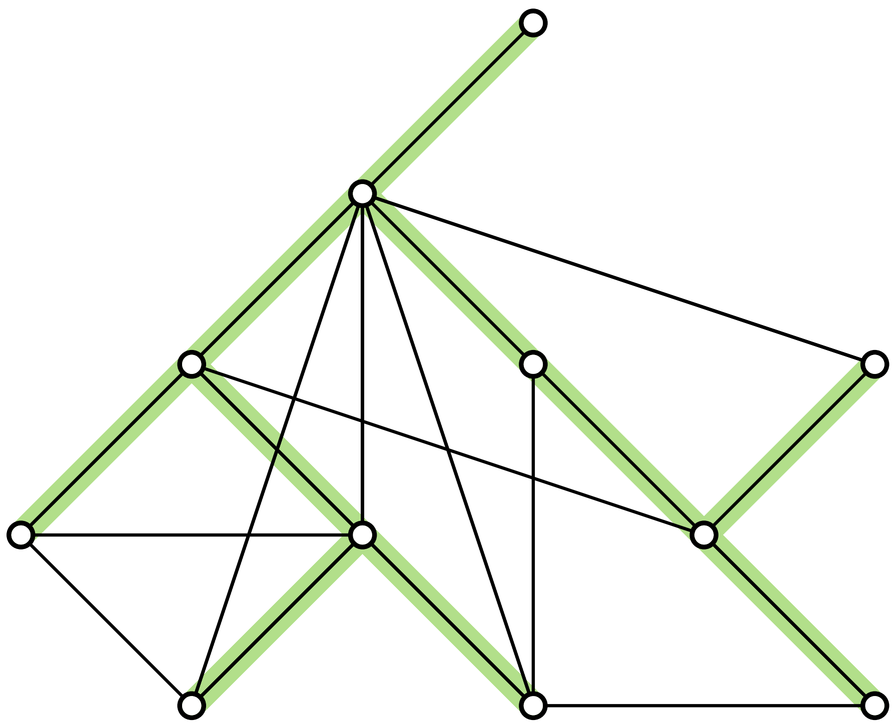
Why?

Special case of Hamiltonian Path!



Warm-up

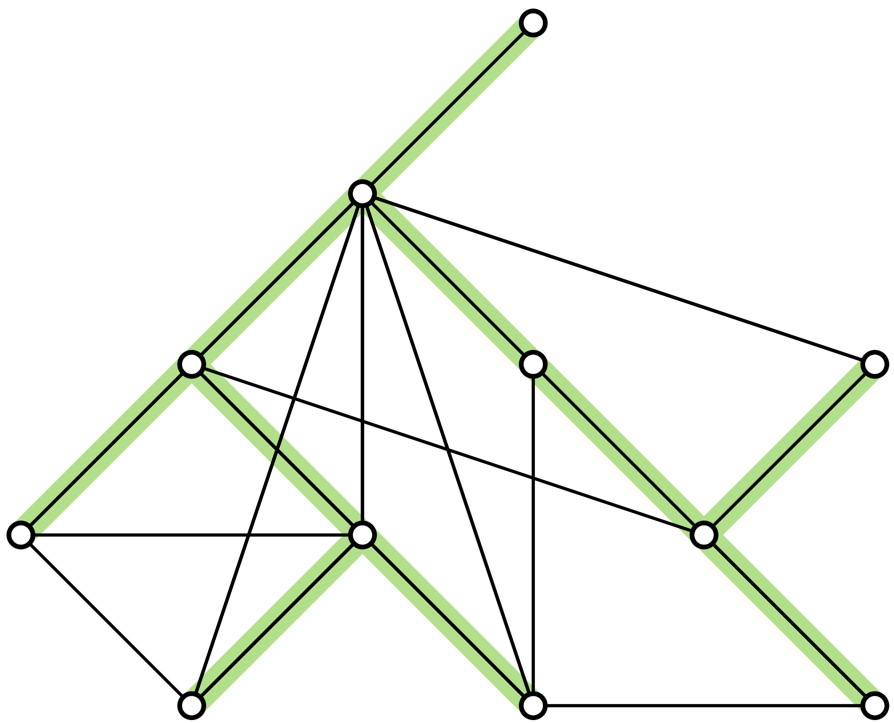
Obs. 1. A spanning tree T has...



Warm-up

Obs. 1. A spanning tree T has. . .

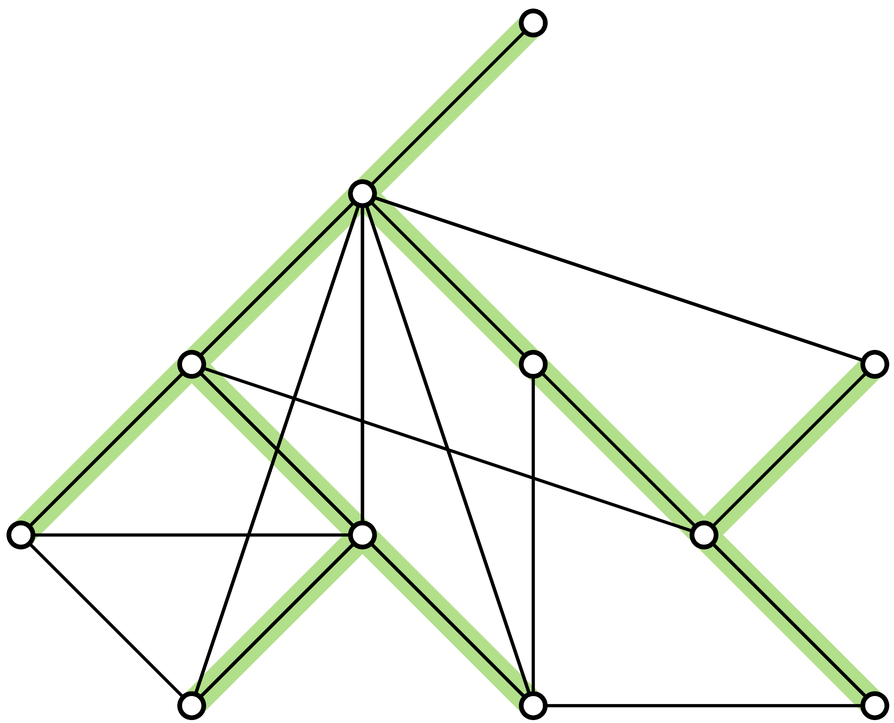
- n vertices and ? edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) =$?
- average degree $<$?



Warm-up

Obs. 1. A spanning tree T has...

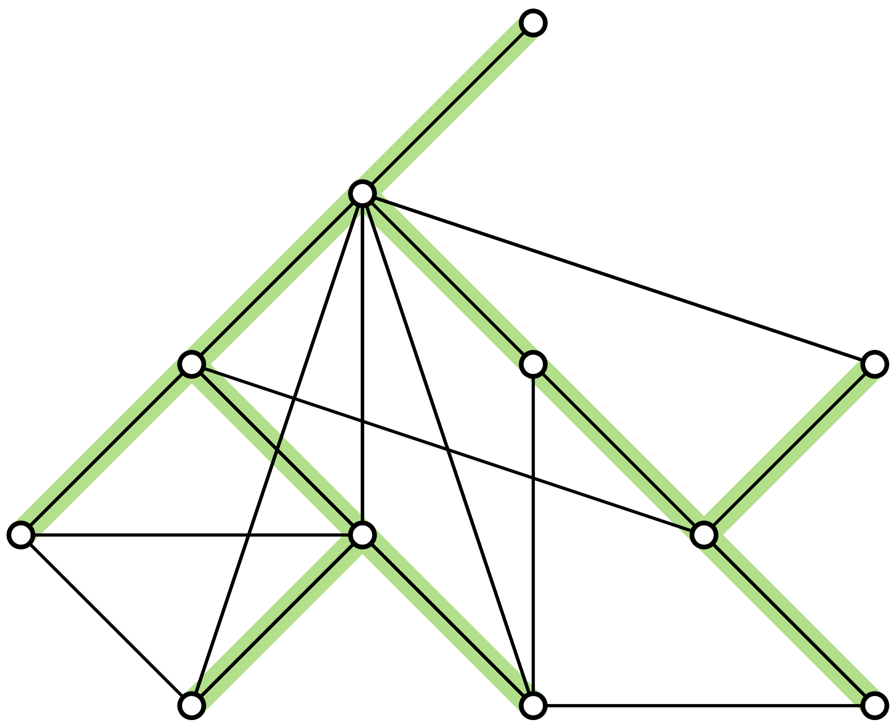
- n vertices and $n-1$ edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) =$?
- average degree $<$?



Warm-up

Obs. 1. A spanning tree T has...

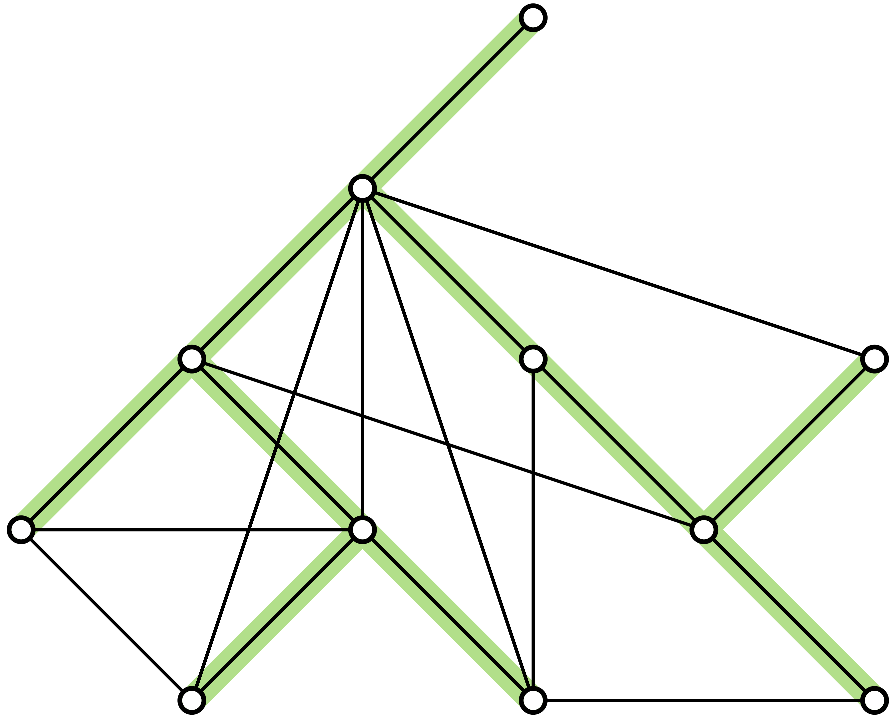
- n vertices and $n-1$ edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) = 2n-2$,
- average degree $<$?



Warm-up

Obs. 1. A spanning tree T has...

- n vertices and $n-1$ edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) = 2n-2$,
- average degree < 2.

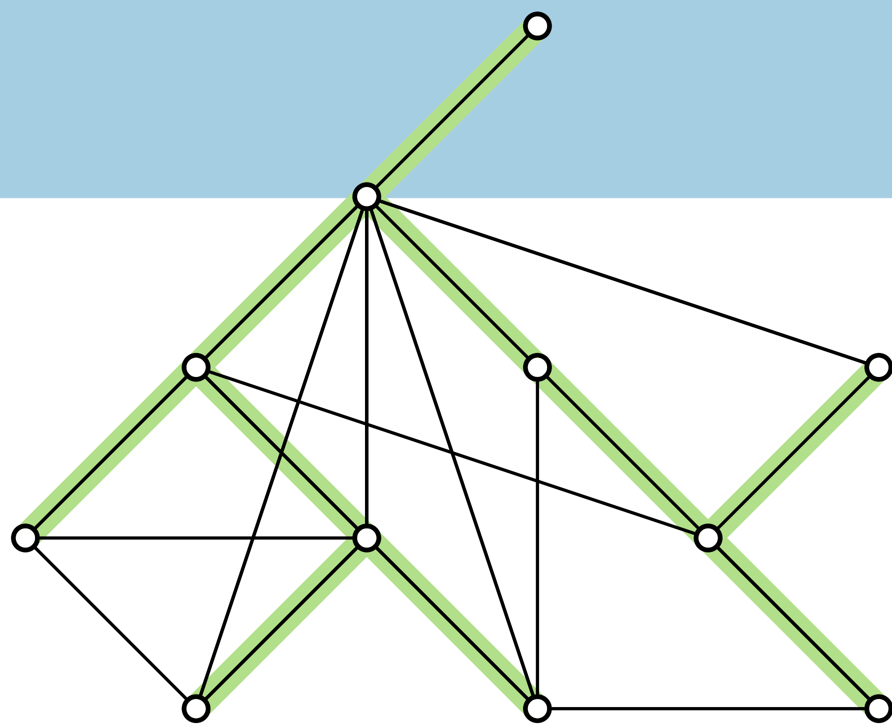


Warm-up

Obs. 1. A spanning tree T has...

- n vertices and $n-1$ edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) = 2n - 2$,
- average degree < 2.

Obs. 2. Let $V' \subseteq V(G)$.
Then $\Delta(G) \geq$?

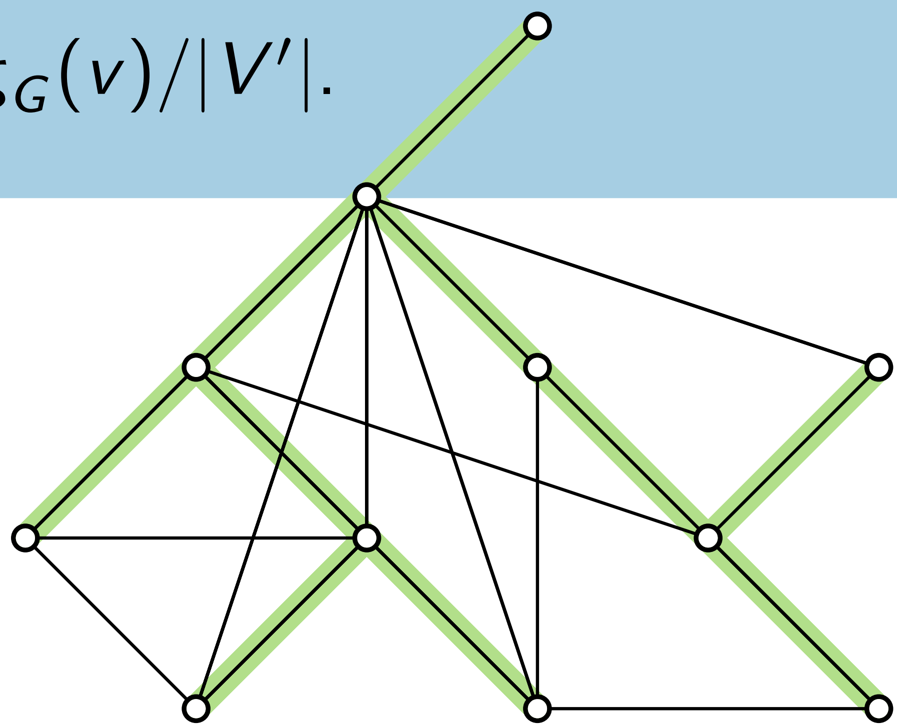


Warm-up

Obs. 1. A spanning tree T has...

- n vertices and $n-1$ edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) = 2n-2$,
- average degree < 2.

Obs. 2. Let $V' \subseteq V(G)$.
Then $\Delta(G) \geq \sum_{v \in V'} \deg_G(v)/|V'|$.



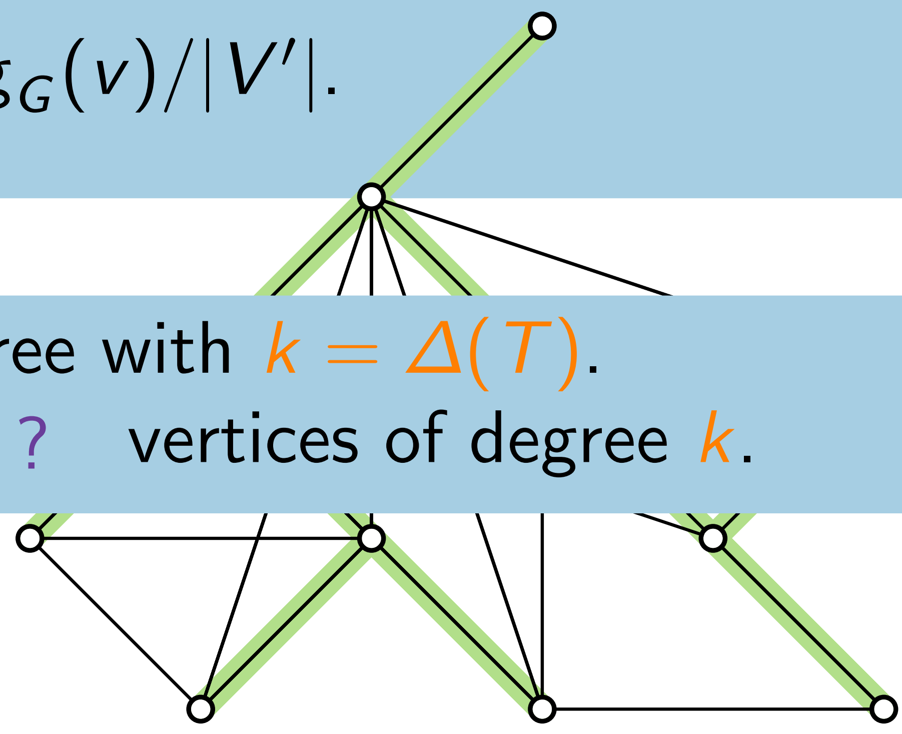
Warm-up

Obs. 1. A spanning tree T has...

- n vertices and $n-1$ edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) = 2n-2$,
- average degree < 2.

Obs. 2. Let $V' \subseteq V(G)$.
Then $\Delta(G) \geq \sum_{v \in V'} \deg_G(v)/|V'|$.

Obs. 3. Let T be a spanning tree with $k = \Delta(T)$.
Then T has at most ? vertices of degree k.



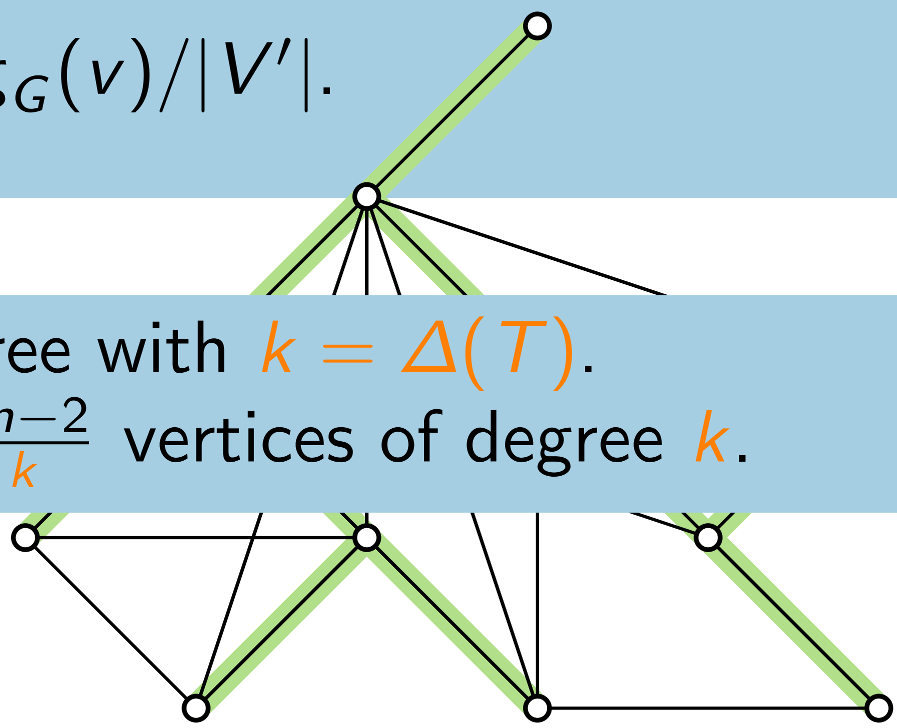
Warm-up

Obs. 1. A spanning tree T has...

- n vertices and $n-1$ edges,
- sum of degrees $\sum_{v \in V(G)} \deg_T(v) = 2n-2$,
- average degree < 2.

Obs. 2. Let $V' \subseteq V(G)$.
Then $\Delta(G) \geq \sum_{v \in V'} \deg_G(v)/|V'|$.

Obs. 3. Let T be a spanning tree with $k = \Delta(T)$.
Then T has at most $\frac{2n-2}{k}$ vertices of degree k.

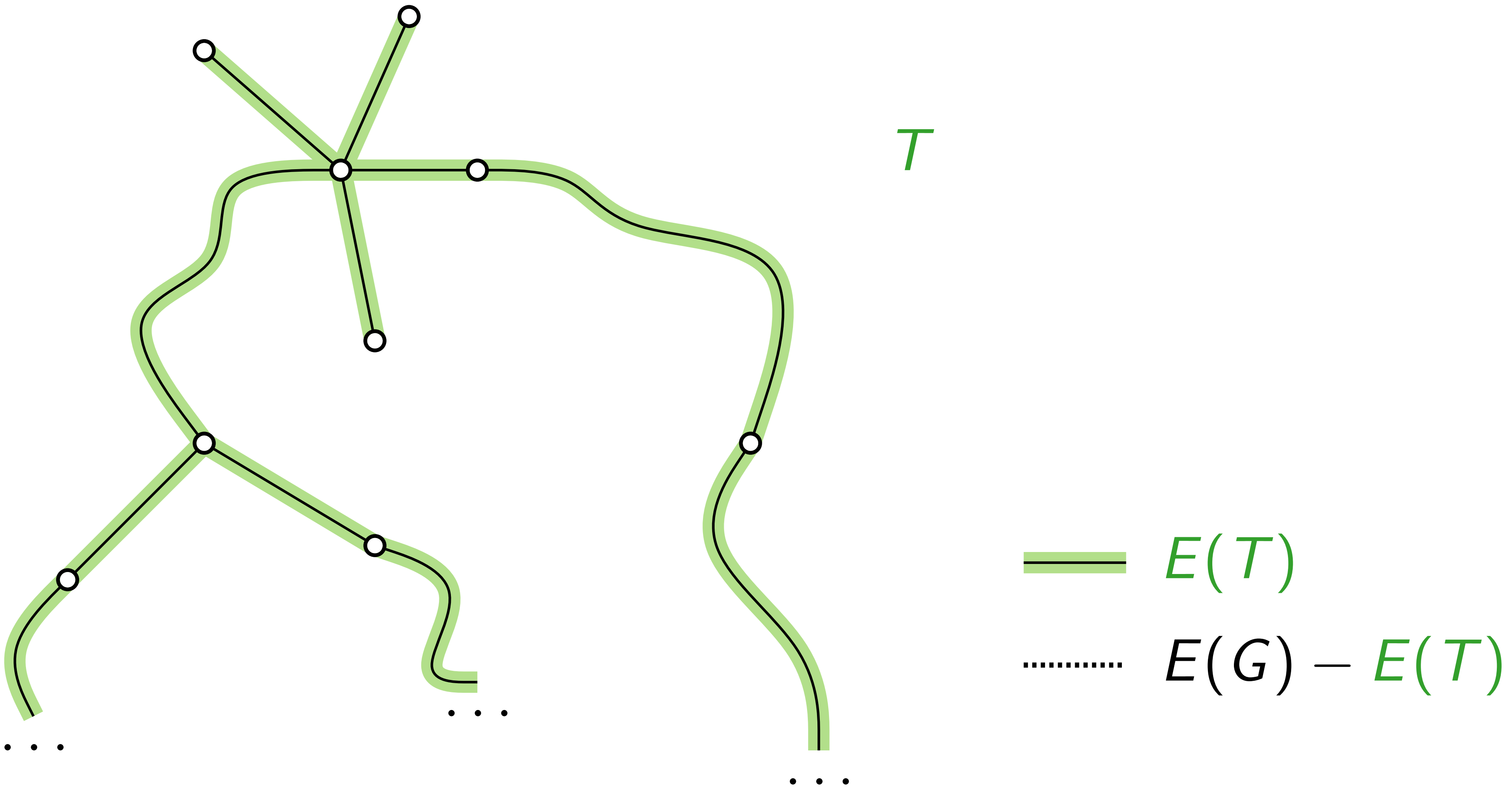


Approximation Algorithms

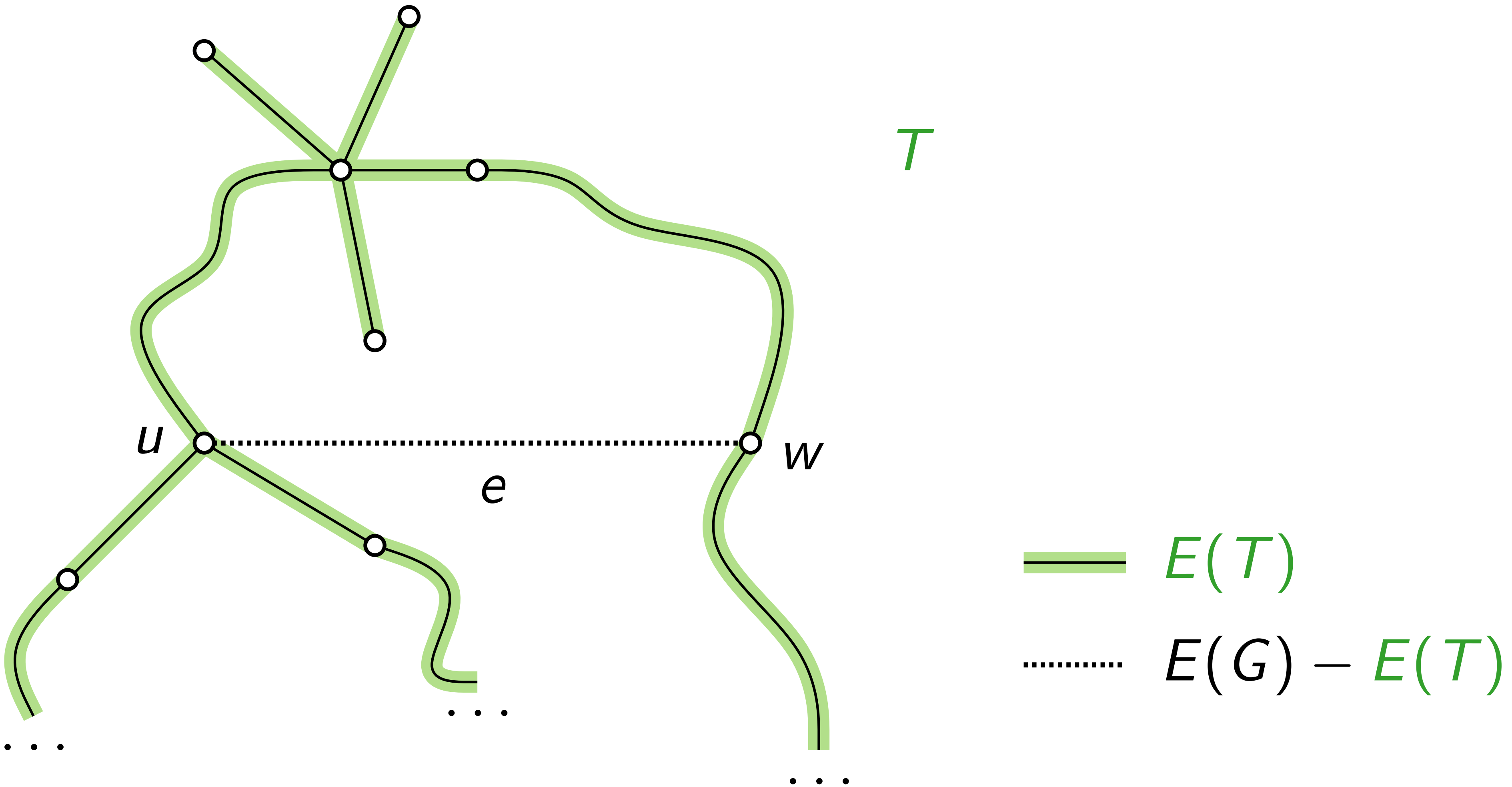
Lecture 10:
Minimum-Degree Spanning Tree
via Local Search

Part II:
Edge Flips and Local Search

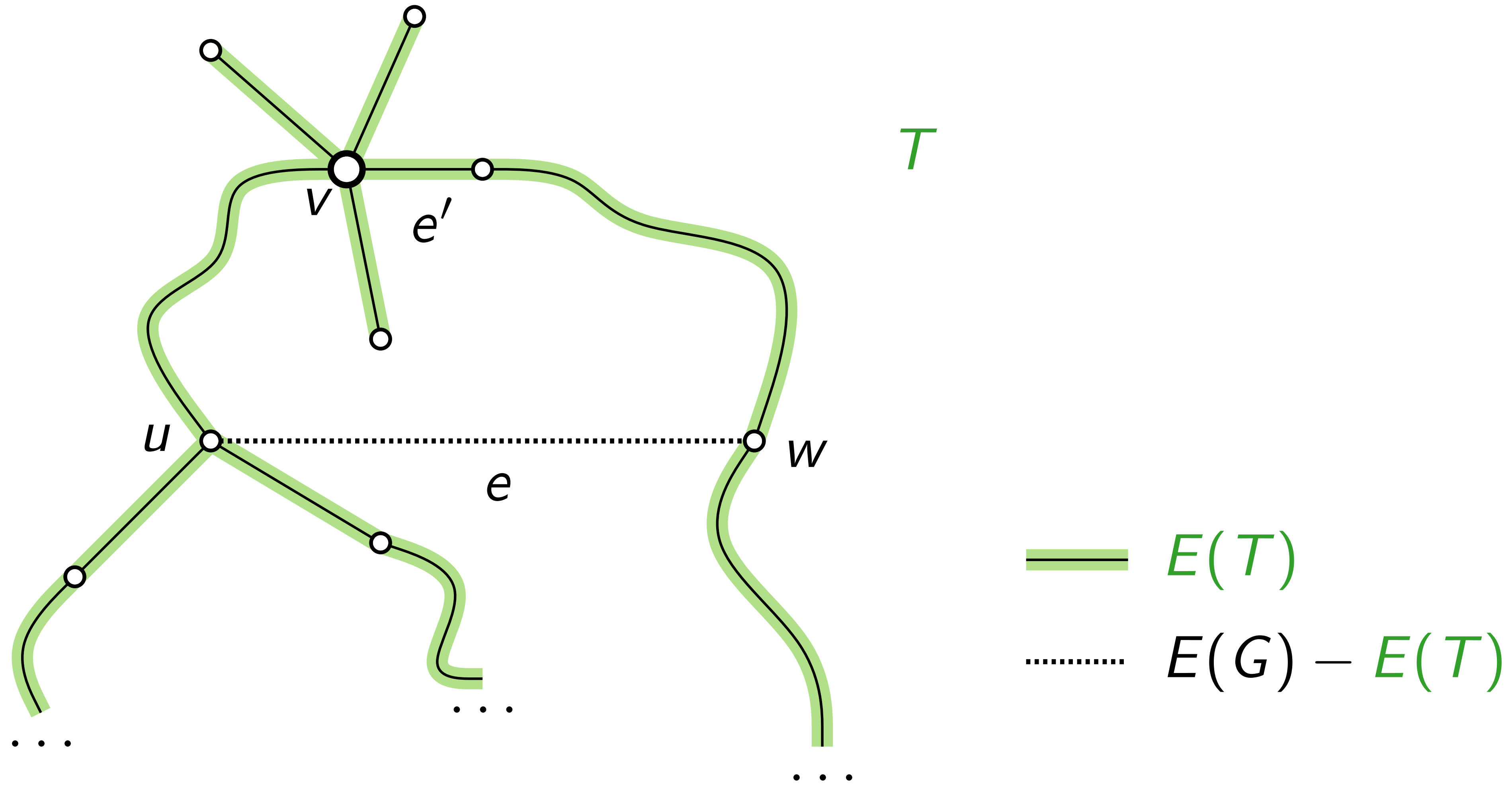
Edge Flips



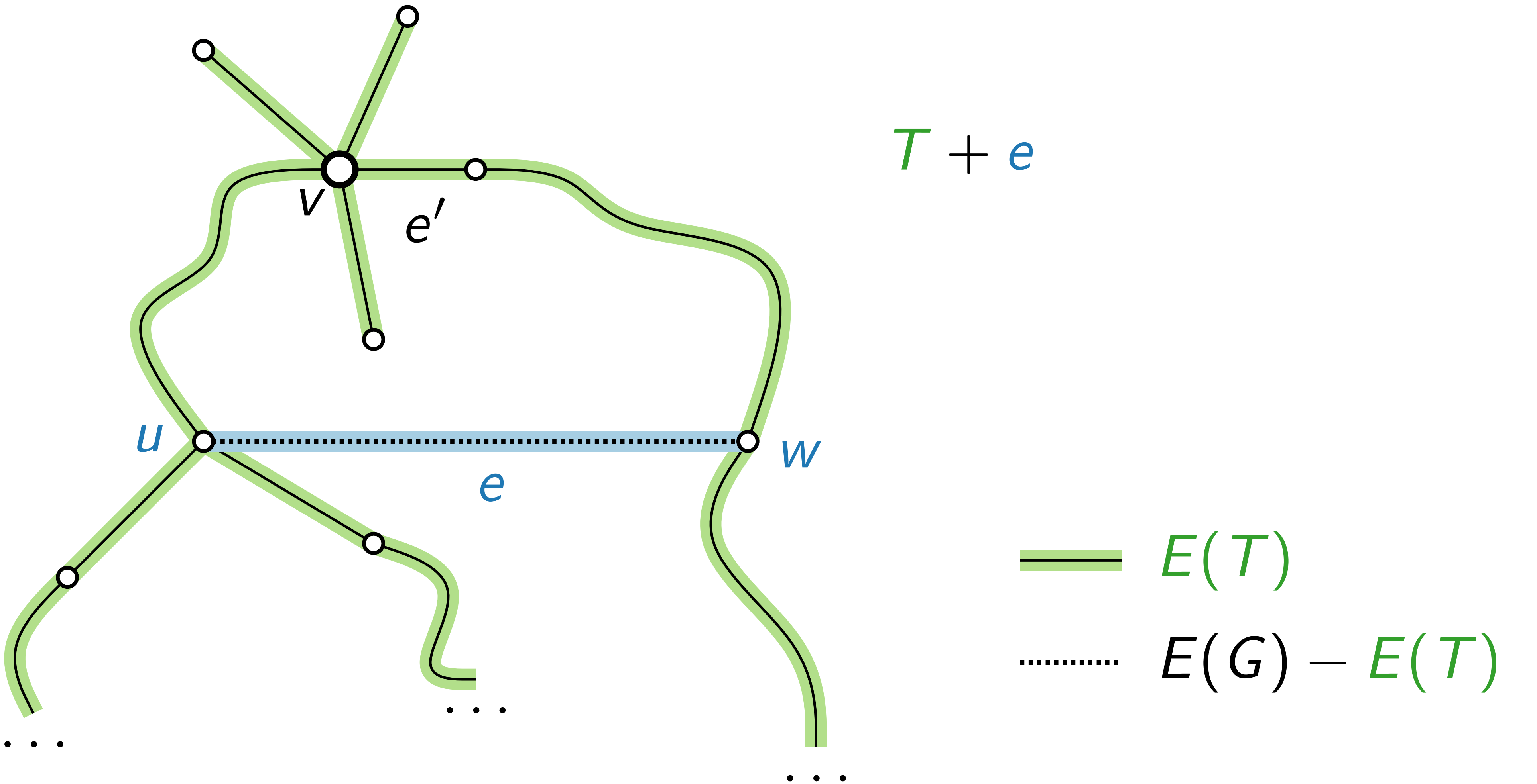
Edge Flips



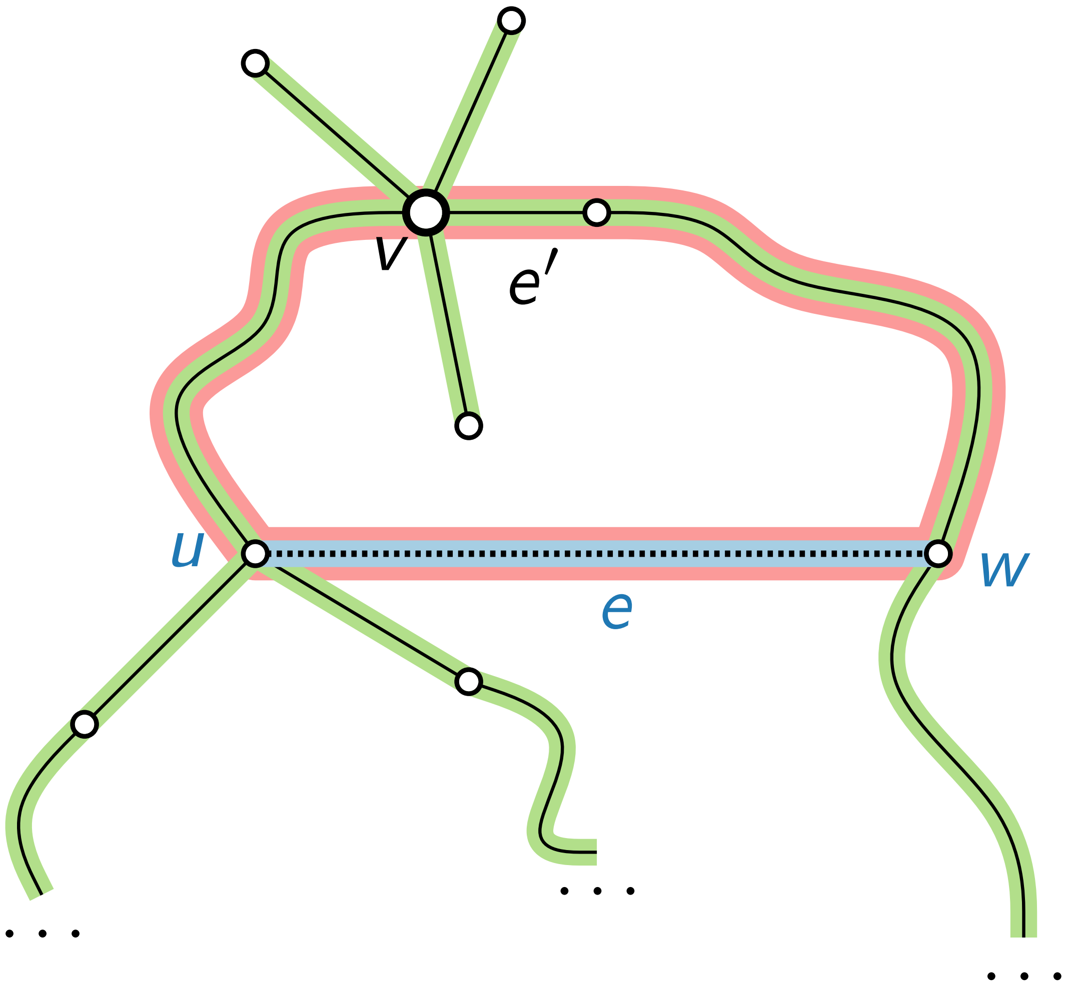
Edge Flips



Edge Flips



Edge Flips



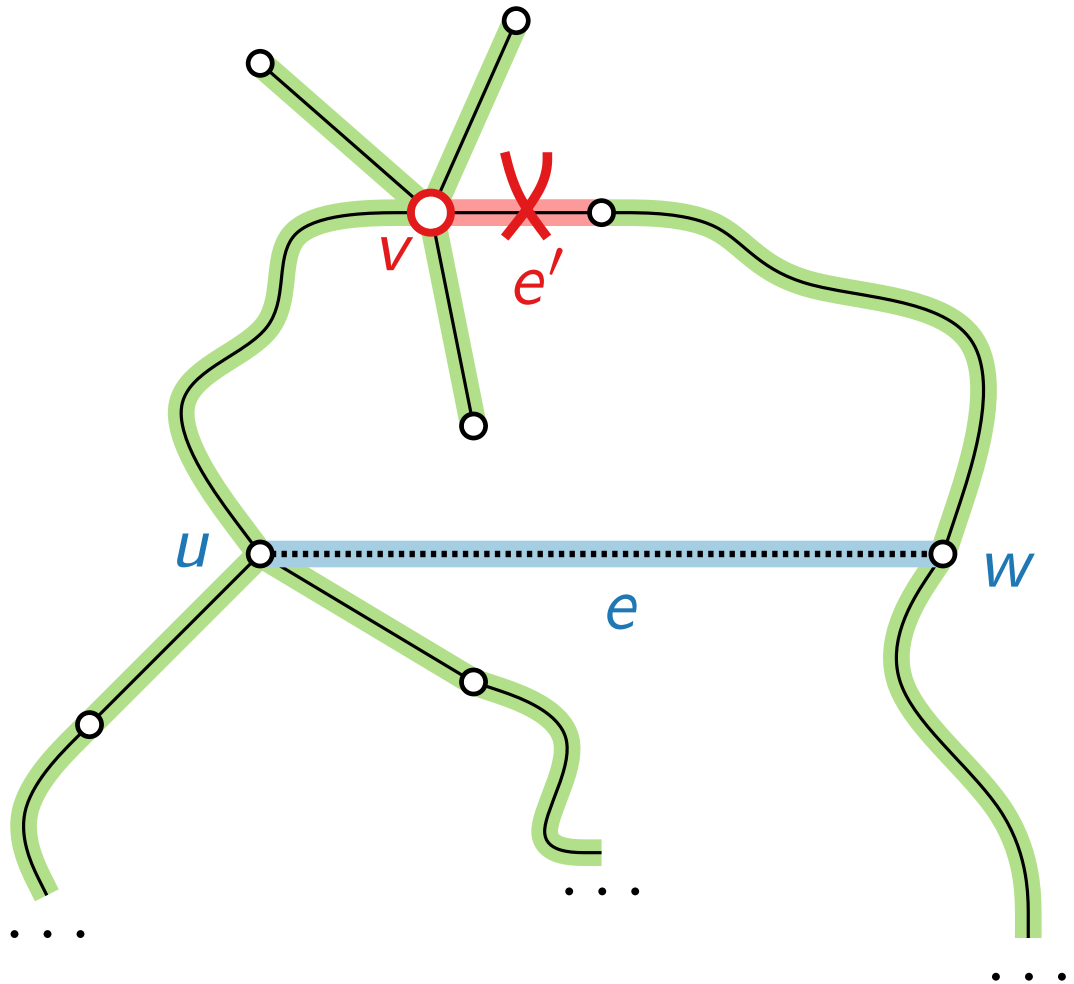
$T + e$

contains a cycle!

$E(T)$

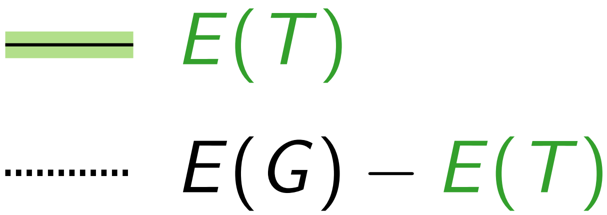
$E(G) - E(T)$

Edge Flips



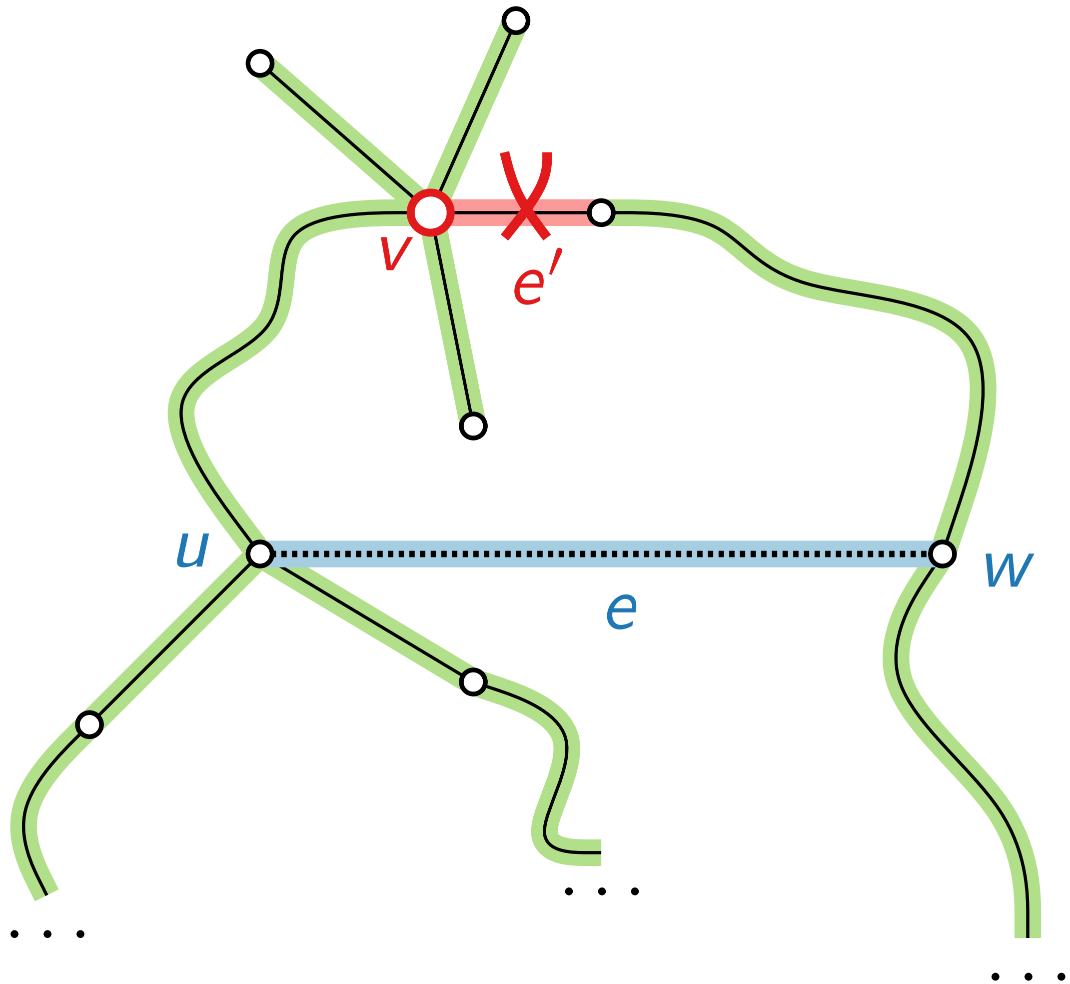
$T + e - e'$

is a new spanning tree.

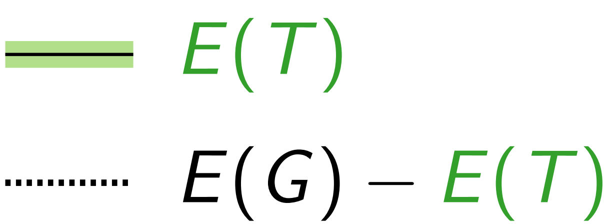


Edge Flips

Def. An **improving flip** in T for a vertex v and an edge $uw \in E(G) \setminus E(T)$ is a flip with $\deg_T(v) >$

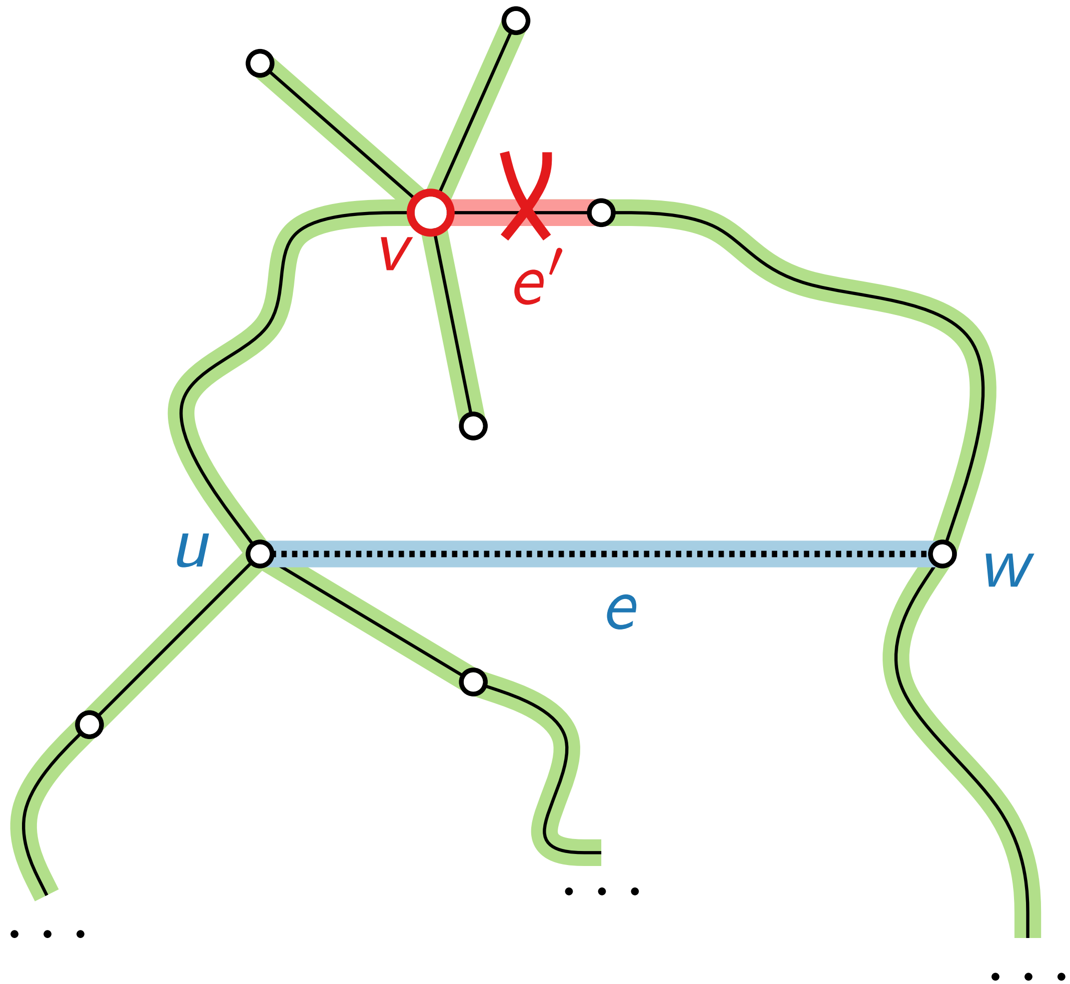


$T + e - e'$ is a new spanning tree.



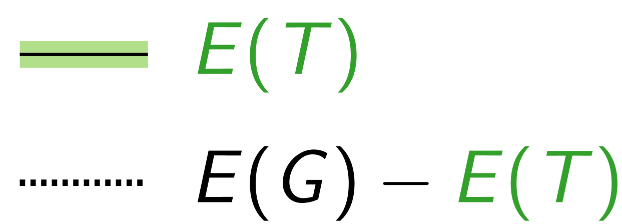
Edge Flips

Def. An **improving flip** in T for a vertex v and an edge $uw \in E(G) \setminus E(T)$ is a flip with $\deg_T(v) > \max\{\deg_T(u), \deg_T(w)\} + 1$.



$T + e - e'$

is a new spanning tree.



Local Search

MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T

Local Search

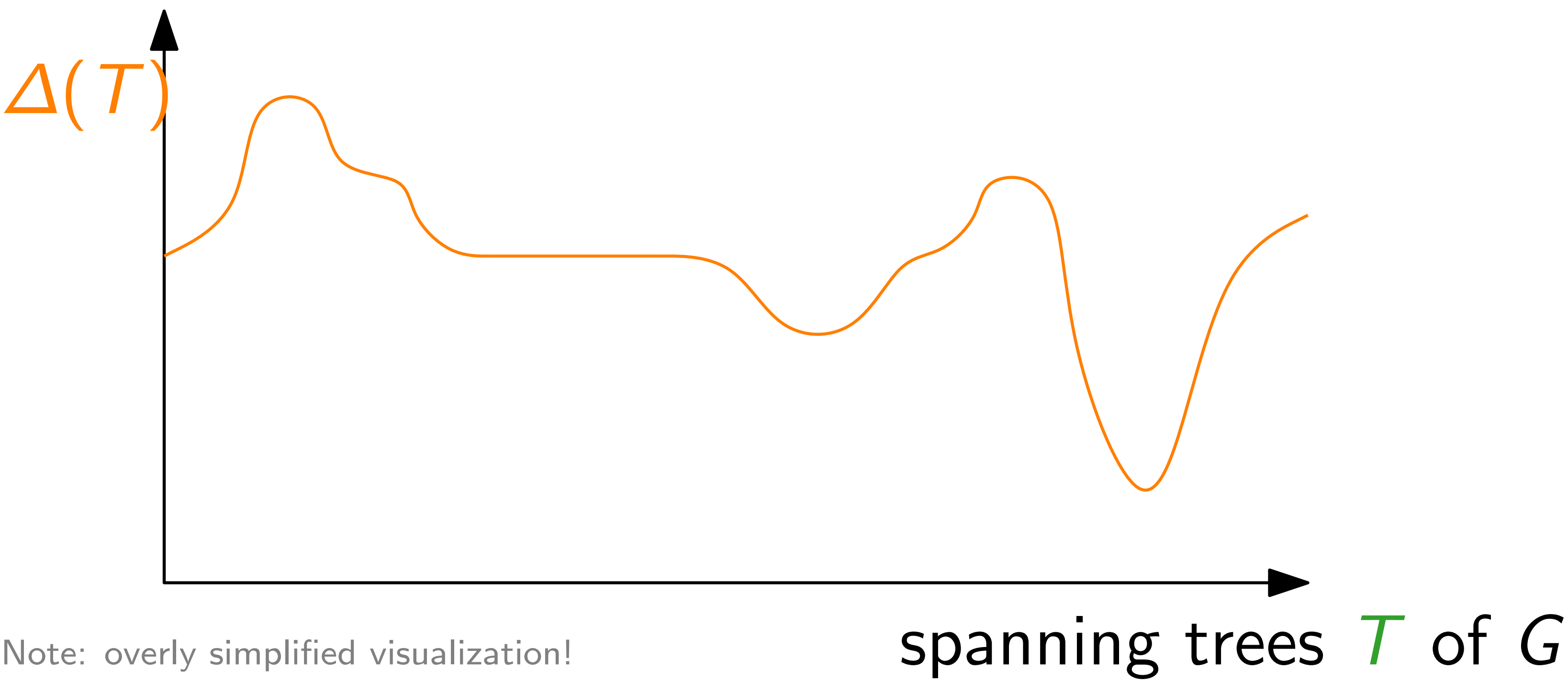
MinDegSpanningTreeLocalSearch(graph G)

$T \leftarrow$ any spanning tree of G

while $\exists$ improving flip in T for a vertex v with $\deg_T(v) \geq \Delta(T) - \ell$ **do**

- do the improving flip

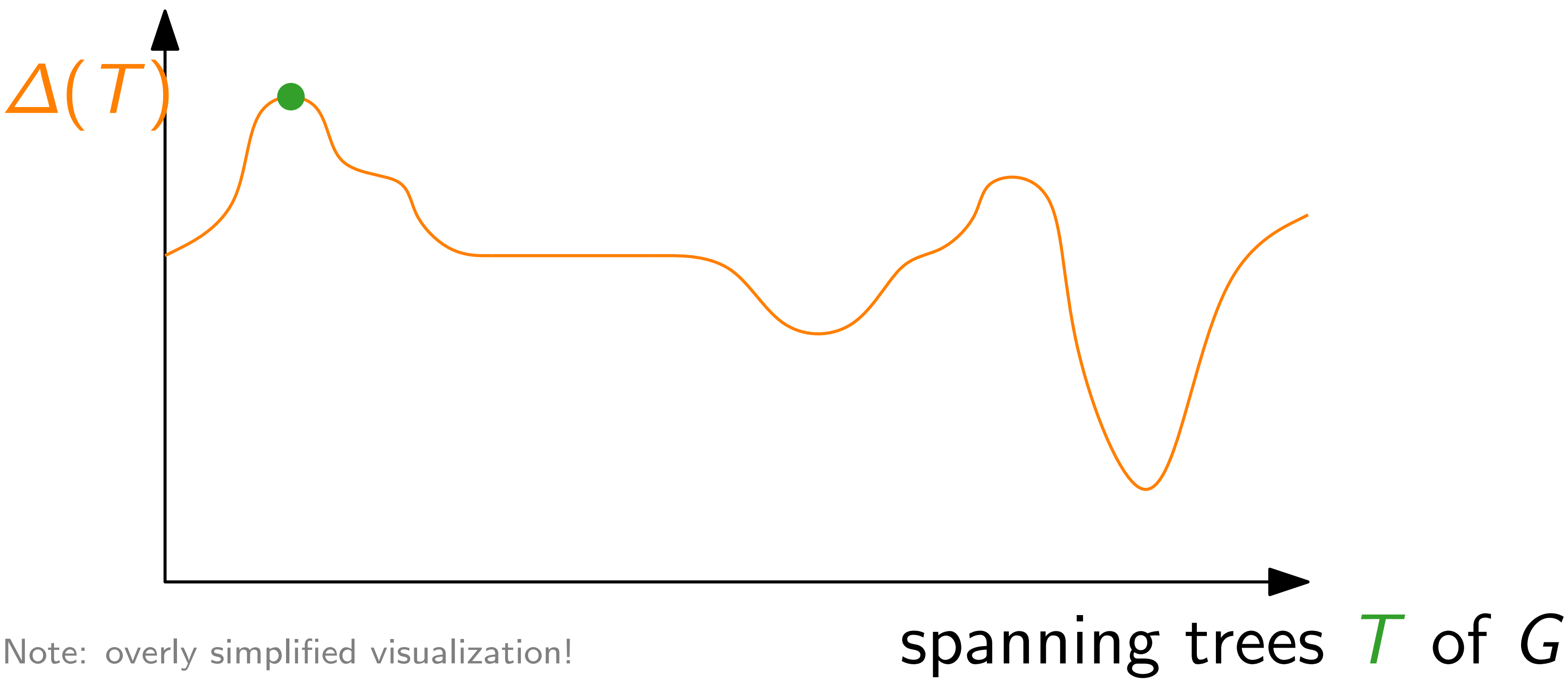
return T



Note: overly simplified visualization!

Local Search

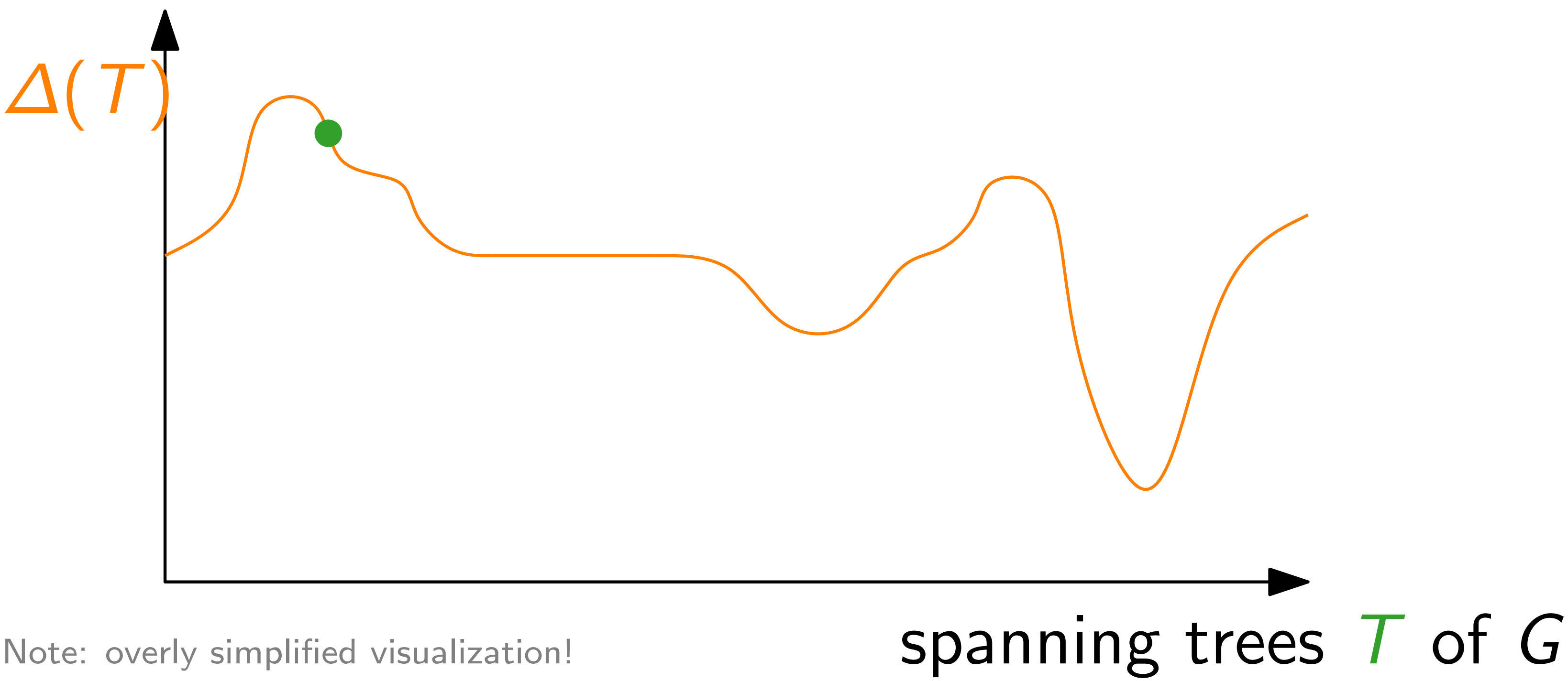
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

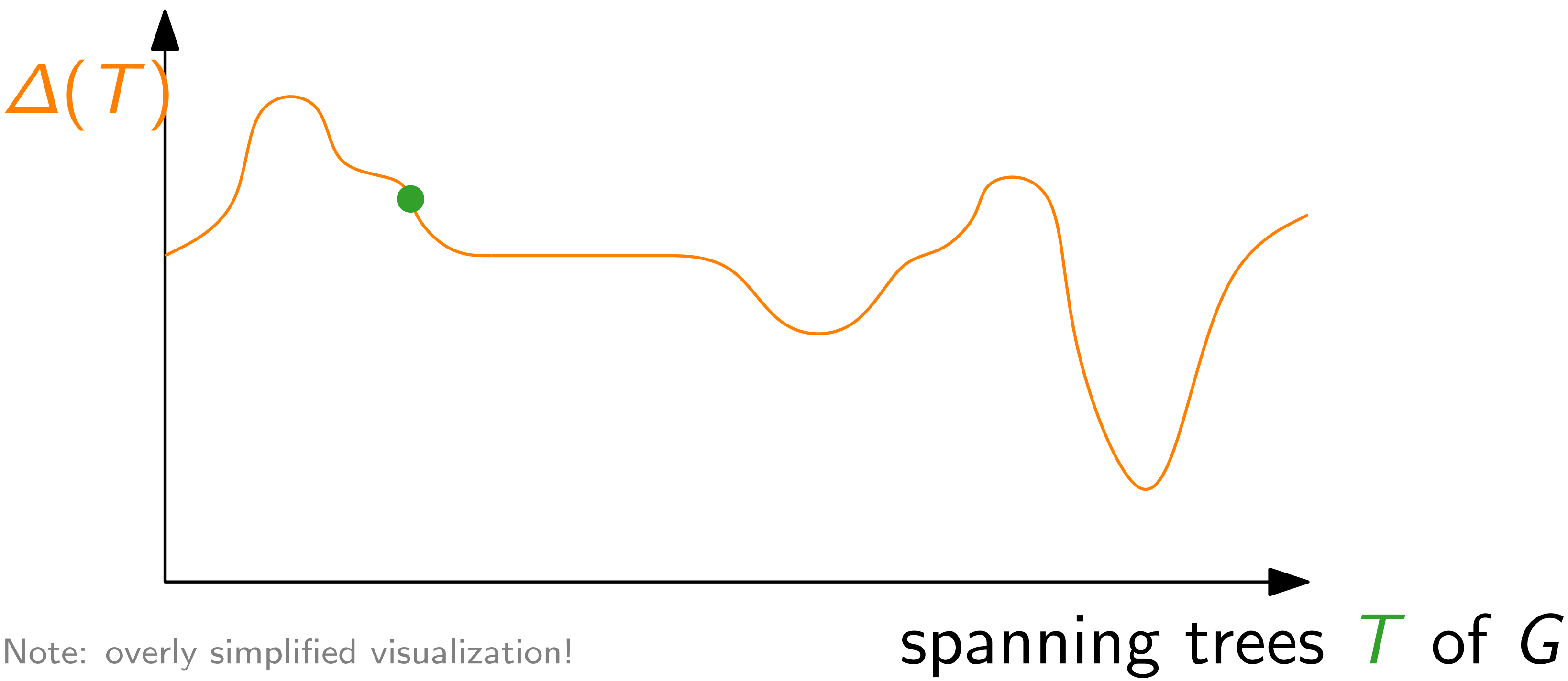
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
    with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

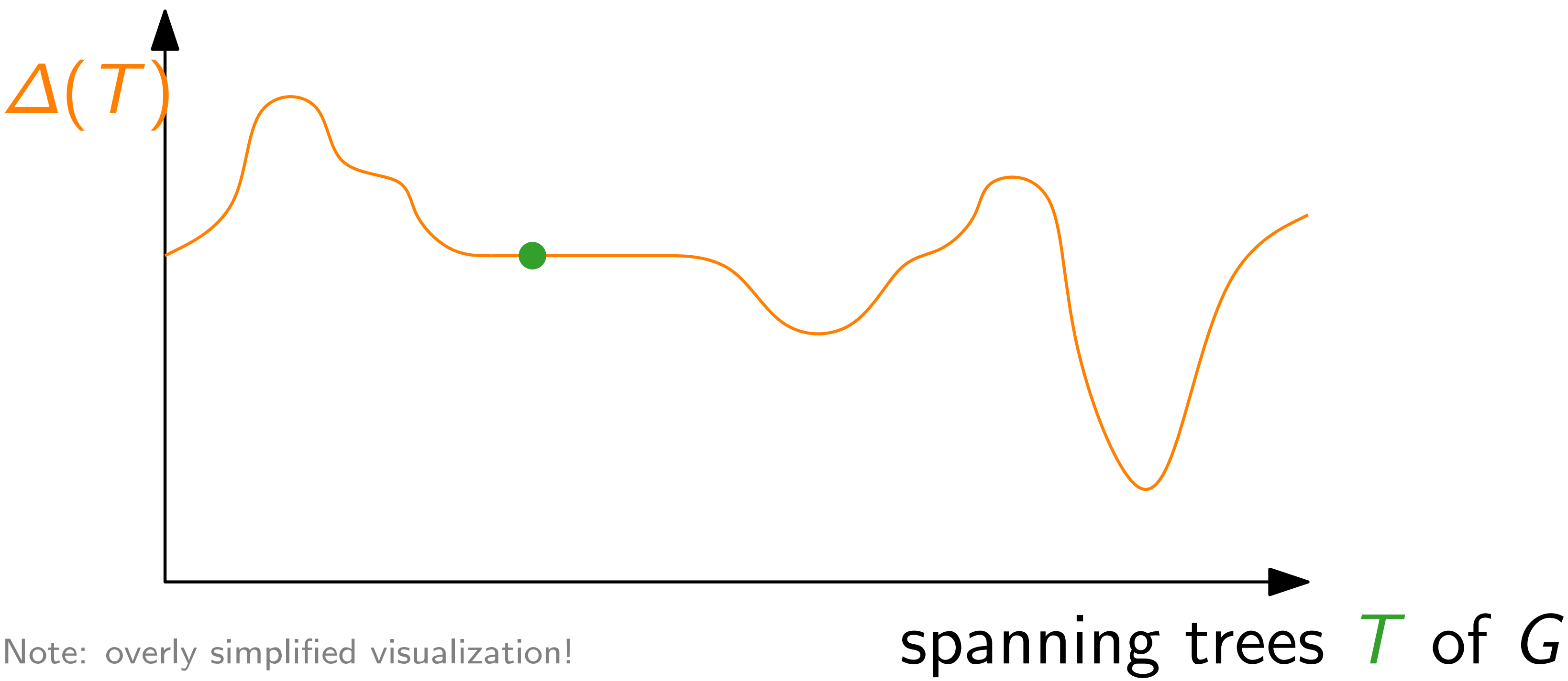
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

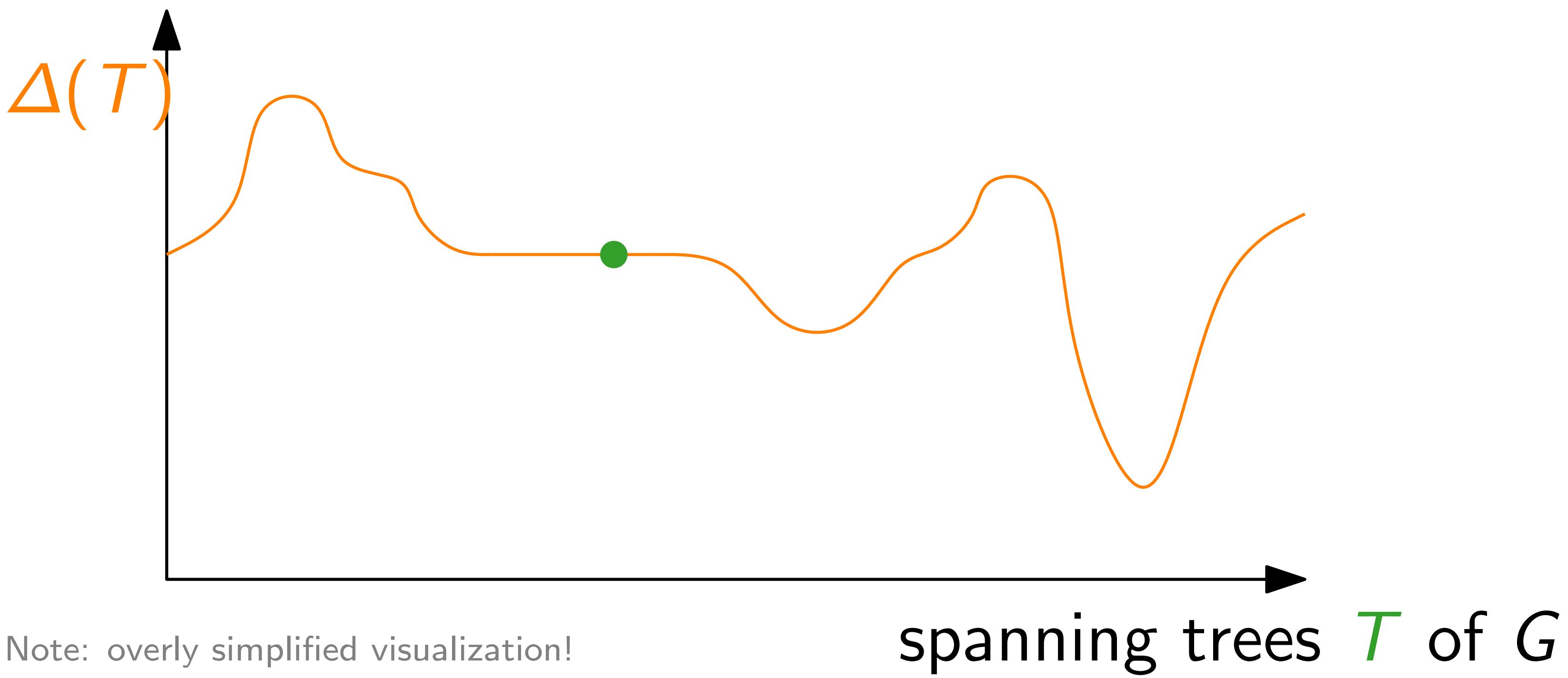
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
    with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

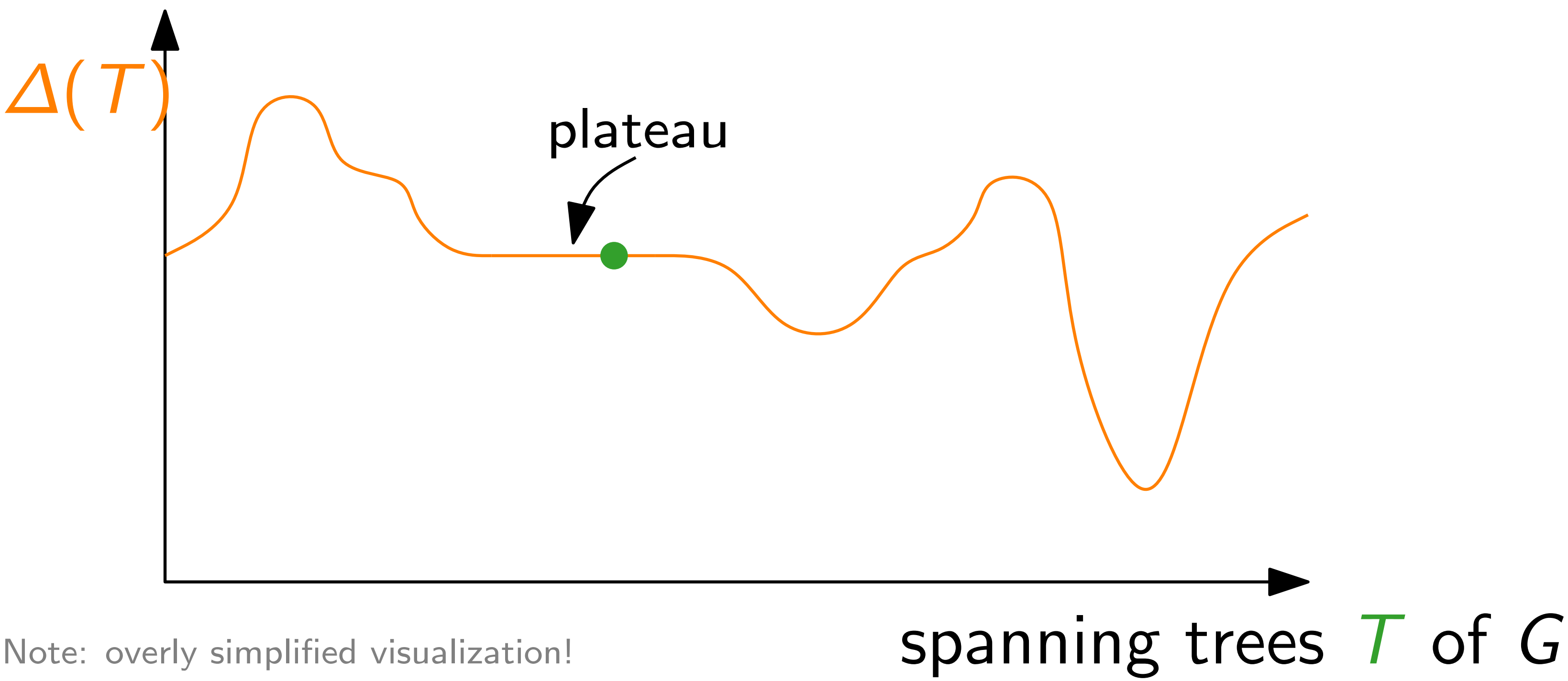
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
    with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

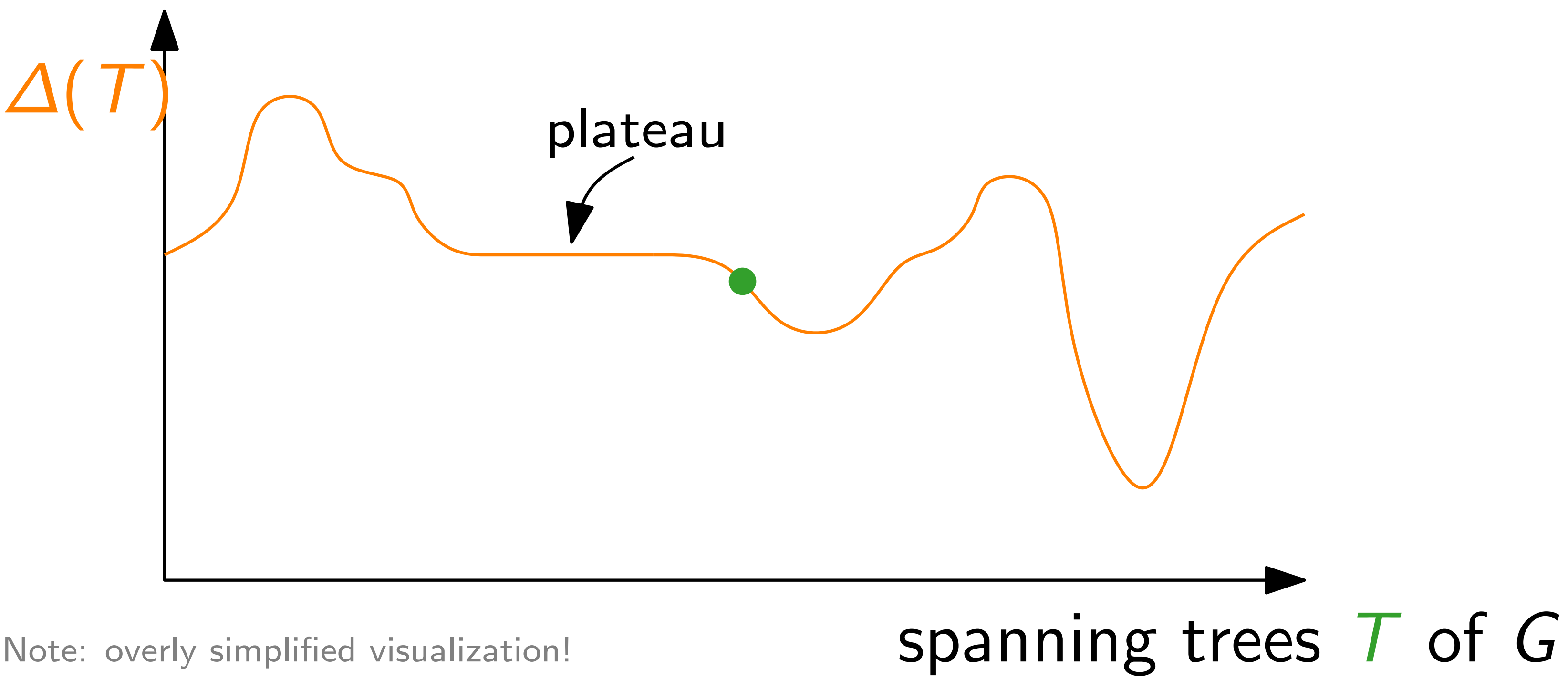
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

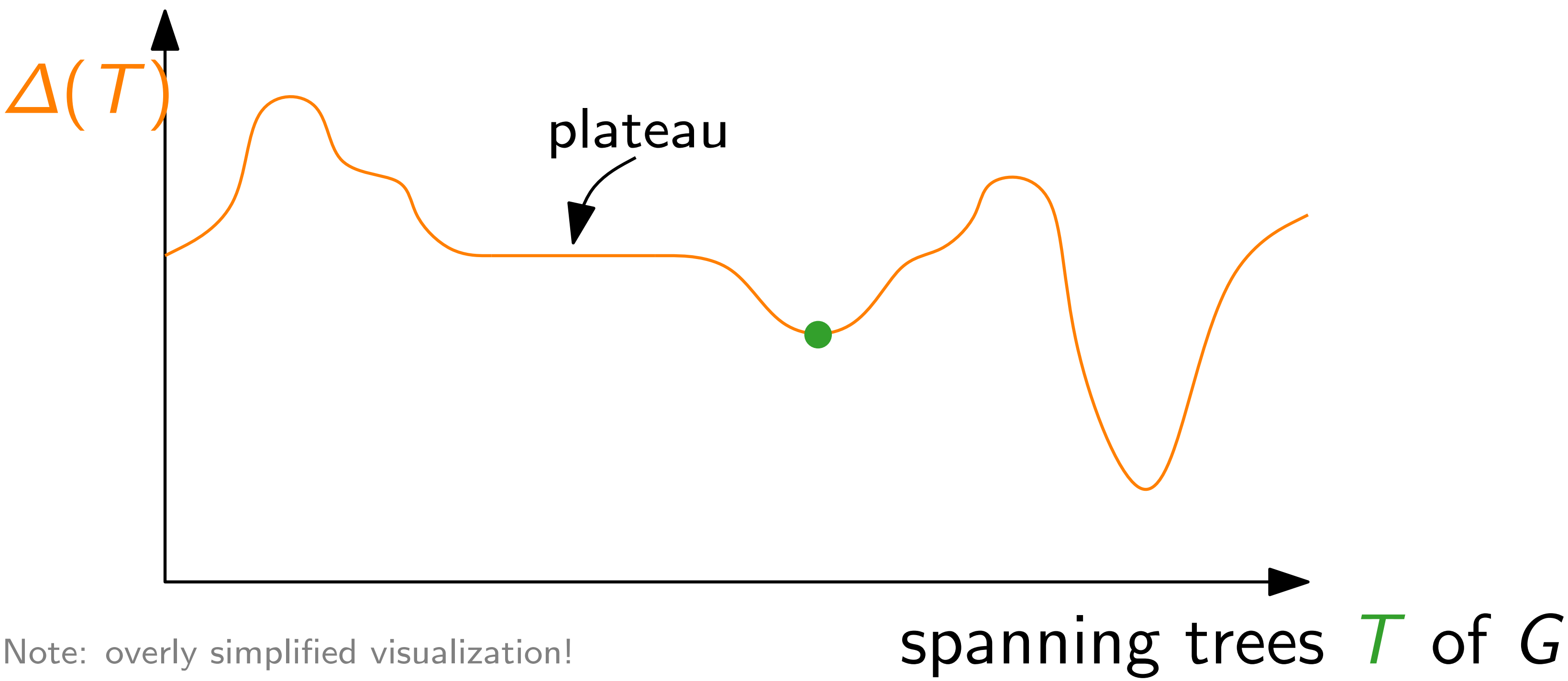
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
    with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

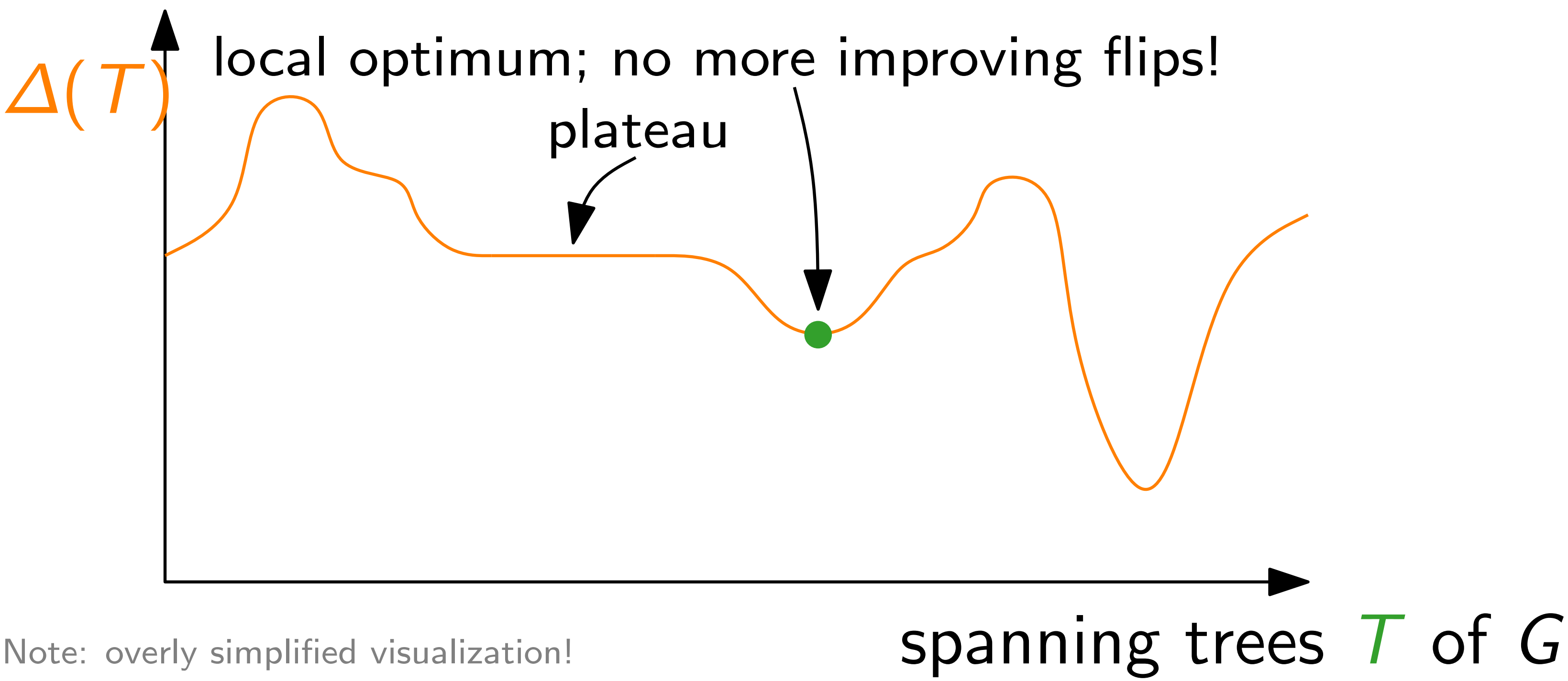
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
    with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

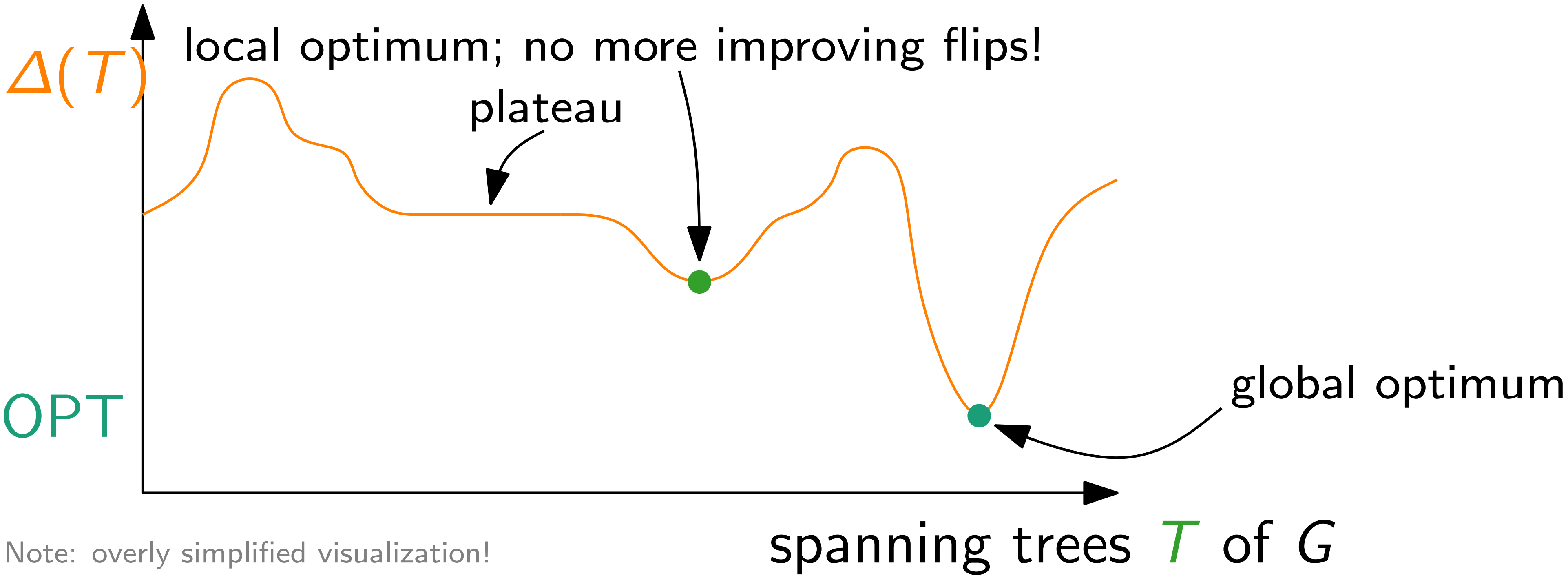
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

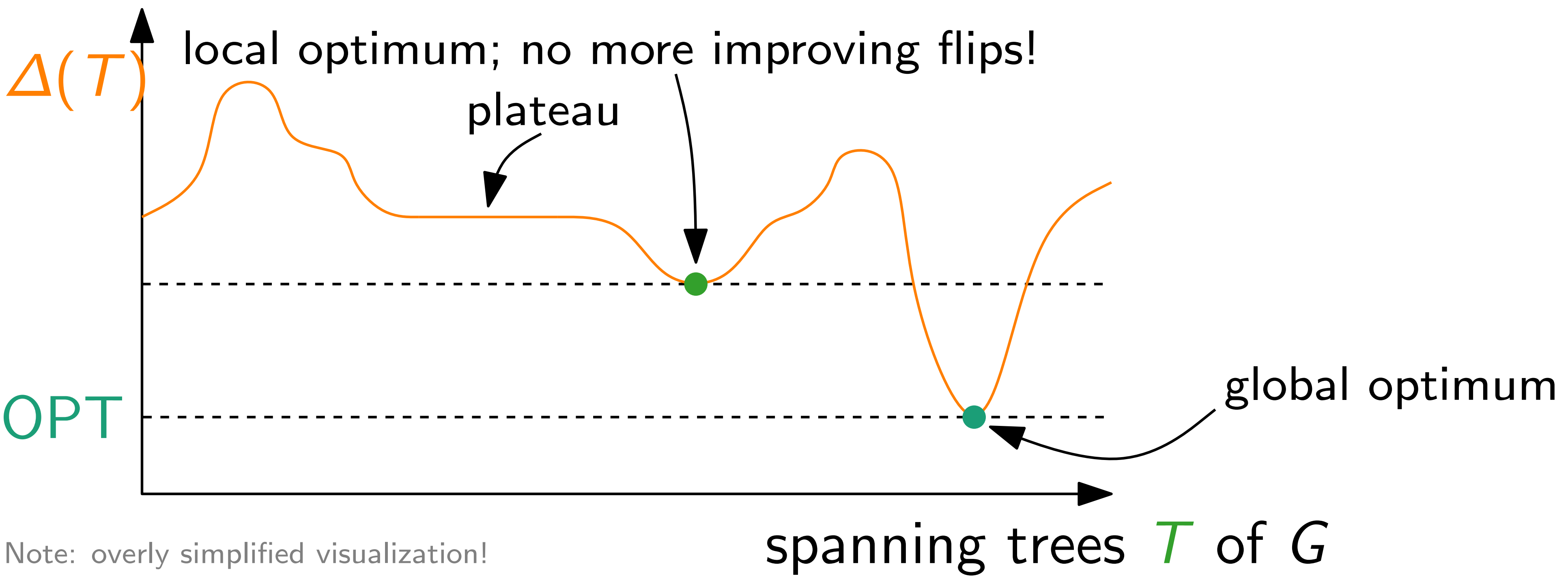
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

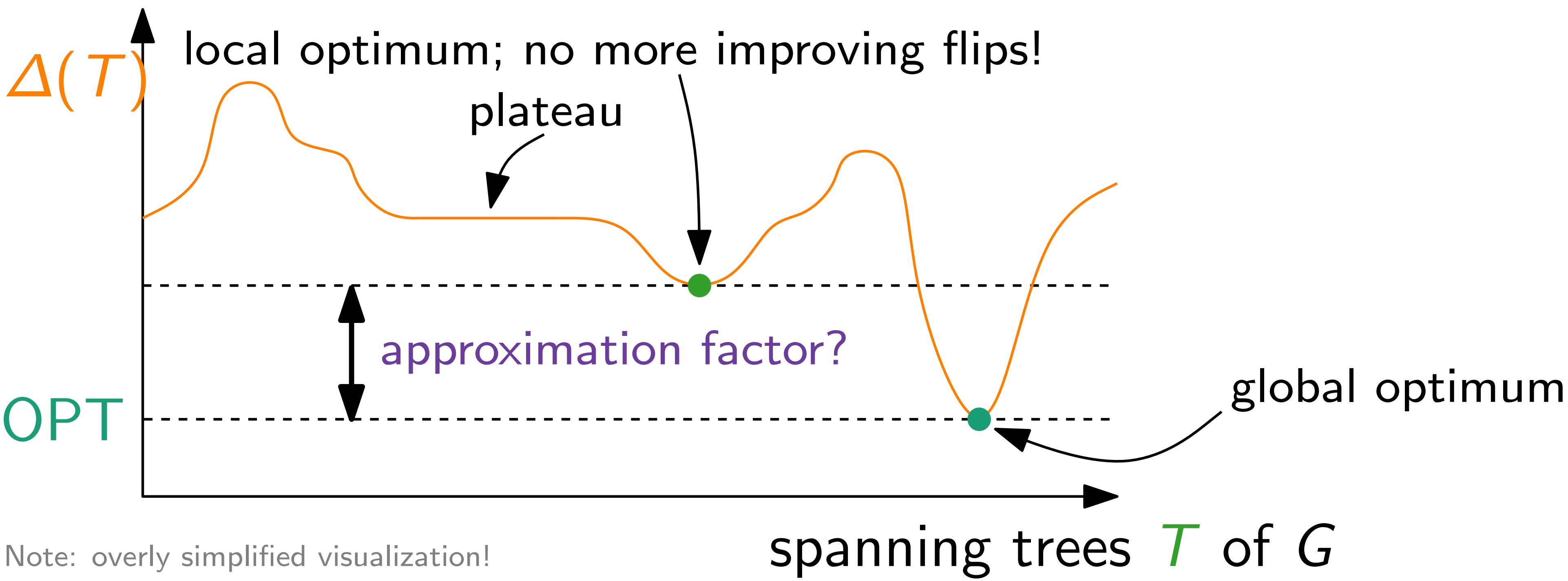
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T



Note: overly simplified visualization!

Local Search

MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T

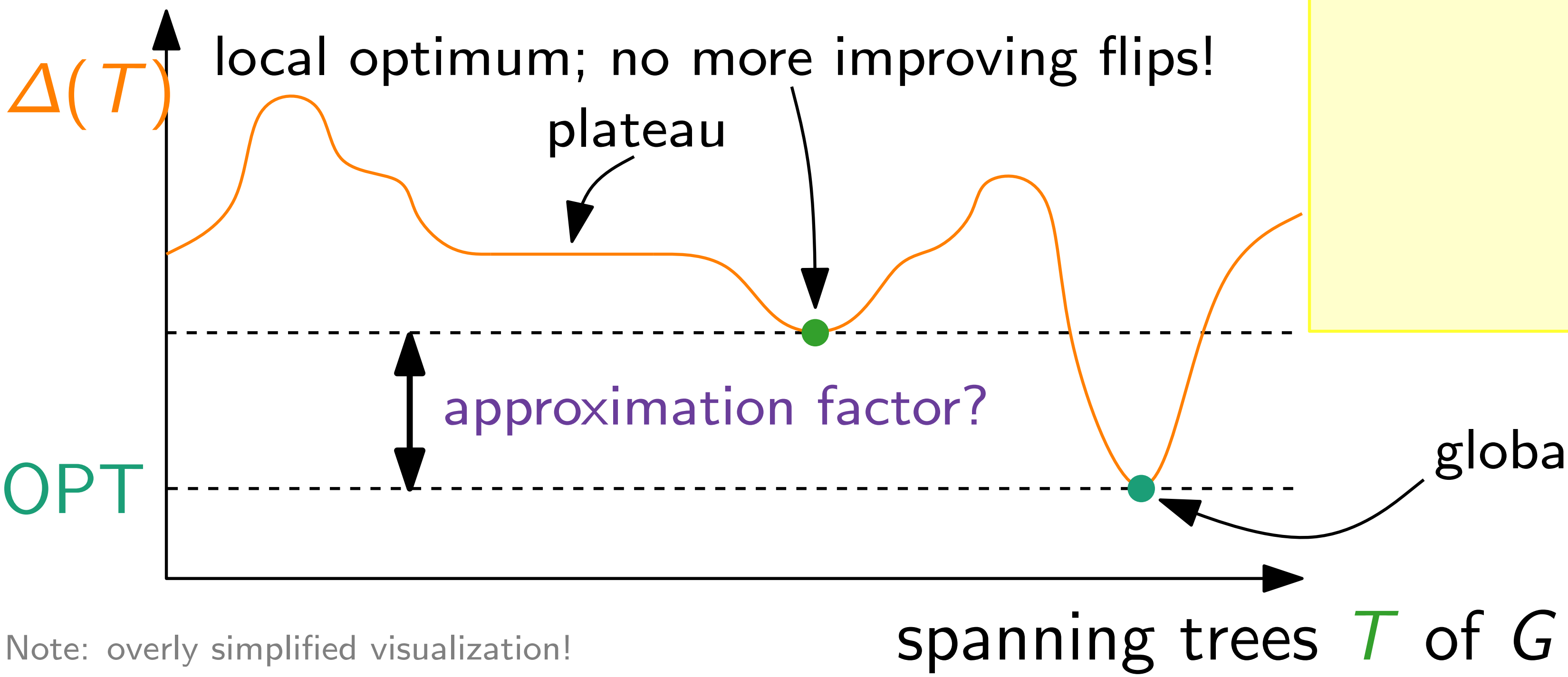


Note: overly simplified visualization!

Local Search

MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T

- Termination?

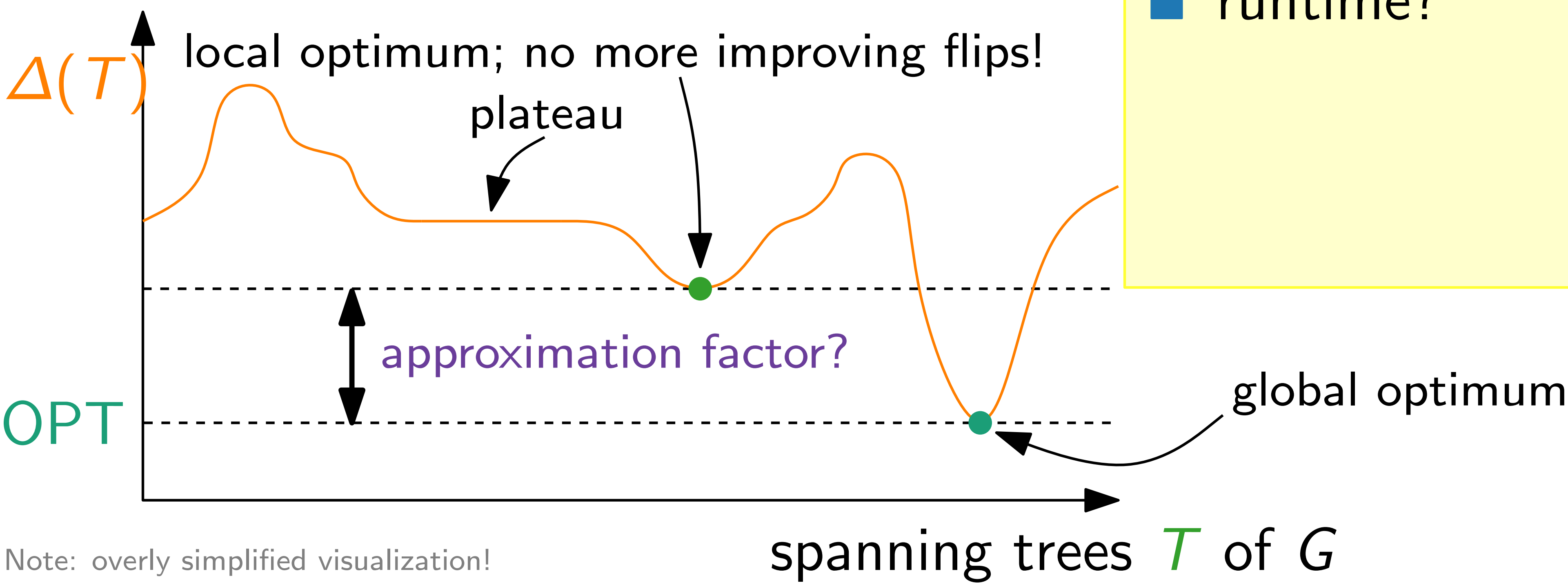


Note: overly simplified visualization!

Local Search

MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T

- Termination?
- runtime?

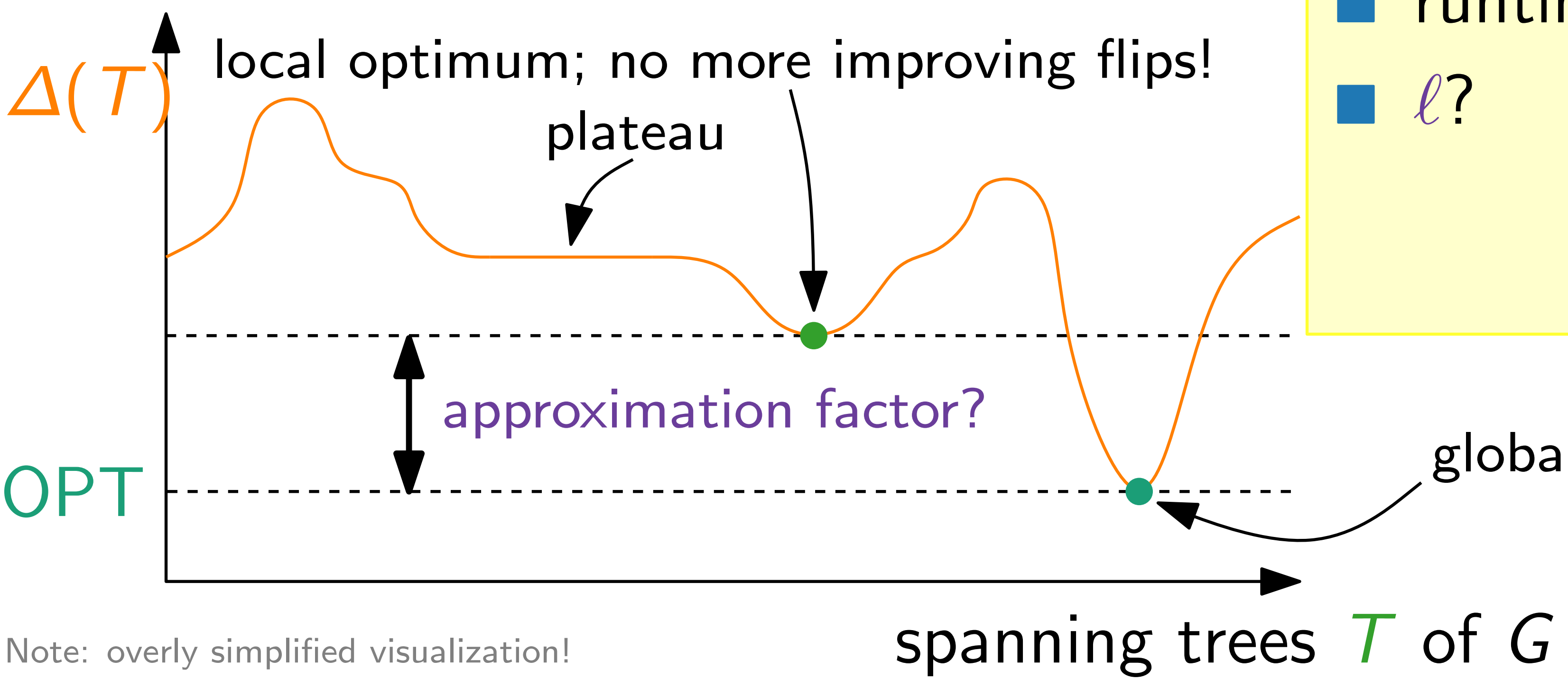


Note: overly simplified visualization!

Local Search

MinDegSpanningTreeLocalSearch(graph G)
 $T \leftarrow$ any spanning tree of G
 while $\exists$ improving flip in T for a vertex v
  with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
  do the improving flip
 return T

- Termination?
- runtime?
- ℓ?

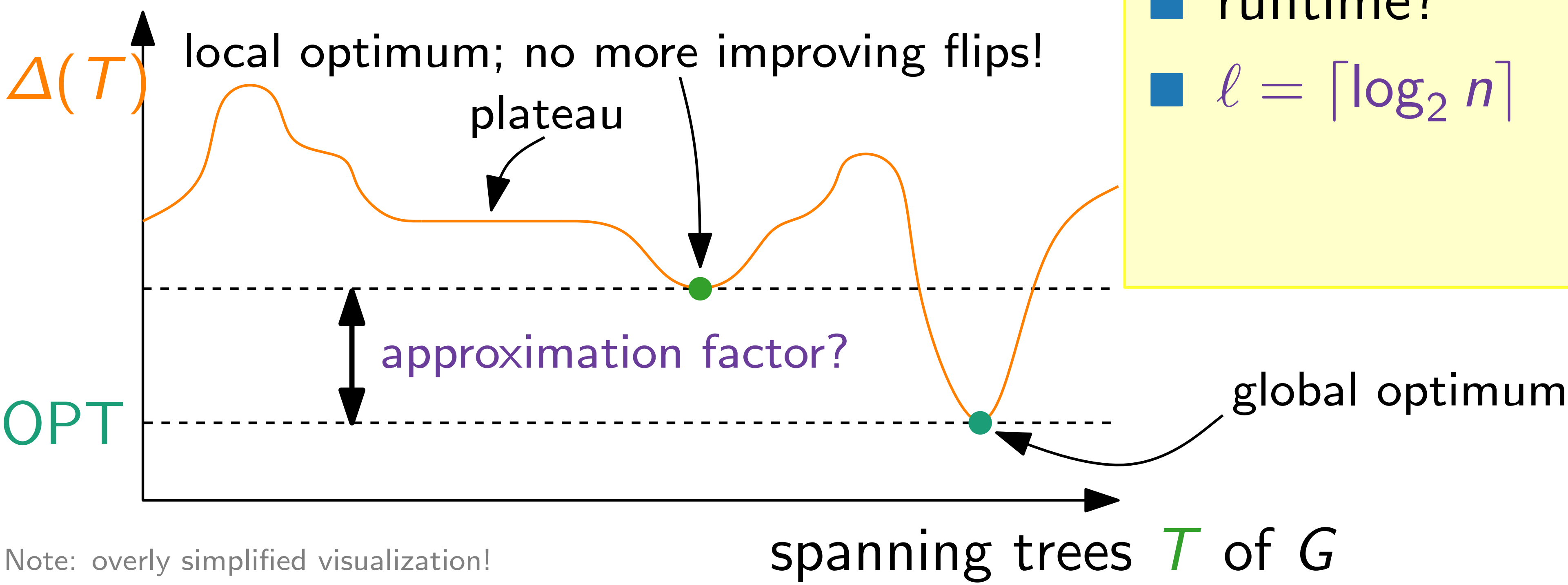


Note: overly simplified visualization!

Local Search

MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
    with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T

- Termination?
- runtime?
- $\ell = \lceil \log_2 n \rceil$

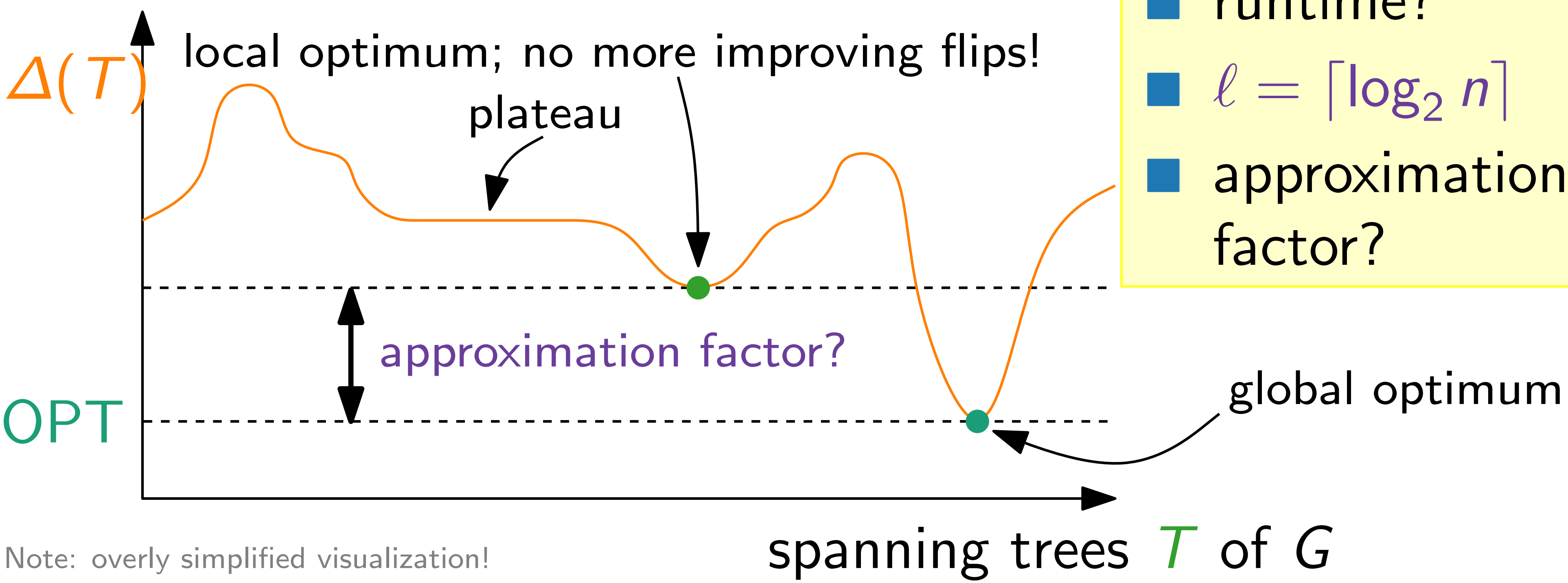


Note: overly simplified visualization!

Local Search

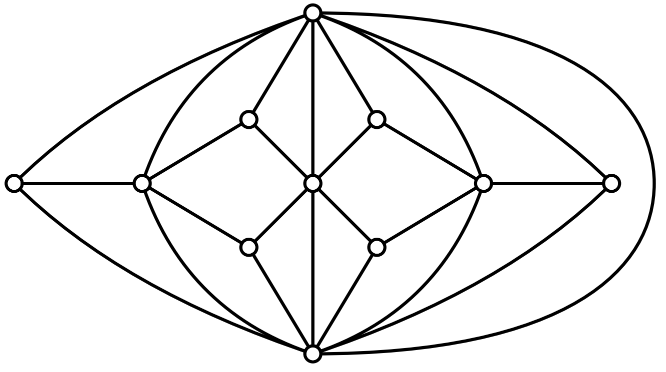
MinDegSpanningTreeLocalSearch(graph G)
  $T \leftarrow$ any spanning tree of G
  while $\exists$ improving flip in T for a vertex v
      with $\deg_T(v) \geq \Delta(T) - \ell$ **do**
    do the improving flip
  return T

- Termination?
- runtime?
- $\ell = \lceil \log_2 n \rceil$
- approximation factor?

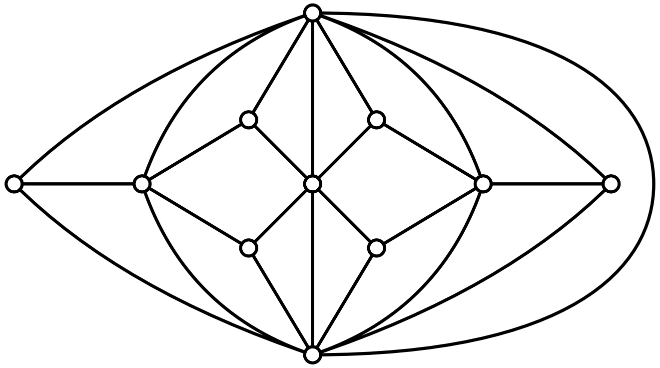


Note: overly simplified visualization!

Example

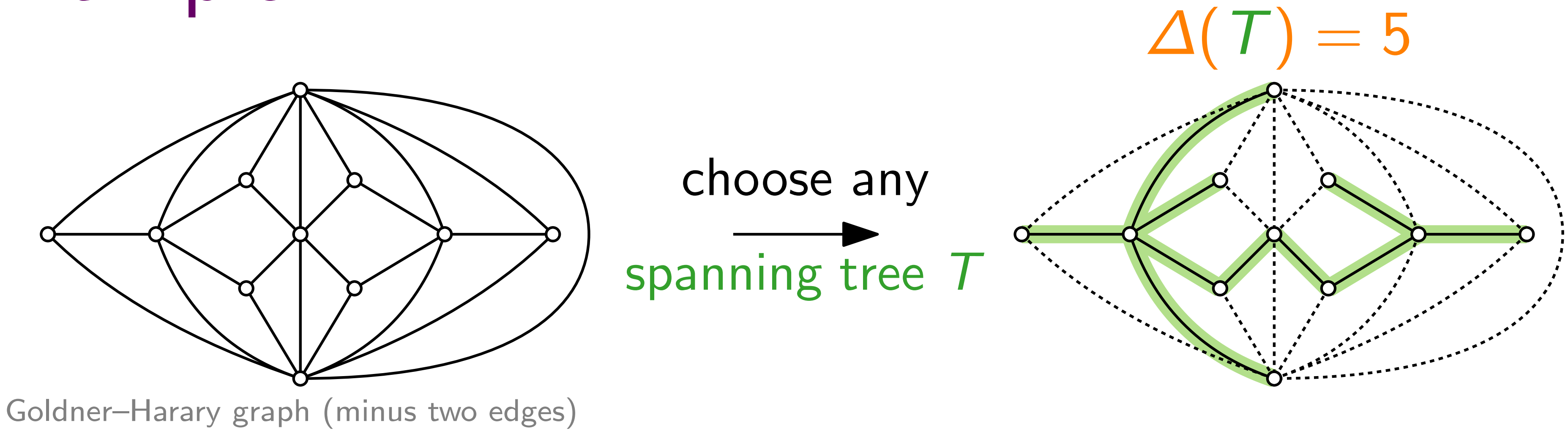


Example

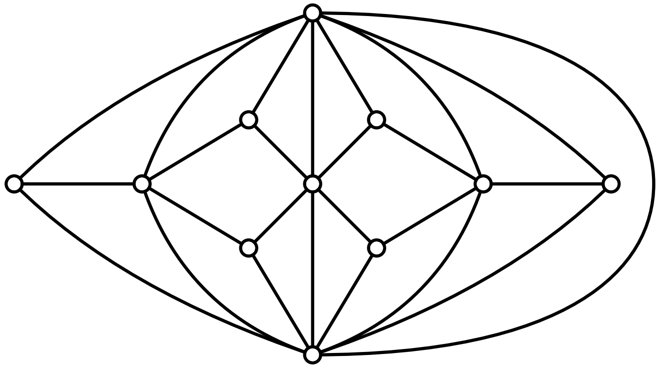


Goldner–Harary graph (minus two edges)

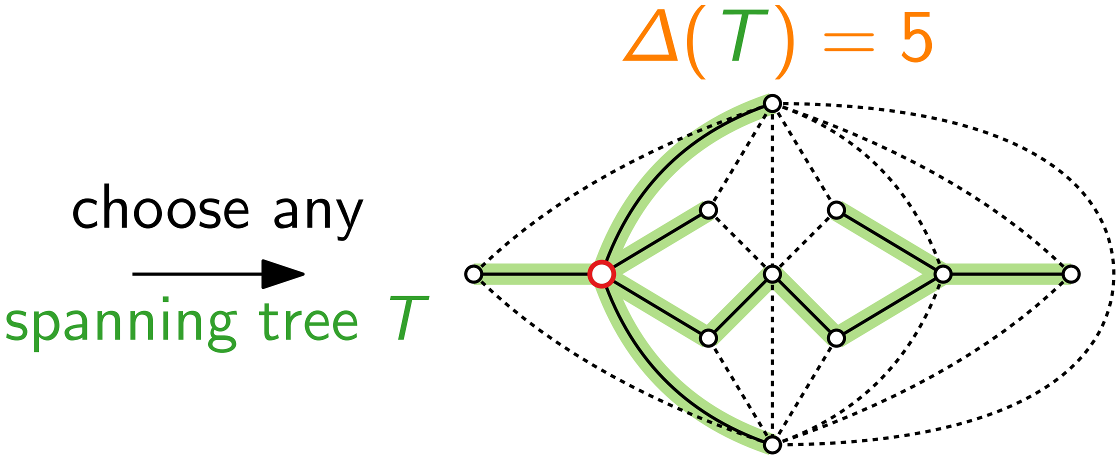
Example


$\Delta(T) = 5$
choose any
spanning tree T
Goldner–Harary graph (minus two edges)

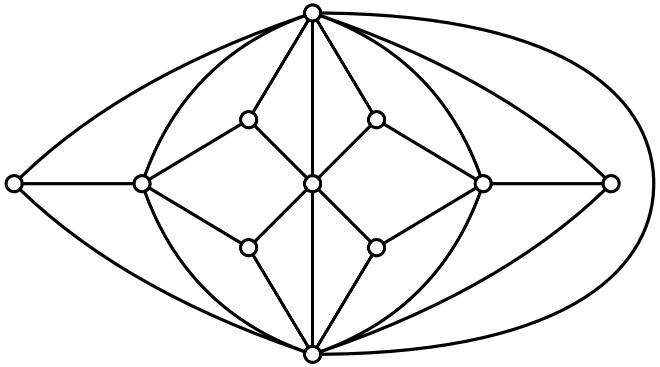
Example



Goldner–Harary graph (minus two edges)



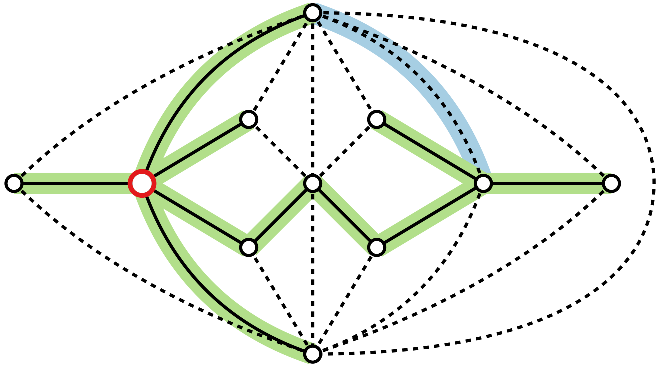
Example



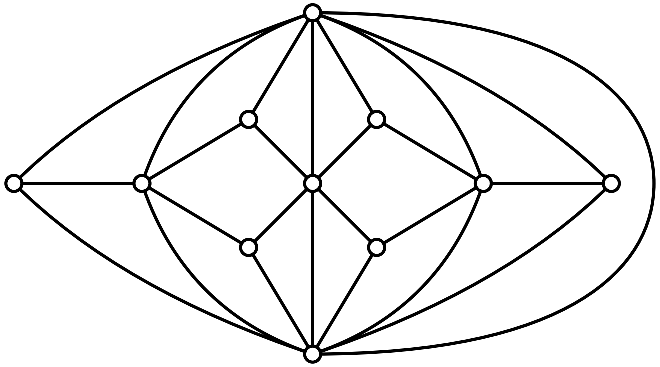
Goldner–Harary graph (minus two edges)

choose any → spanning tree T

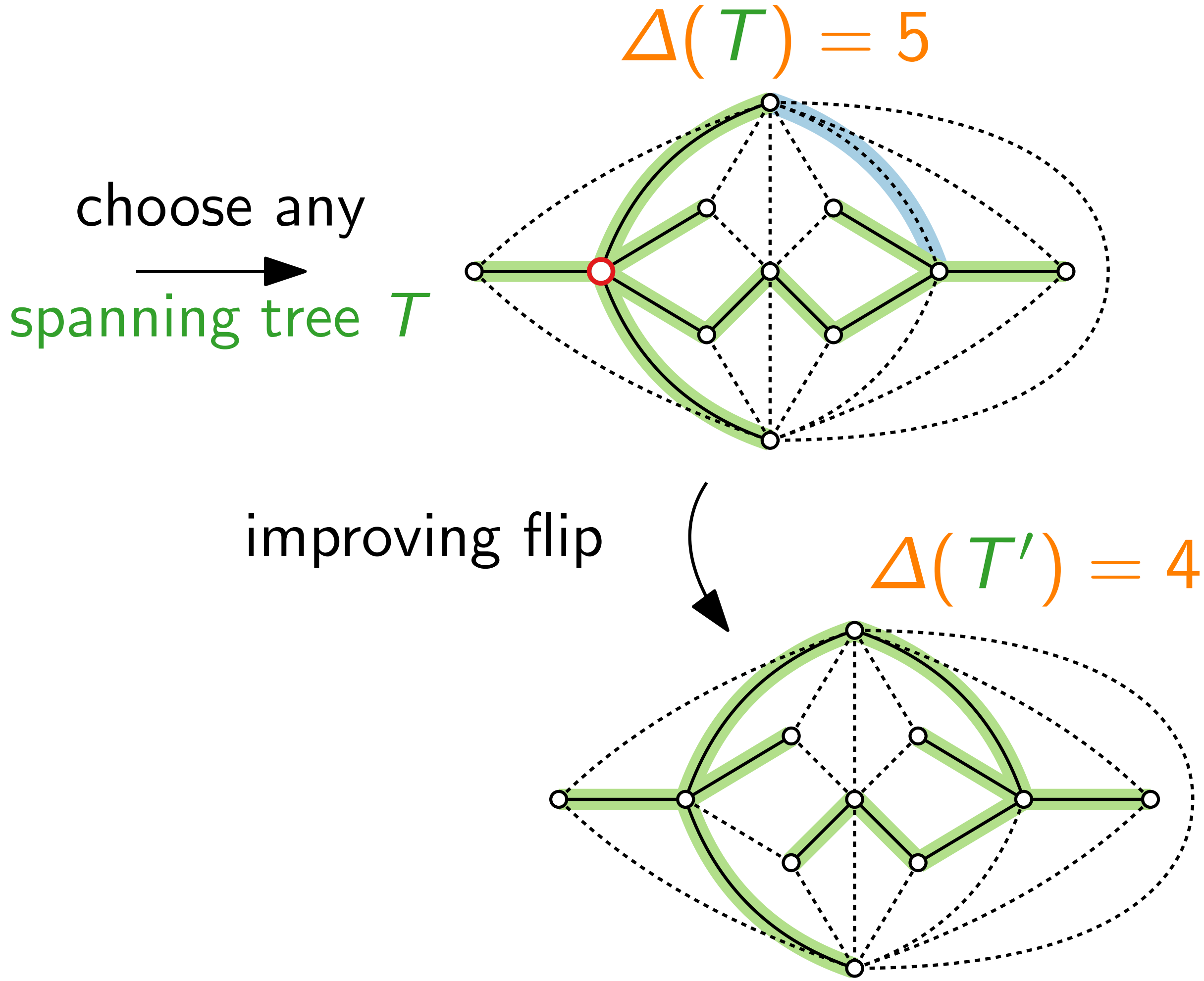
$\Delta(T) = 5$



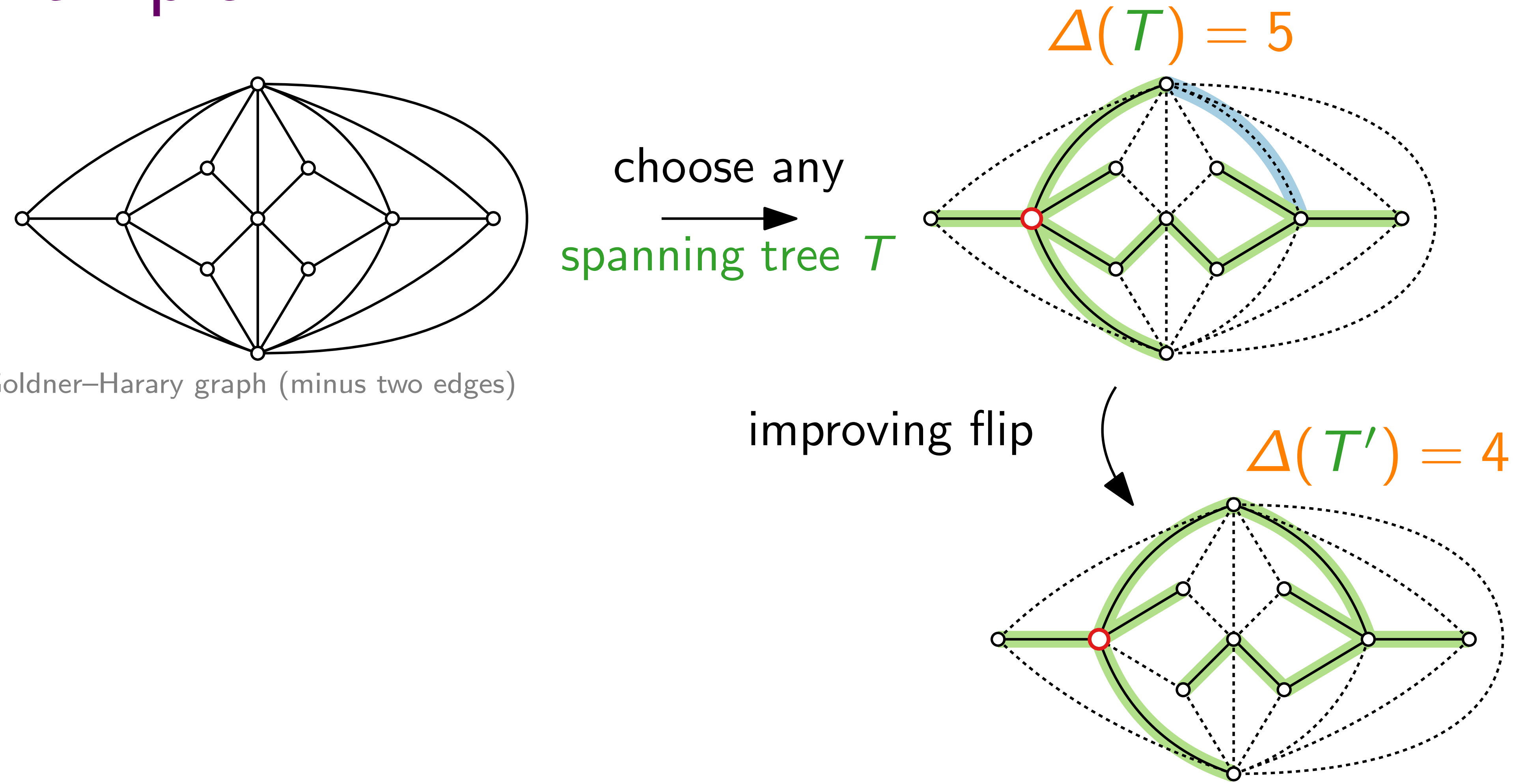
Example



Goldner–Harary graph (minus two edges)

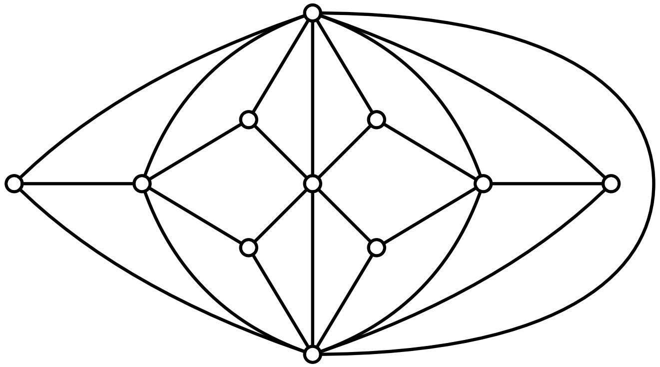


Example

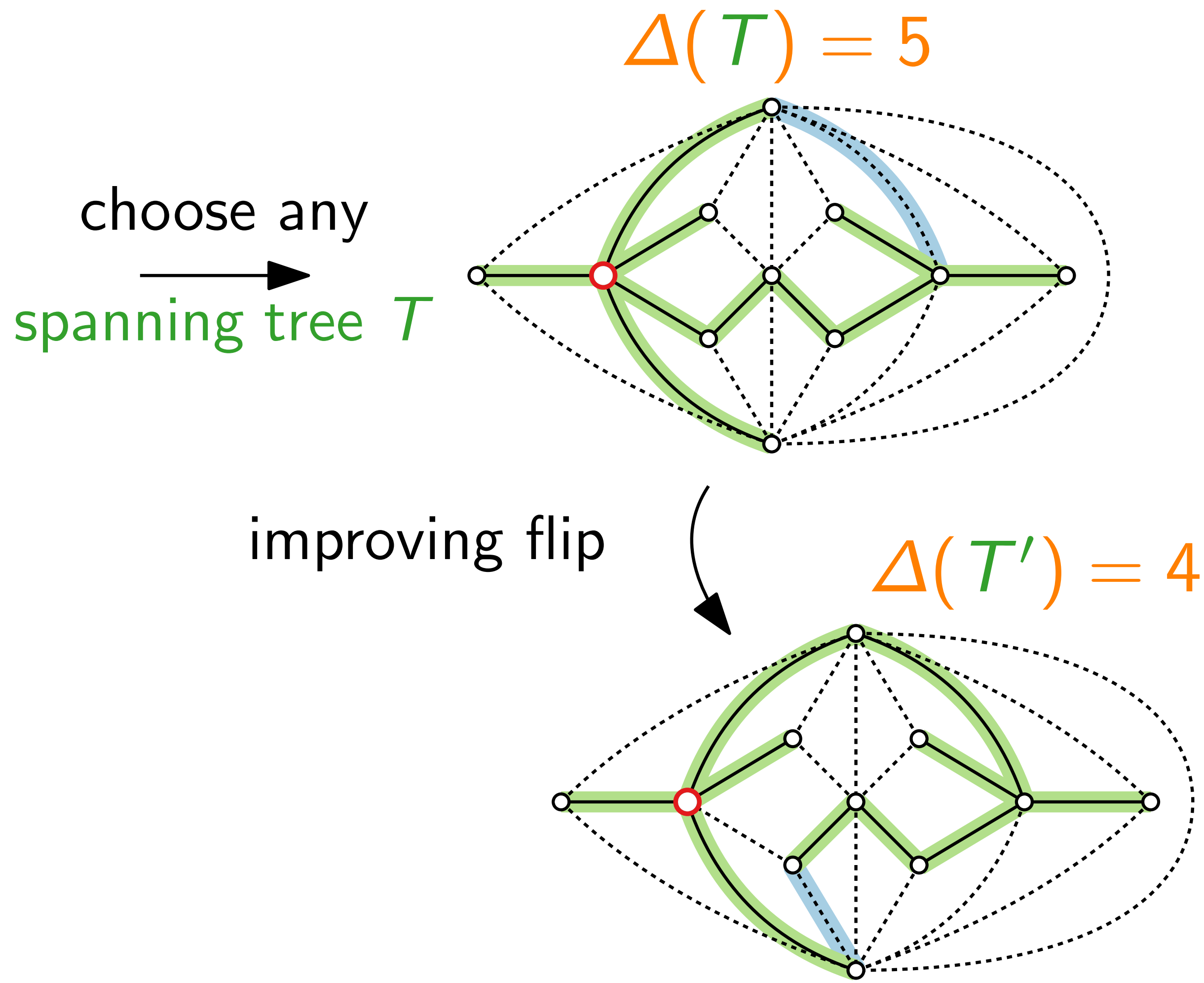


Goldner–Harary graph (minus two edges)
choose any
spanning tree T
$\Delta(T) = 5$
improving flip
$\Delta(T') = 4$

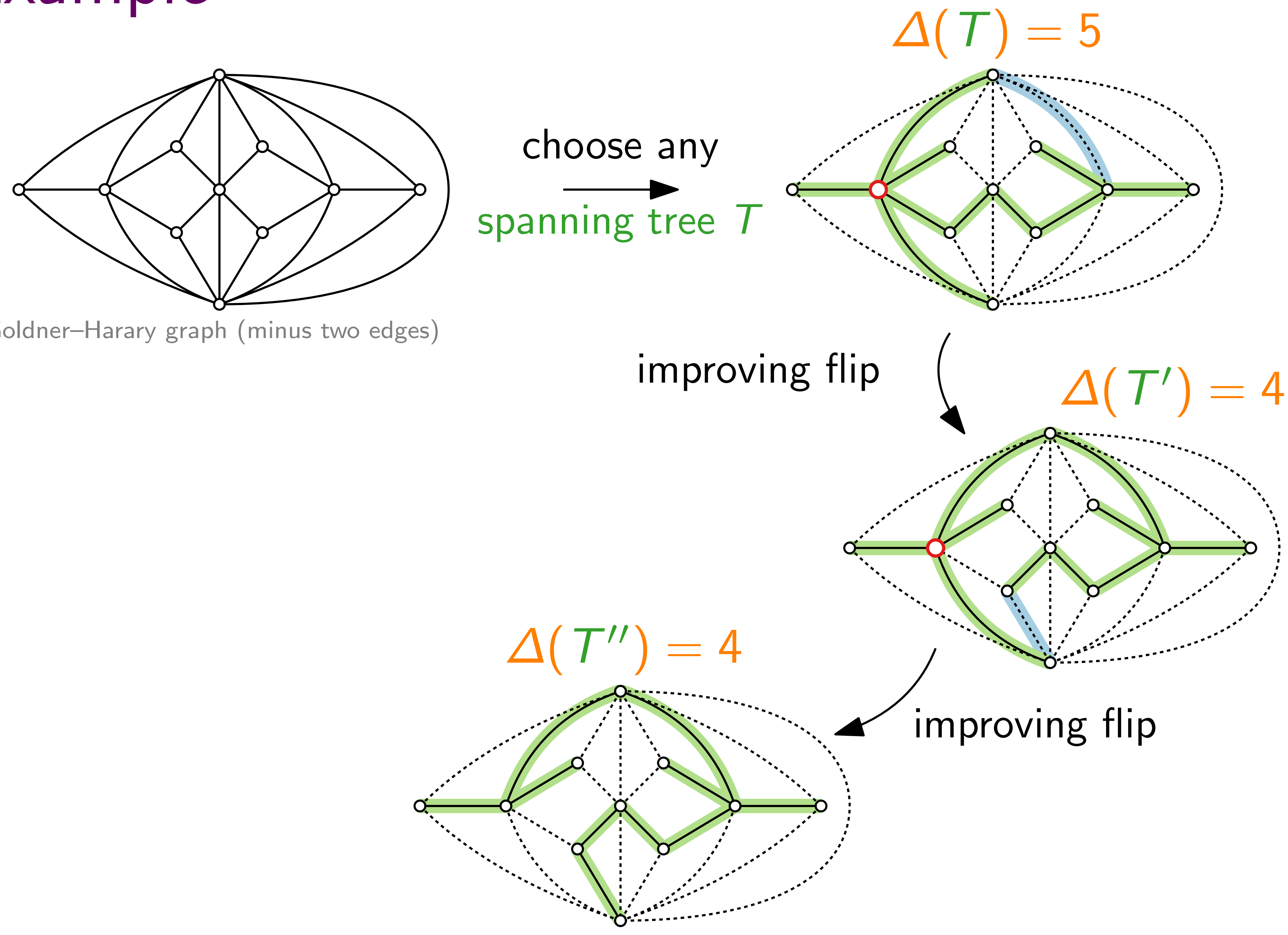
Example



Goldner–Harary graph (minus two edges)

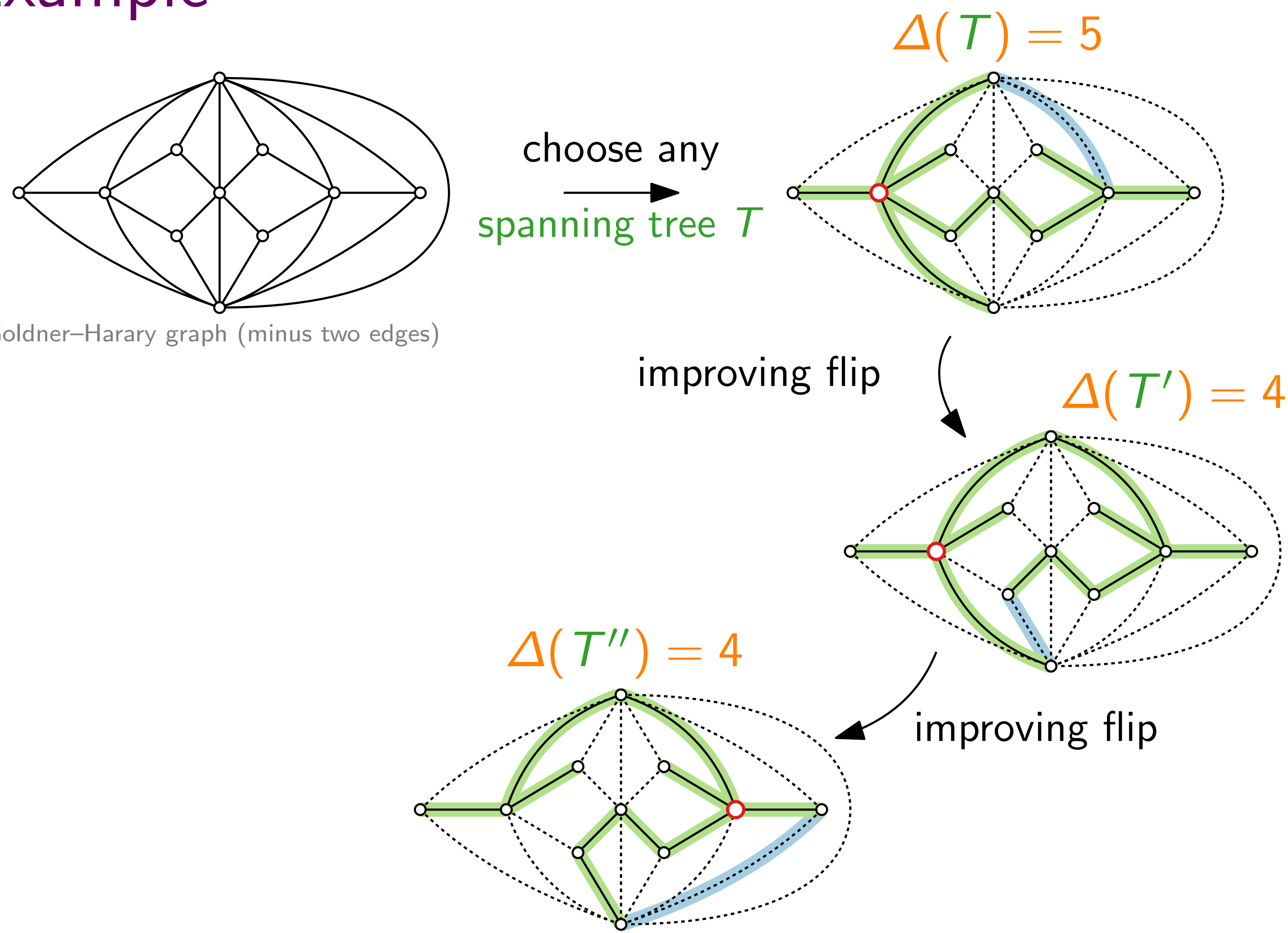


Example



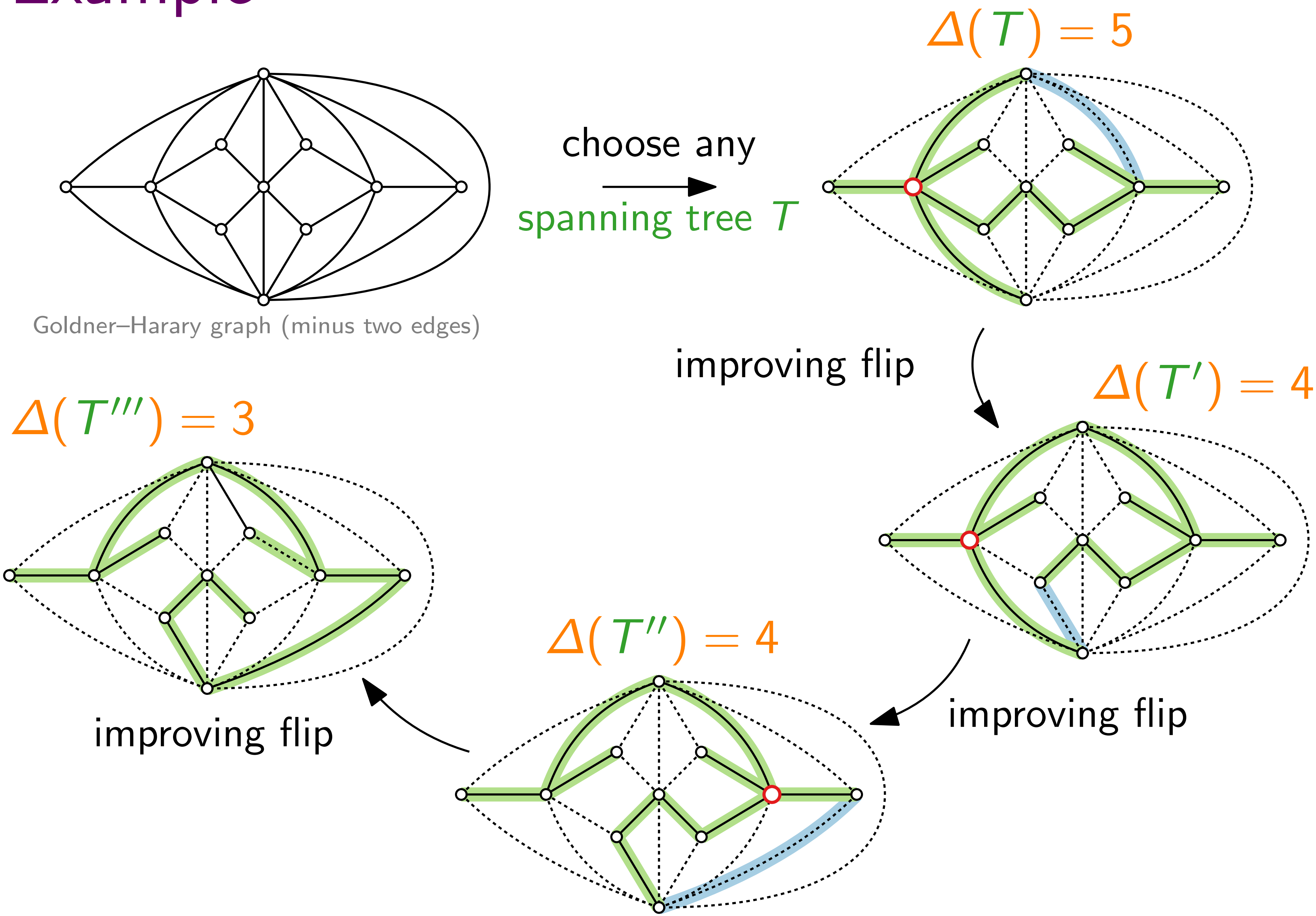
Goldner–Harary graph (minus two edges)
choose any
spanning tree T
$\Delta(T) = 5$
improving flip
$\Delta(T') = 4$
improving flip
$\Delta(T'') = 4$

Example



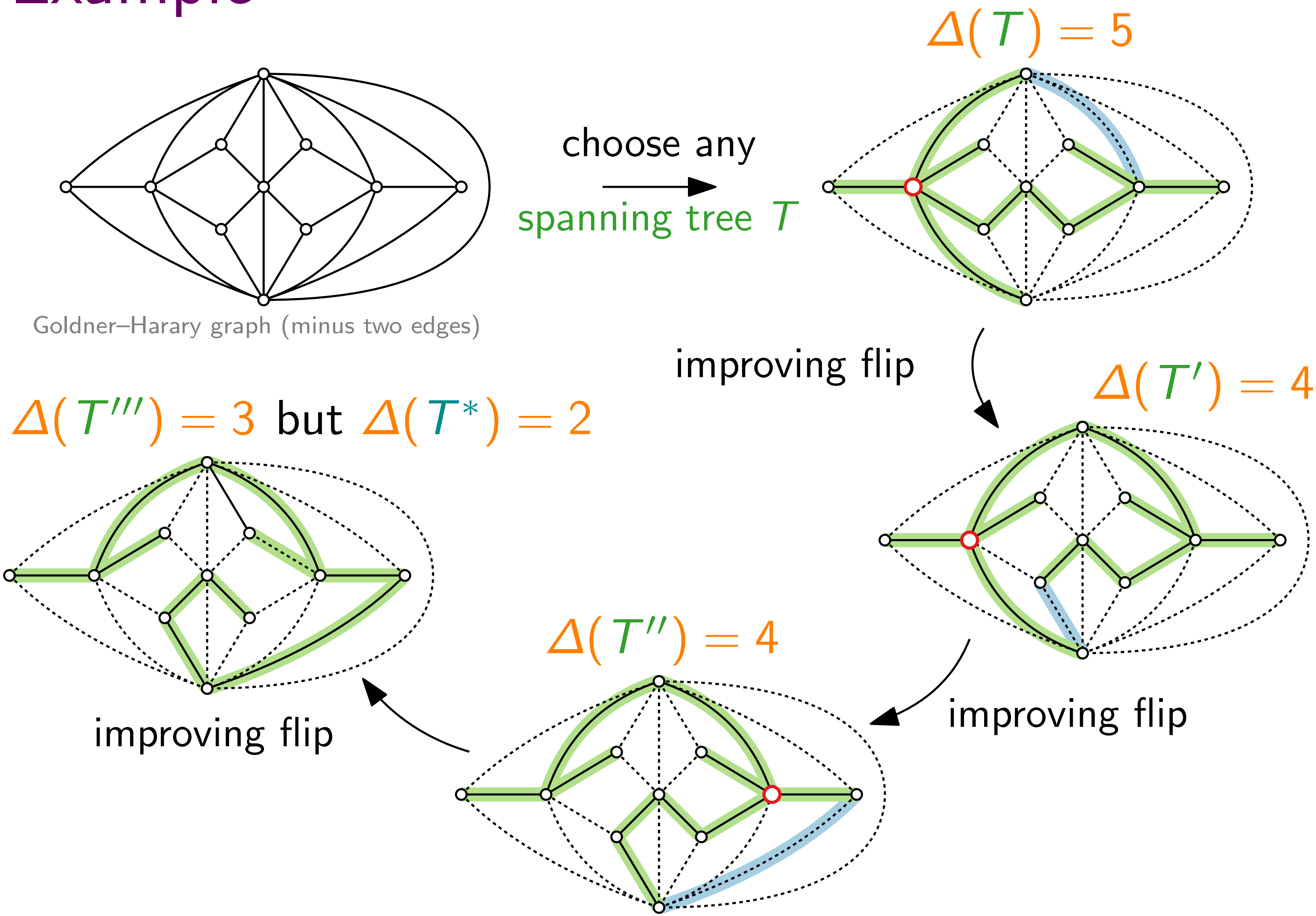
Goldner–Harary graph (minus two edges)

Example



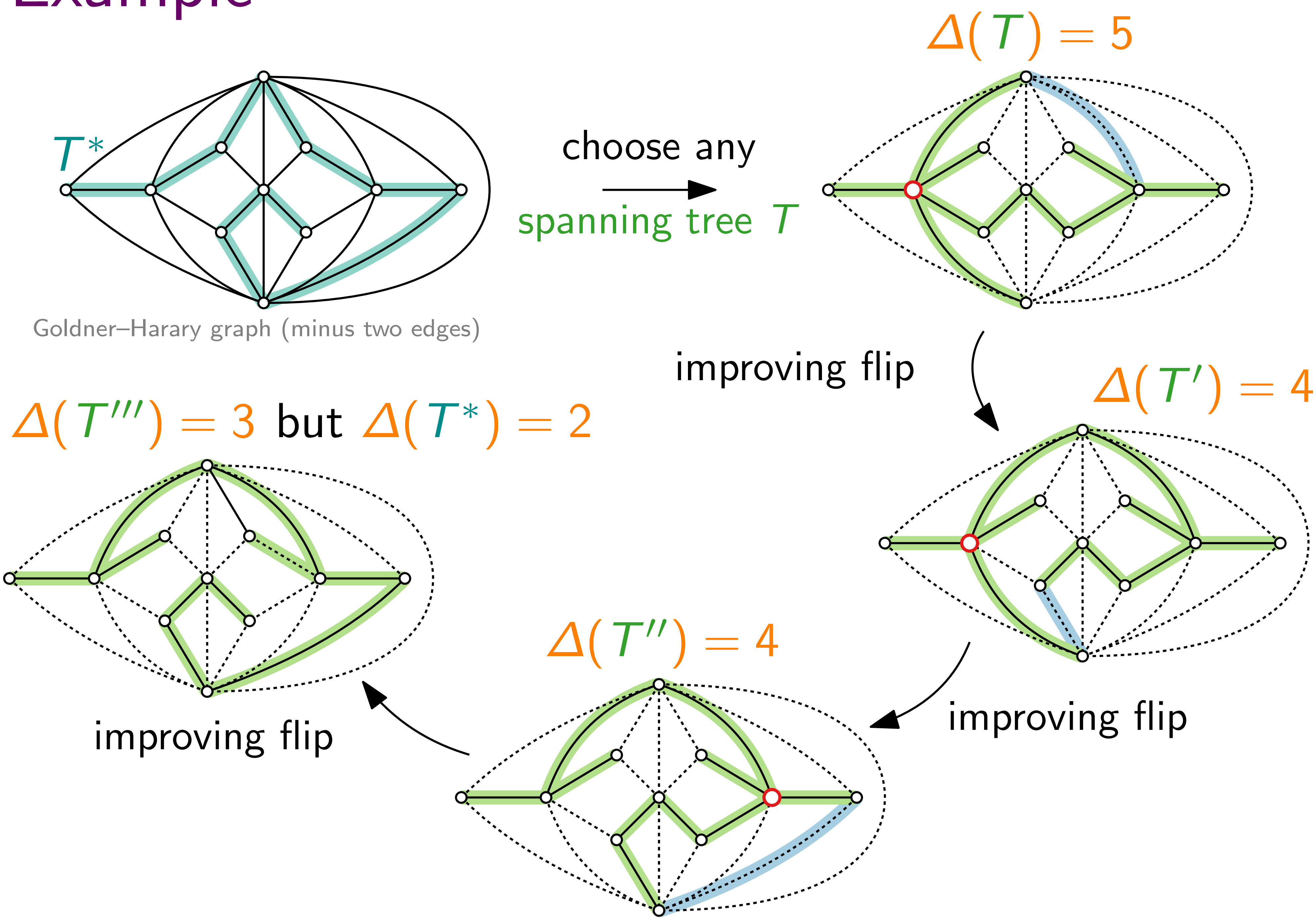
Goldner–Harary graph (minus two edges)
choose any
spanning tree T
$\Delta(T) = 5$
improving flip
$\Delta(T') = 4$
improving flip
$\Delta(T'') = 4$
improving flip
$\Delta(T''') = 3$

Example



Goldner–Harary graph (minus two edges)
choose any
spanning tree T
$\Delta(T) = 5$
improving flip
$\Delta(T') = 4$
improving flip
$\Delta(T'') = 4$
improving flip
$\Delta(T''') = 3$ but $\Delta(T^*) = 2$

Example



T^*
Goldner–Harary graph (minus two edges)
choose any
spanning tree T
$\Delta(T) = 5$
improving flip
$\Delta(T') = 4$
improving flip
$\Delta(T'') = 4$
improving flip
$\Delta(T''') = 3$ but $\Delta(T^*) = 2$

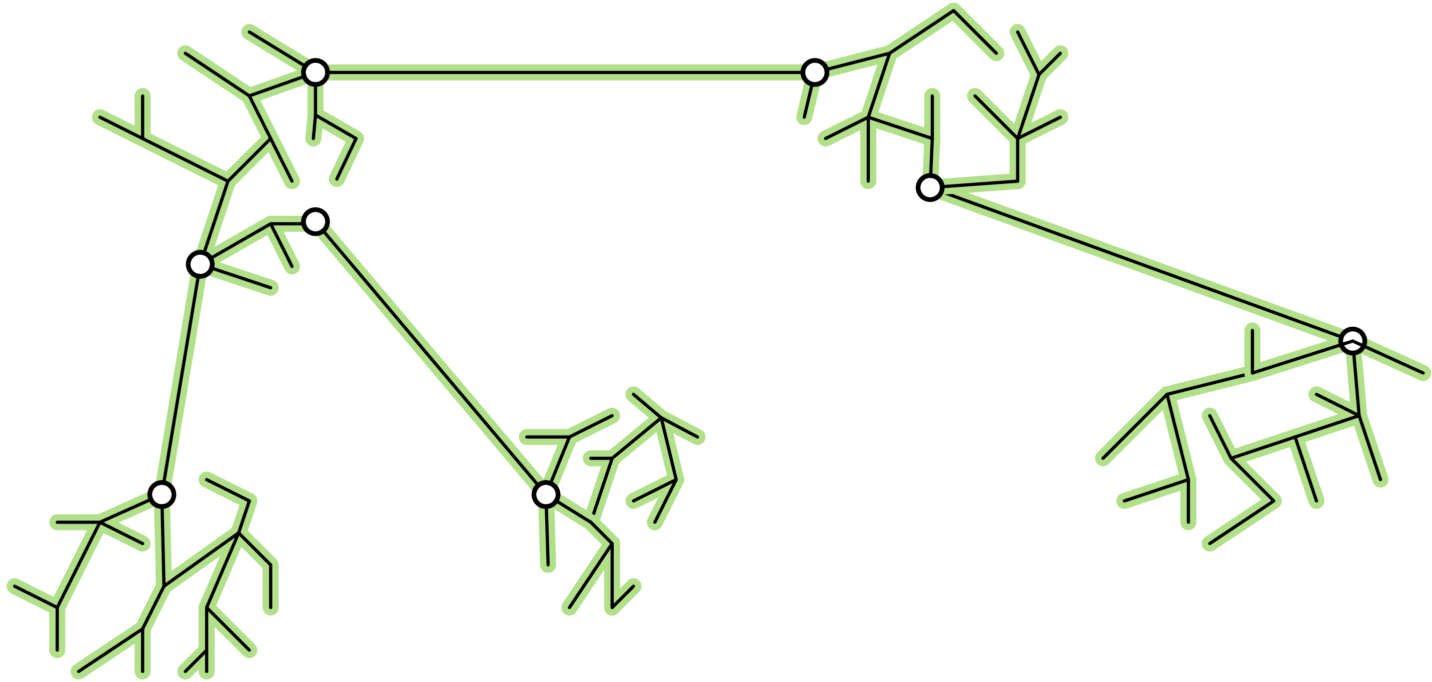
Approximation Algorithms

Lecture 10:
MINIMUM-DEGREE SPANNING TREE
via Local Search

Part III:
Lower Bound

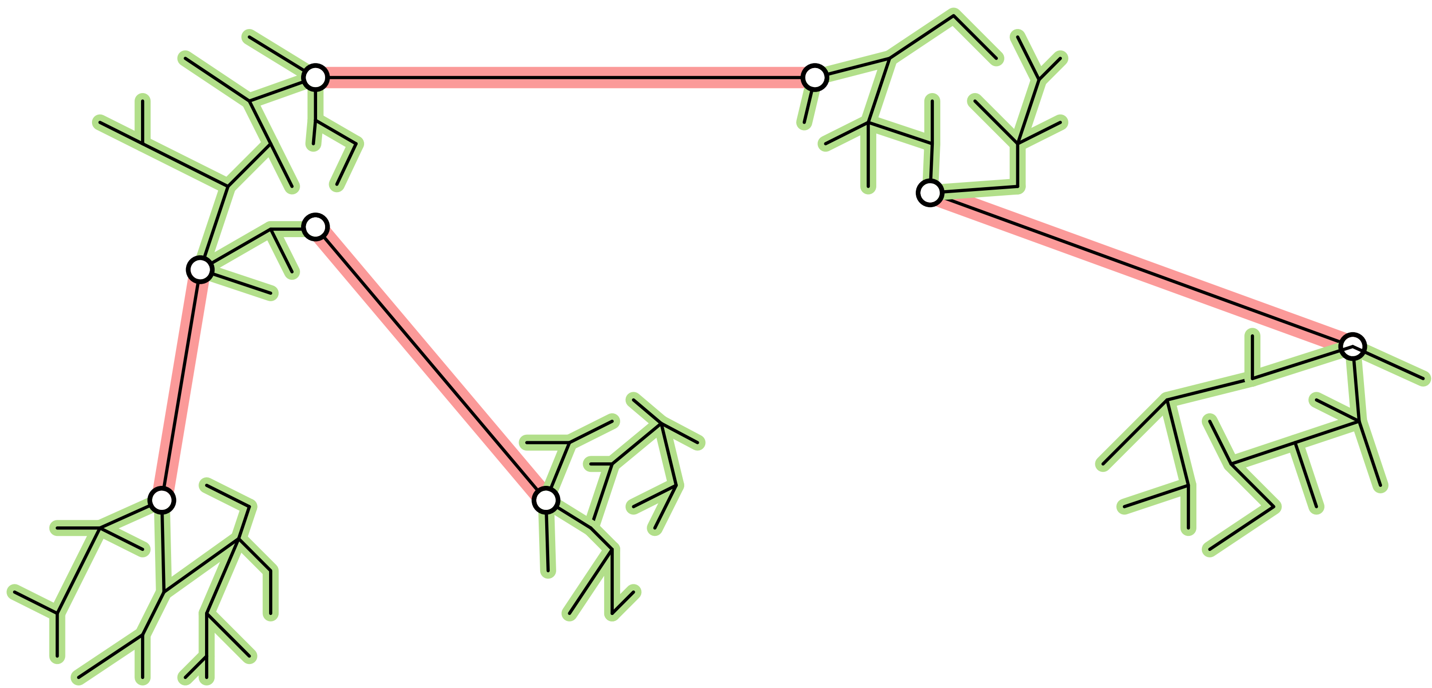
Decomposition

spanning
tree *T*



Decomposition

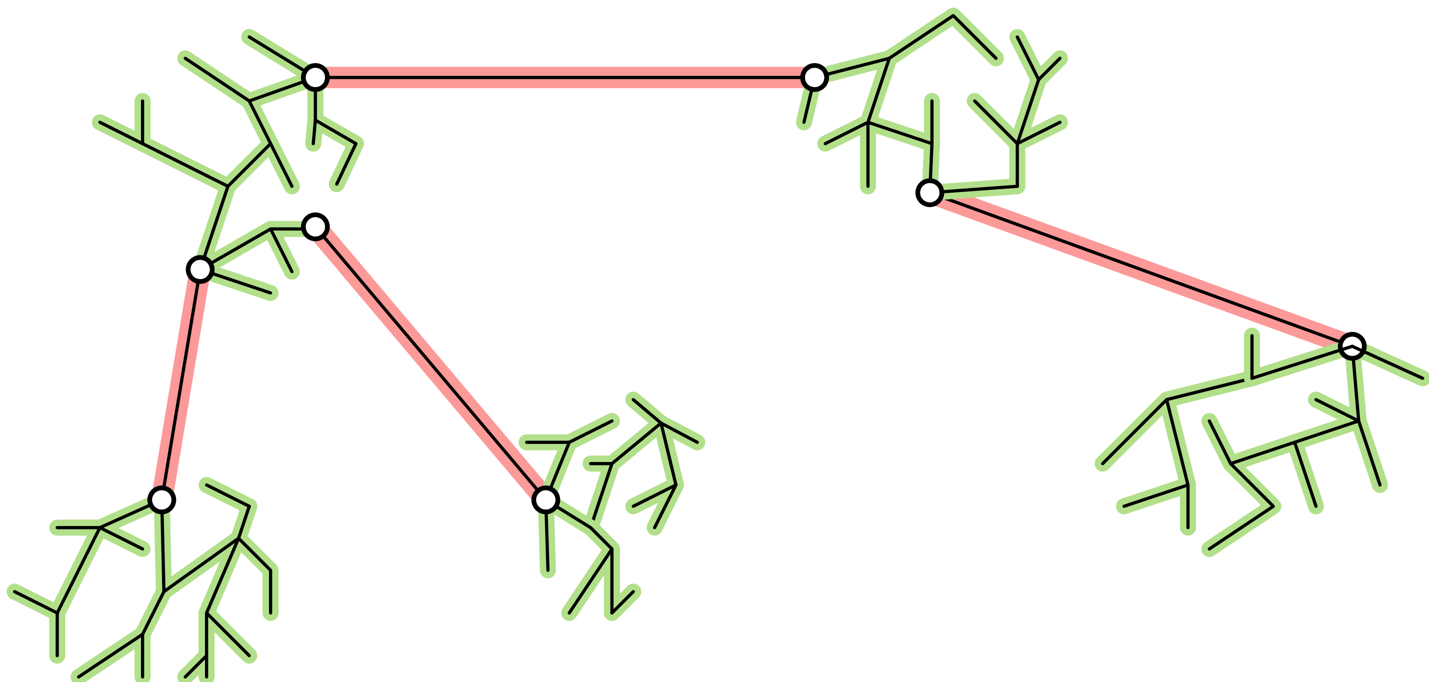
spanning tree T



Decomposition

- Removing k edges decomposes T into $k+1$ components.

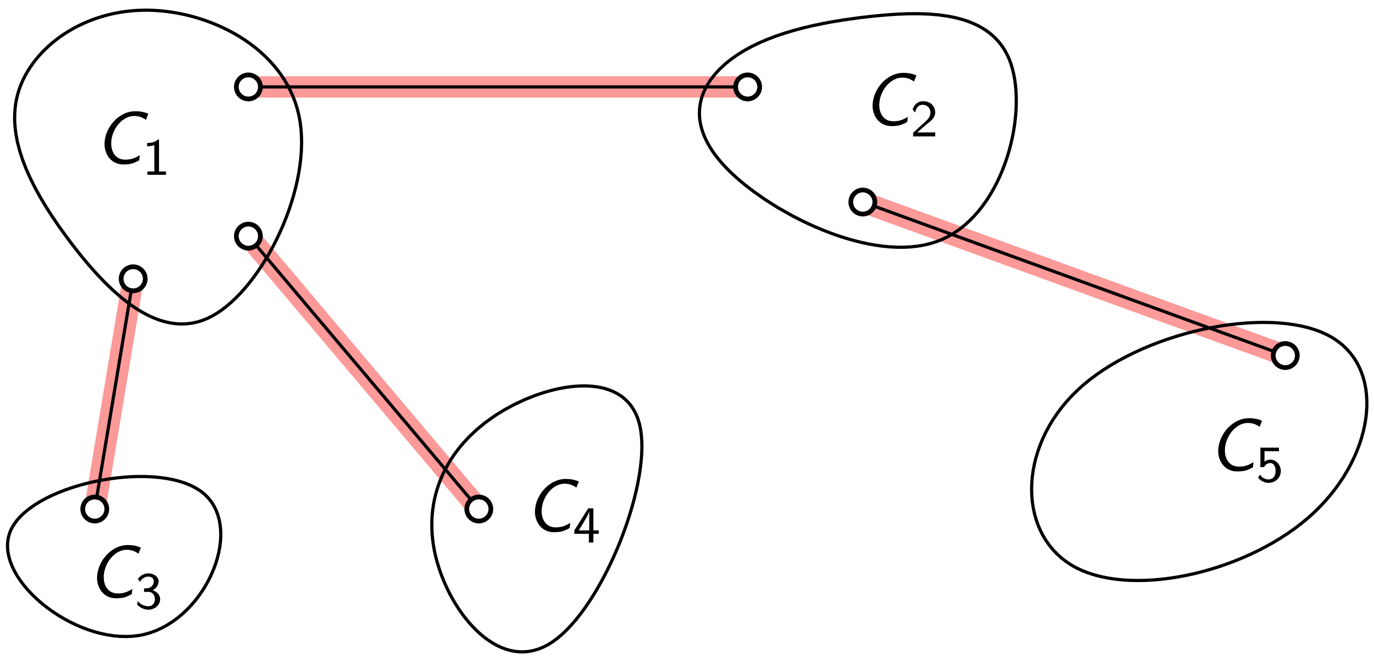
spanning tree T



Decomposition

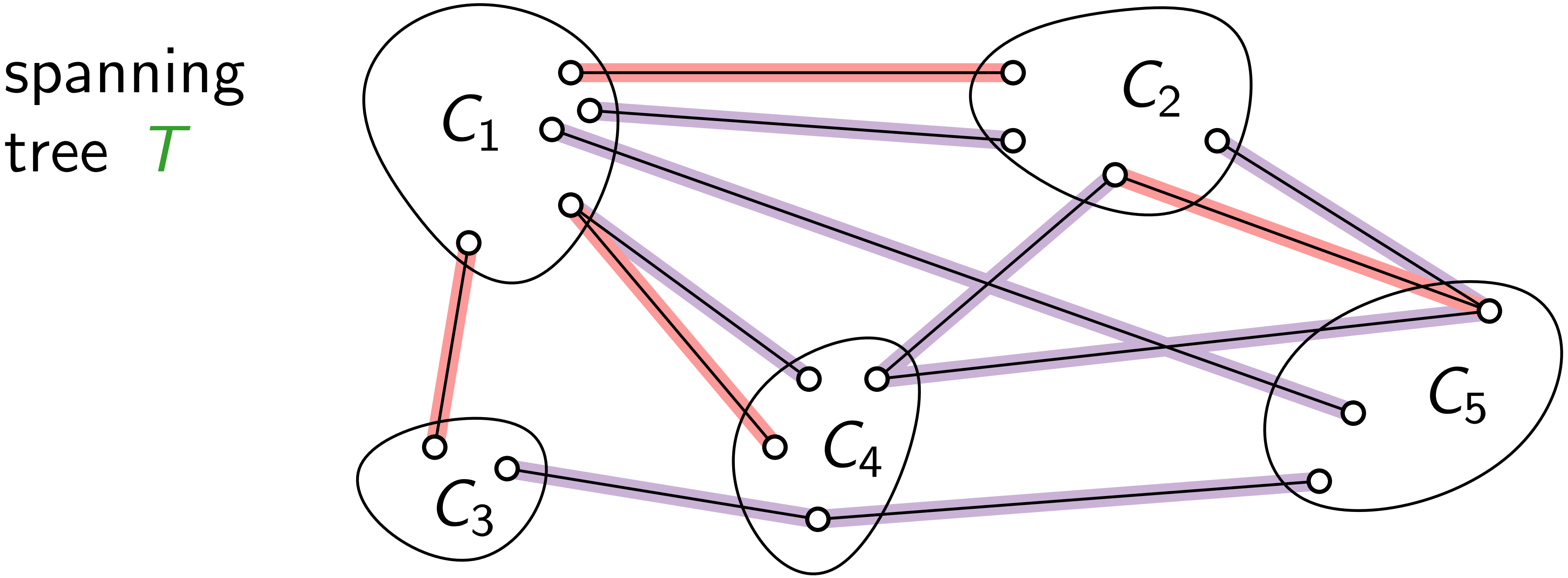
- Removing k edges decomposes T into $k+1$ components.

spanning tree T



Decomposition

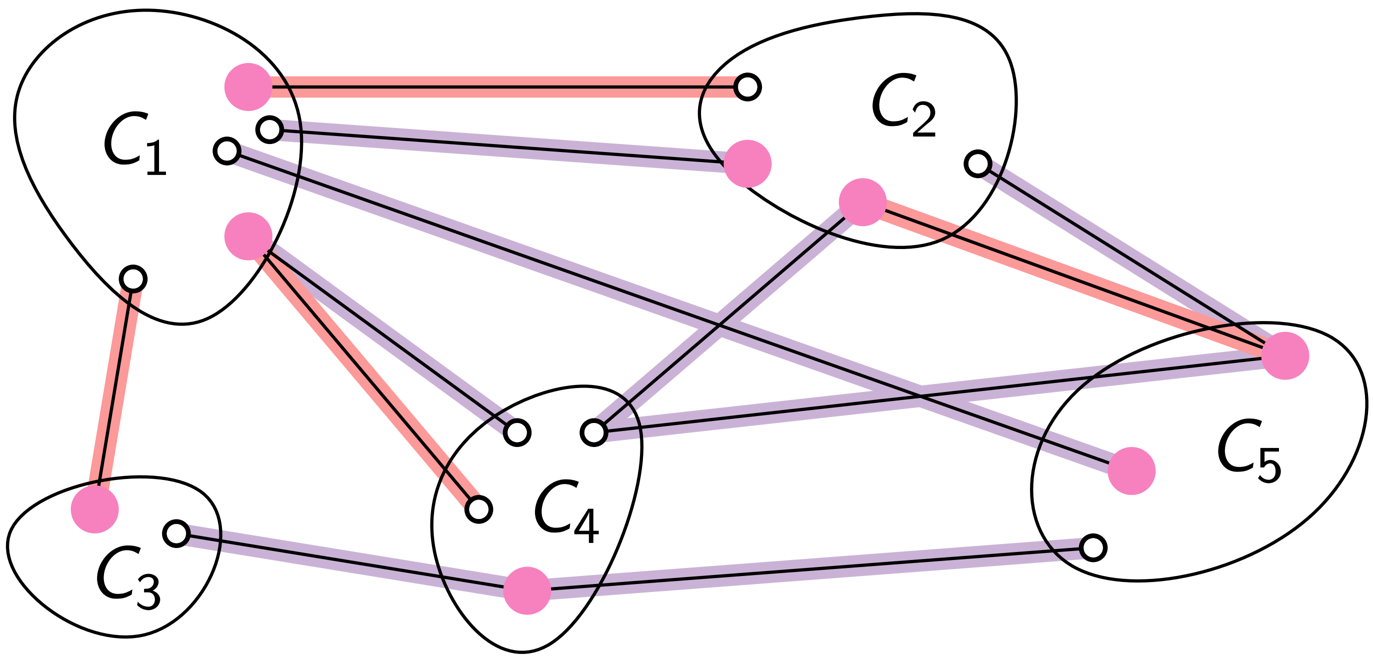
- Removing k edges decomposes T into $k+1$ components.
- $E' = \{\text{edges in } G \text{ between different components } C_i \neq C_j\}$.



Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

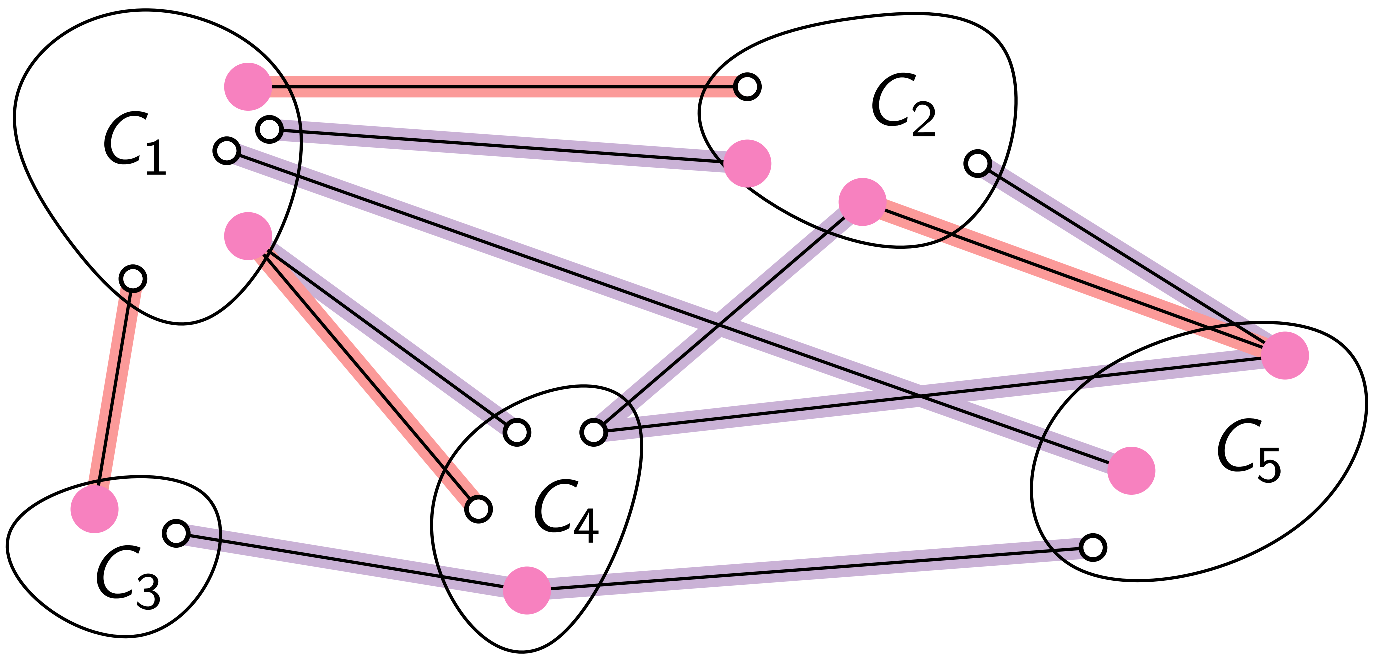
spanning tree T



Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

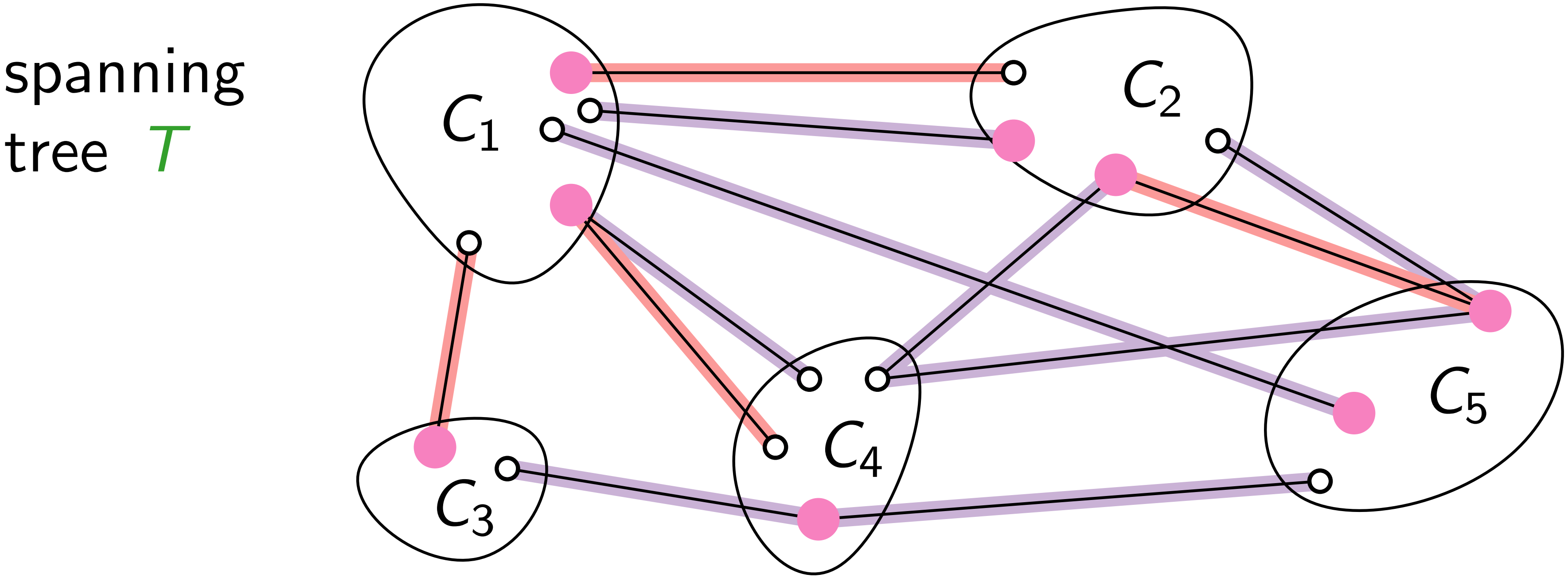
spanning tree T



- For any spanning tree T', $|E(T') \cap E'| \geq$

Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{\text{edges in } G \text{ between different components } C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

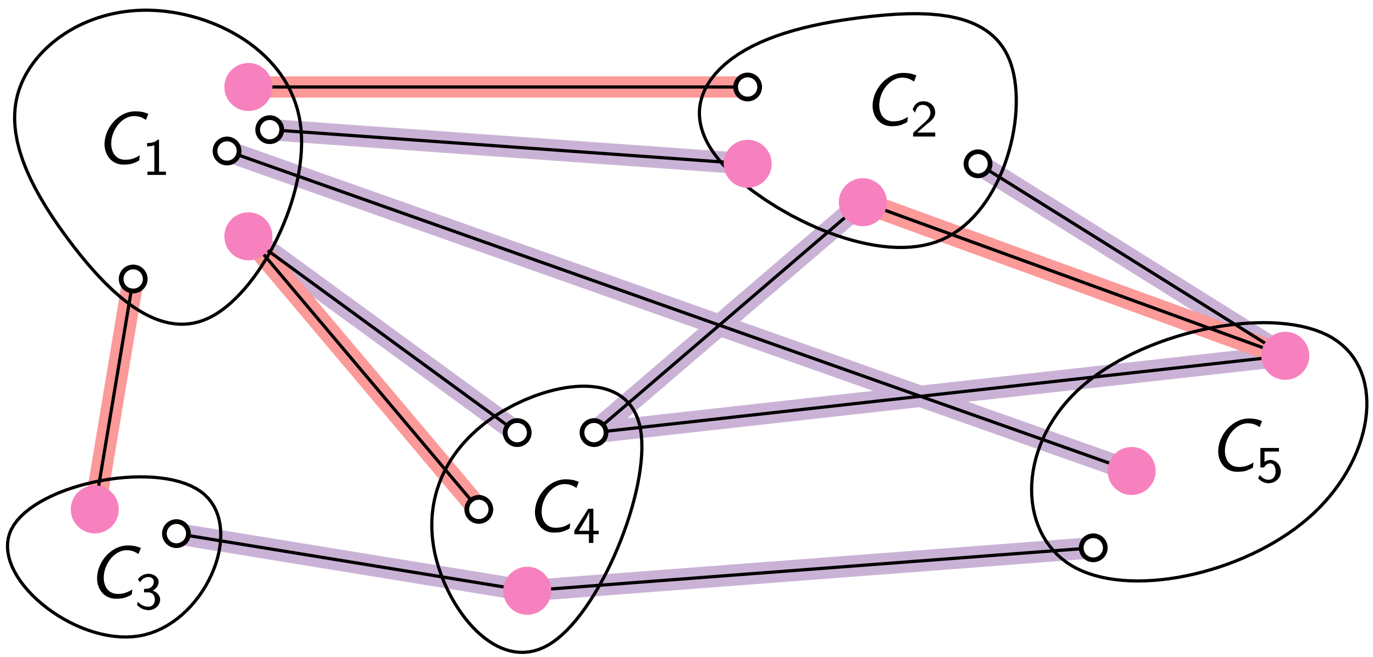


- For any spanning tree T', $|E(T') \cap E'| \geq k$,

Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

spanning tree T

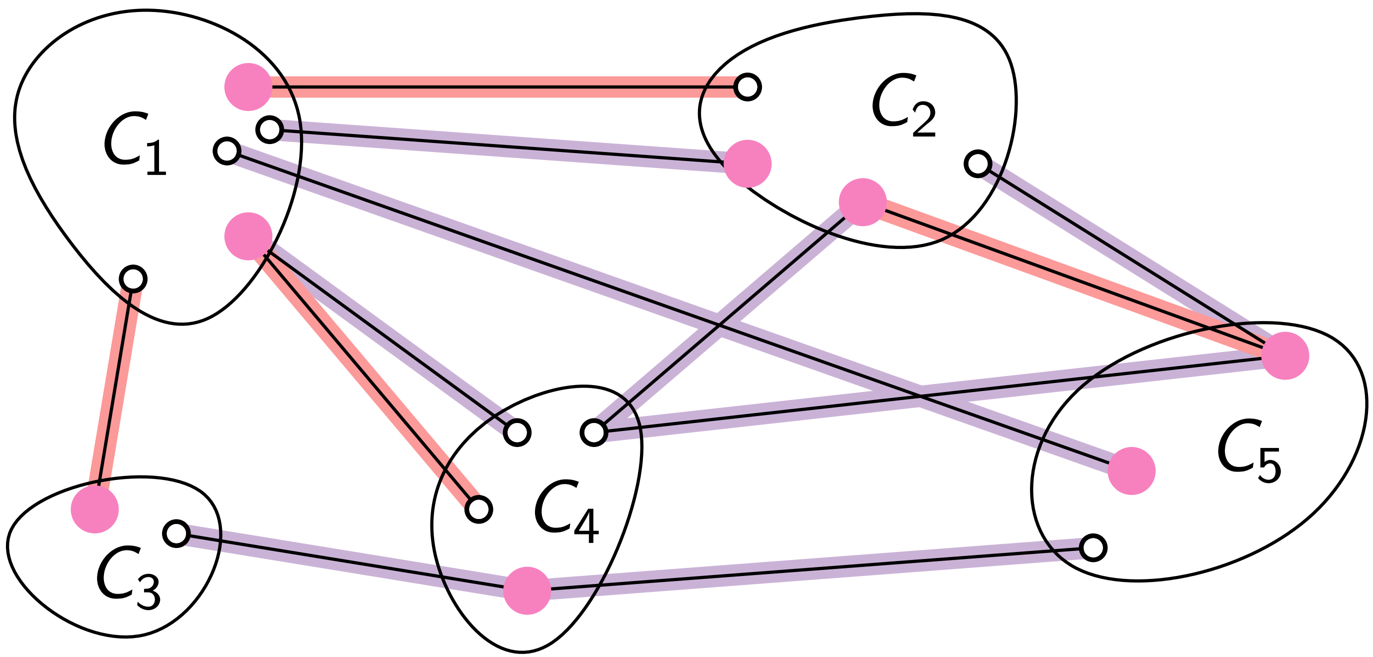


- For any spanning tree T', $|E(T') \cap E'| \geq k$,
- $\sum_{v \in S} \deg_{T'}(v) \geq$

Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

spanning tree T

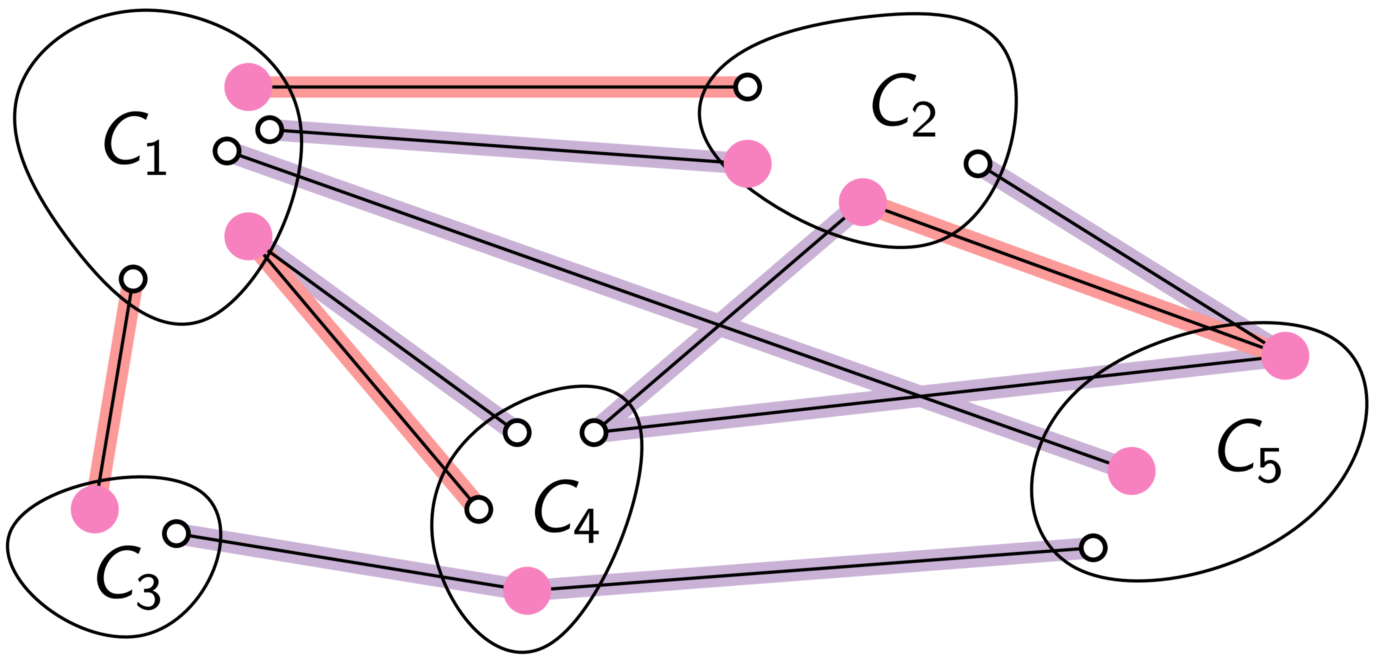


- For any spanning tree T', $|E(T') \cap E'| \geq k$,
- $\sum_{v \in S} \deg_{T'}(v) \geq k$, and (Obs. 2)

Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

spanning tree T

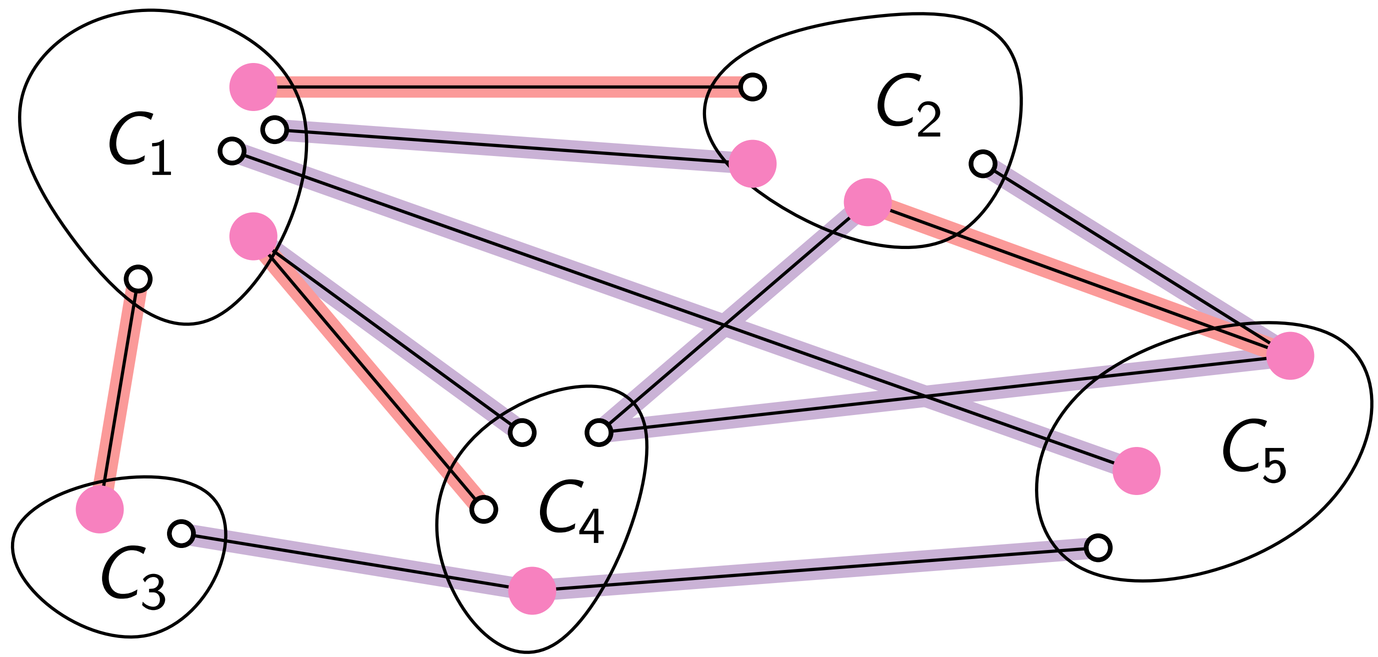


- For any spanning tree T', $|E(T') \cap E'| \geq k$,
- $\sum_{v \in S} \deg_{T'}(v) \geq k$, and $\Delta(T') \geq$
  (Obs. 2)

Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

spanning tree T



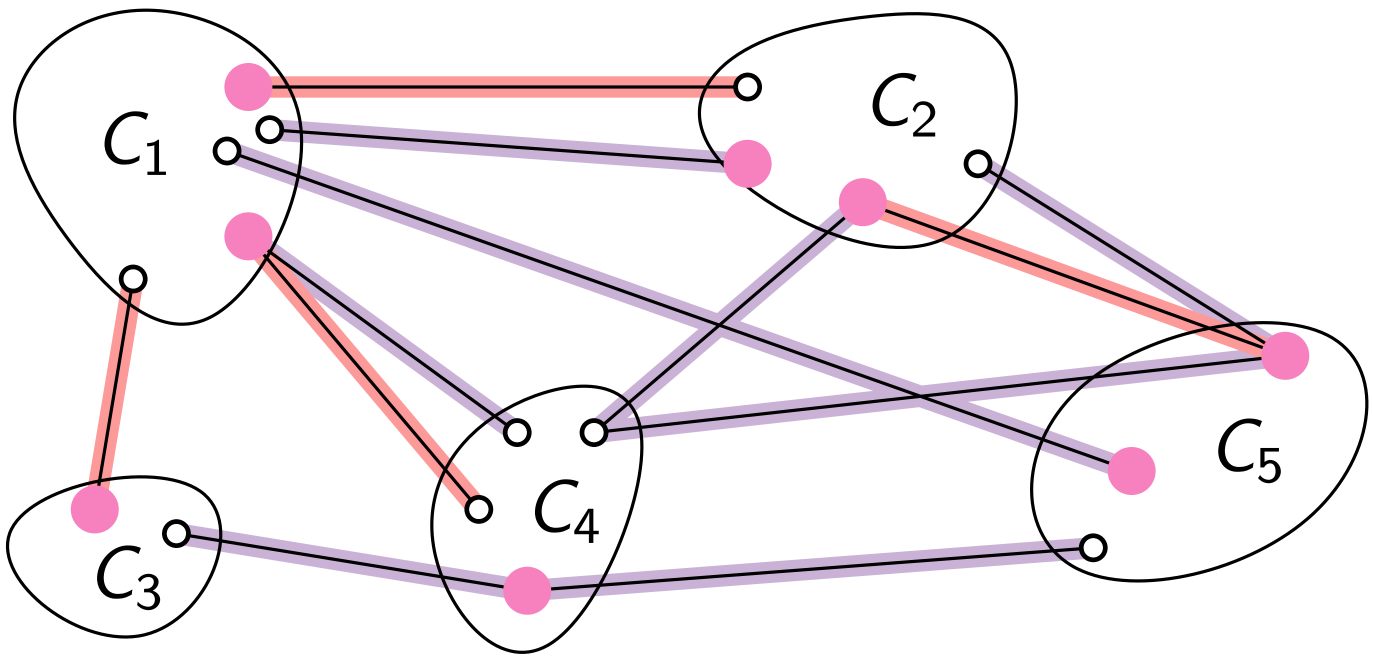
- For any spanning tree T', $|E(T') \cap E'| \geq k$,
- $\sum_{v \in S} \deg_{T'}(v) \geq k$, and $\Delta(T') \geq k/|S|$.

(Obs. 2)

Decomposition

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

spanning tree T

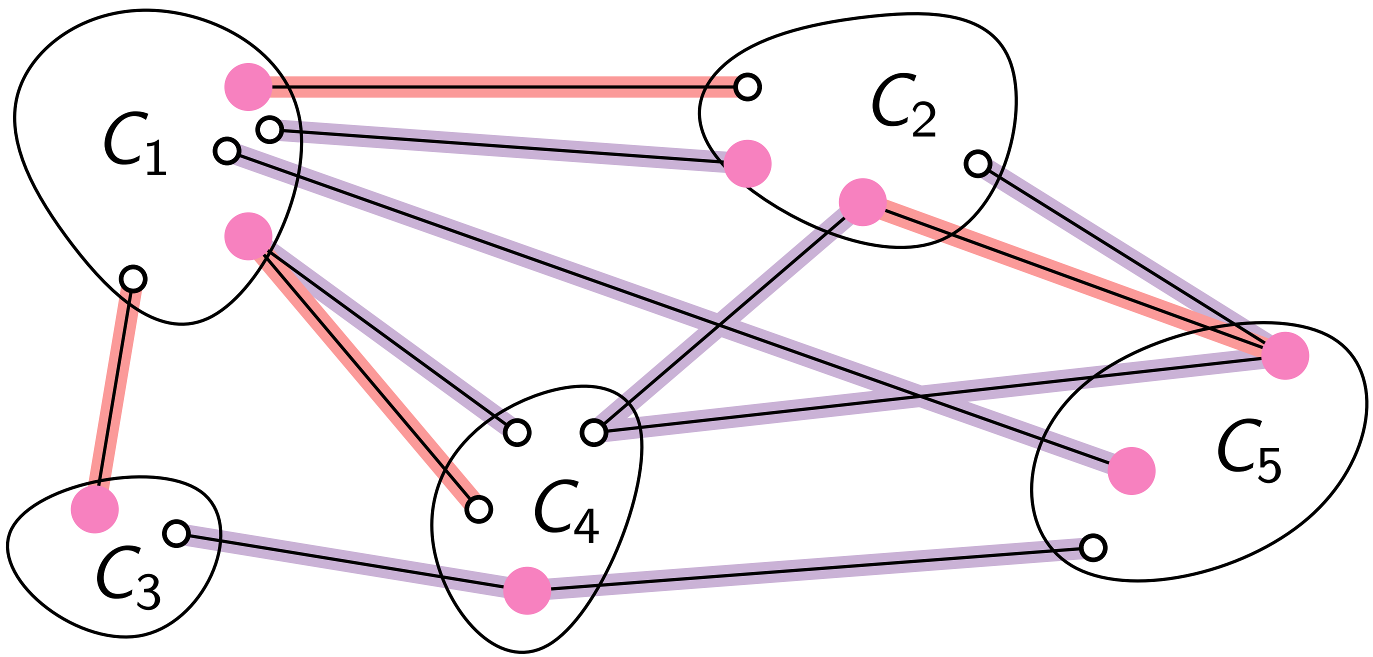


- For any spanning tree T', $|E(T') \cap E'| \geq k$,
- $\sum_{v \in S} \deg_{T'}(v) \geq k$, and $\Delta(T') \geq k/|S|$.
  (Obs. 2)
- Consider the optimal spanning tree T^*.

Decomposition $\Rightarrow$ Lower Bound for OPT

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

spanning tree T



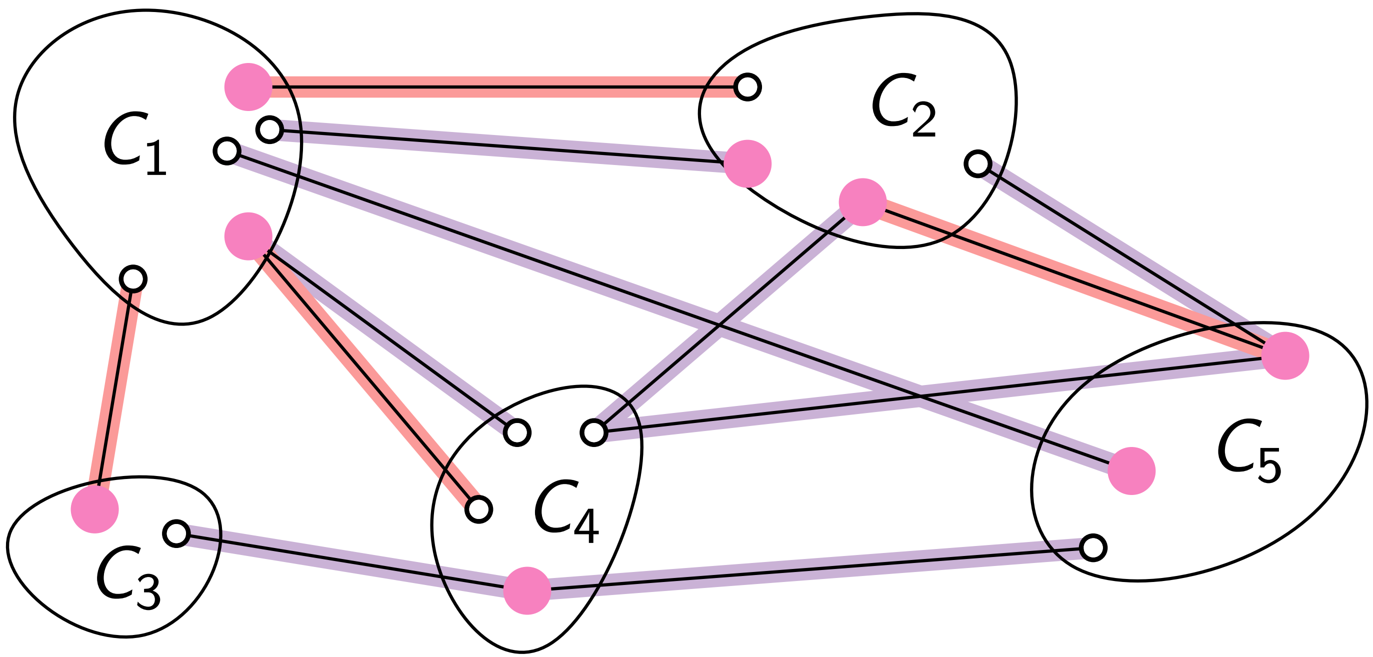
- For any spanning tree T', $|E(T') \cap E'| \geq k$,
- $\sum_{v \in S} \deg_{T'}(v) \geq k$, and $\Delta(T') \geq k/|S|$. (Obs. 2)
- Consider the optimal spanning tree T^*.

Lemma 1.
$\Rightarrow$ OPT $\geq$

Decomposition $\Rightarrow$ Lower Bound for OPT

- Removing k edges decomposes T into $k+1$ components.
- $E' = \{$edges in G between different components $C_i \neq C_j\}$.
- $S :=$ vertex cover of E'.

spanning tree T



- For any spanning tree T', $|E(T') \cap E'| \geq k$,
- $\sum_{v \in S} \deg_{T'}(v) \geq k$, and $\Delta(T') \geq k/|S|$. (Obs. 2)
- Consider the optimal spanning tree T^*.

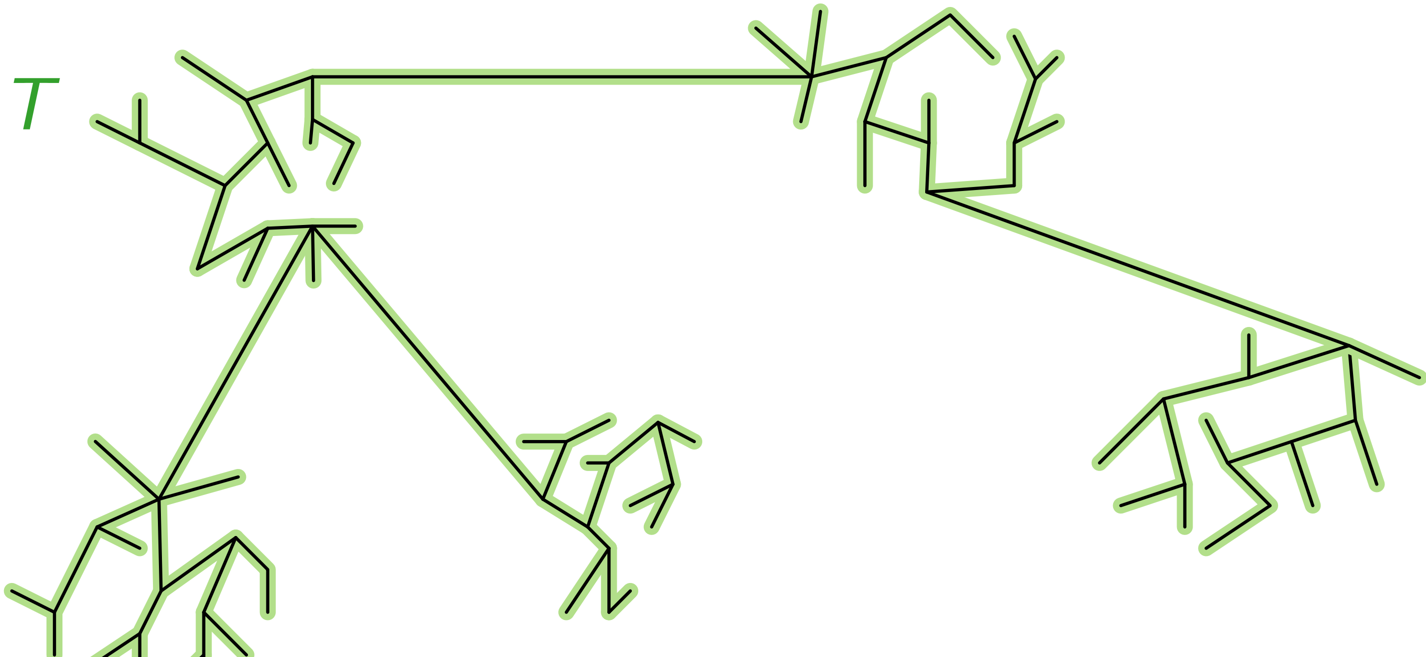
Lemma 1.
$\Rightarrow \mathrm{OPT} \geq k/|S|$

Approximation Algorithms

Lecture 10:
Minimum-Degree Spanning Tree
via Local Search

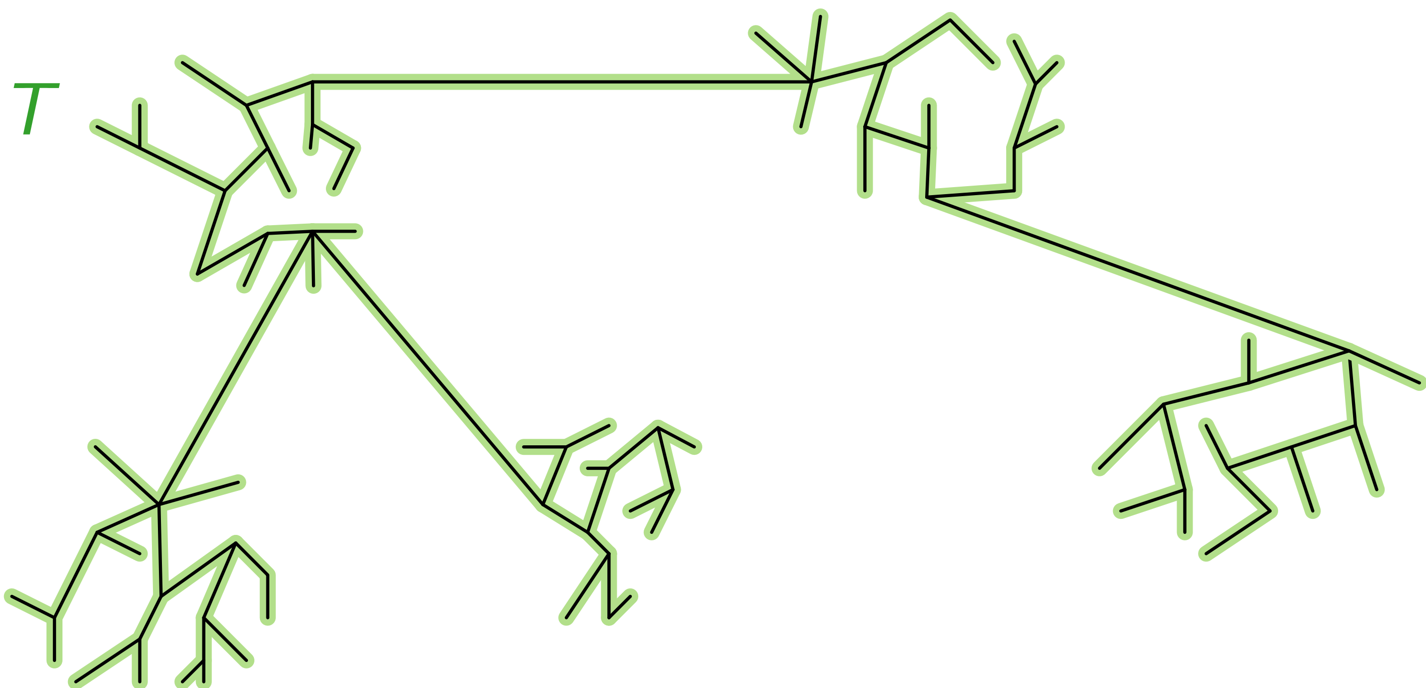
Part IV:
Structure of a Decomposition

Structure of a Decomposition



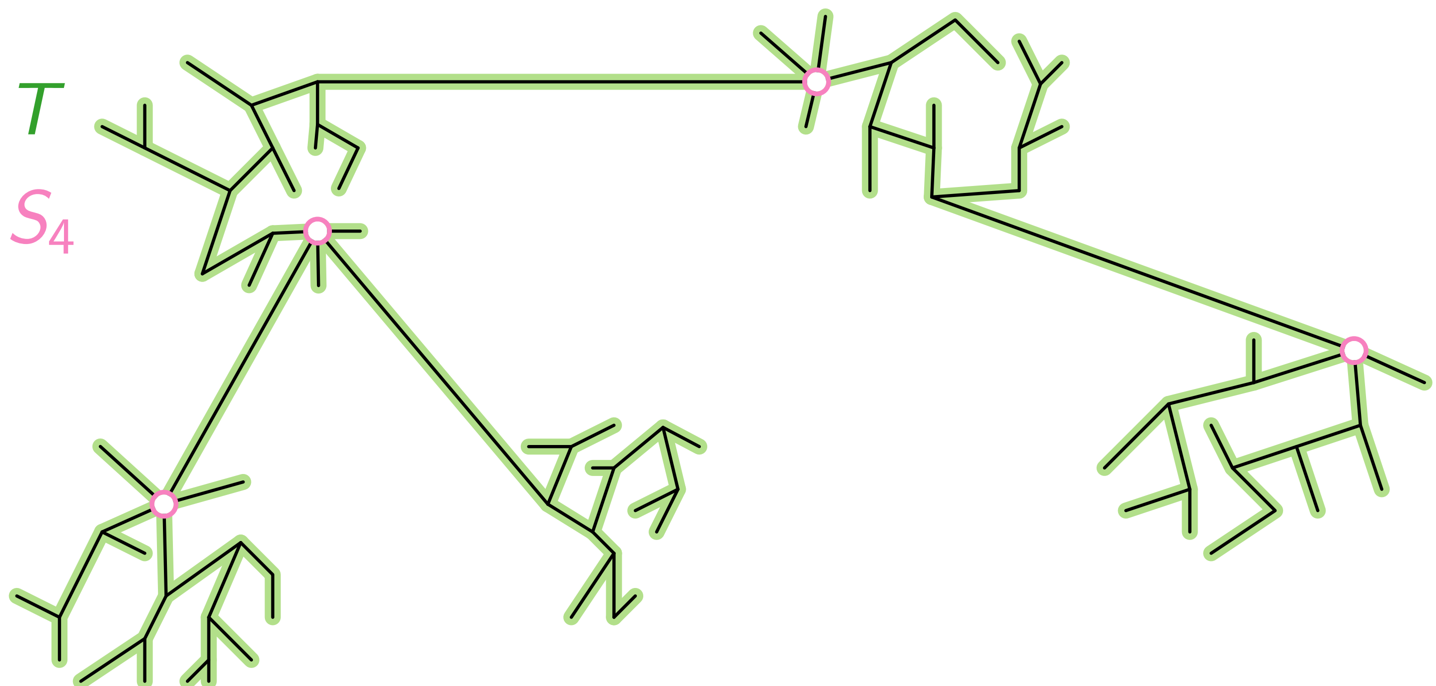
Structure of a Decomposition

Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.



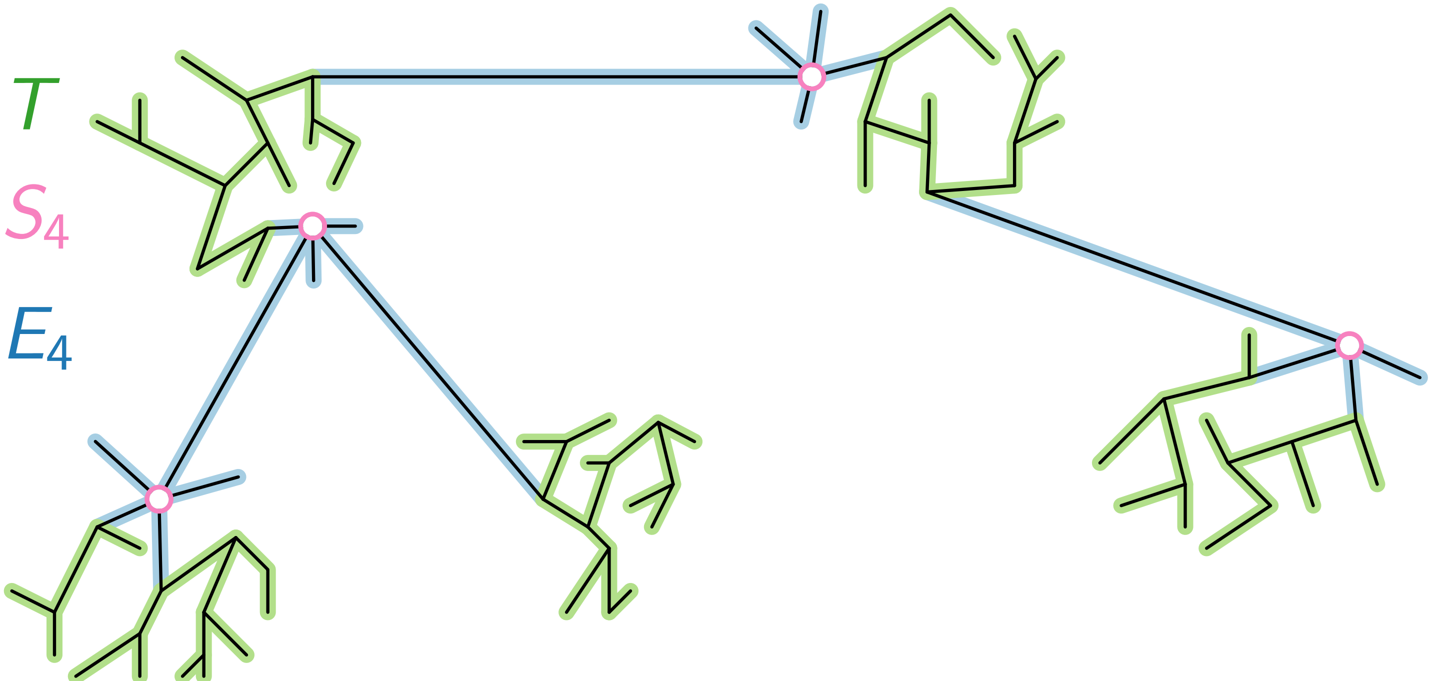
Structure of a Decomposition

Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.



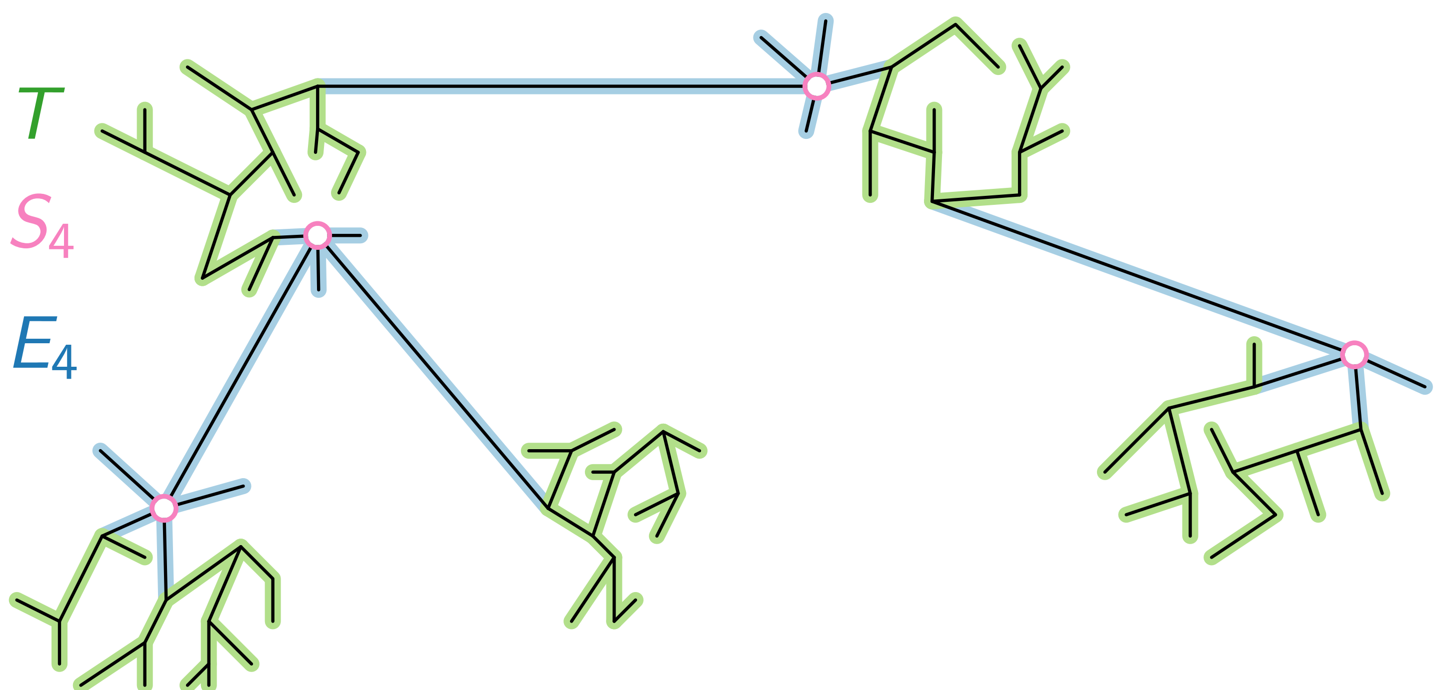
Structure of a Decomposition

Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.



Structure of a Decomposition

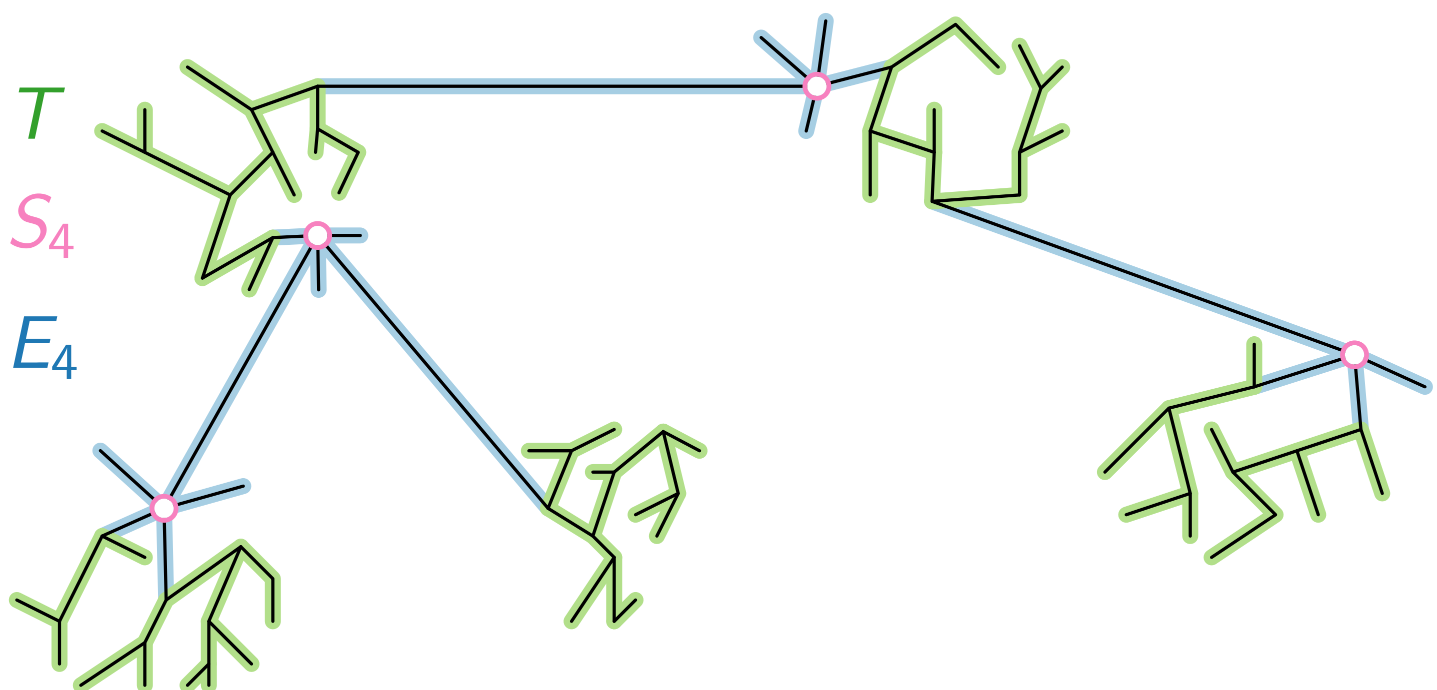
Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.



Structure of a Decomposition

$\Rightarrow S_1 \supseteq S_2 \supseteq \dots$
$\Rightarrow S_1 = V(G)$

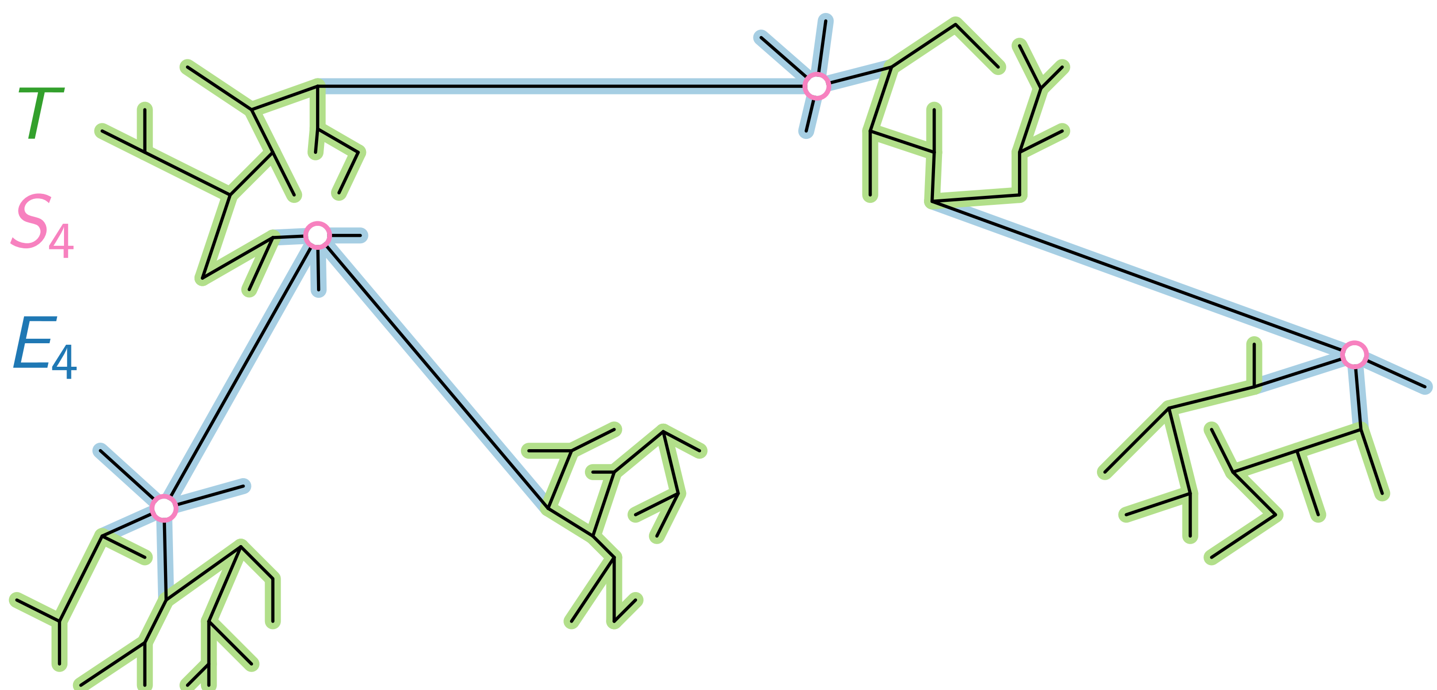
Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.



Structure of a Decomposition

$\Rightarrow S_1 \supseteq S_2 \supseteq \ldots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

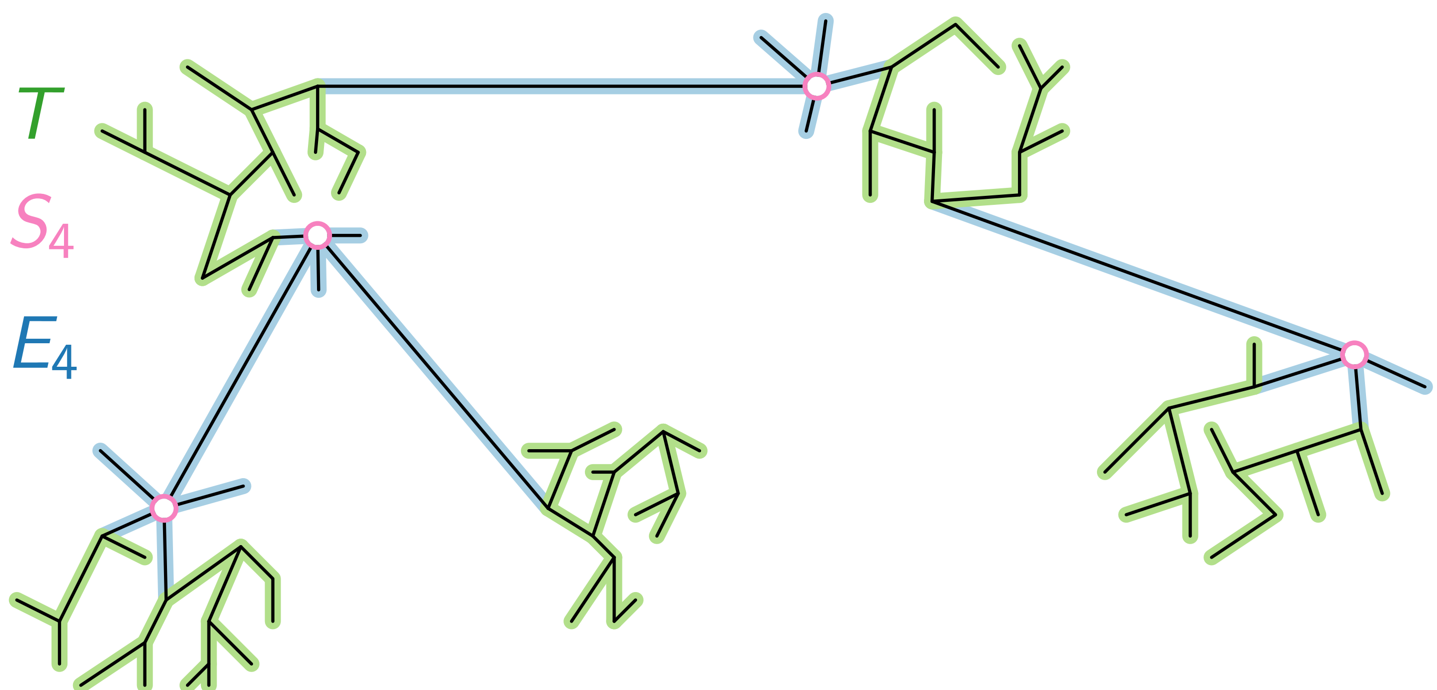


Structure of a Decomposition

$\Rightarrow S_1 \supseteq S_2 \supseteq \dots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

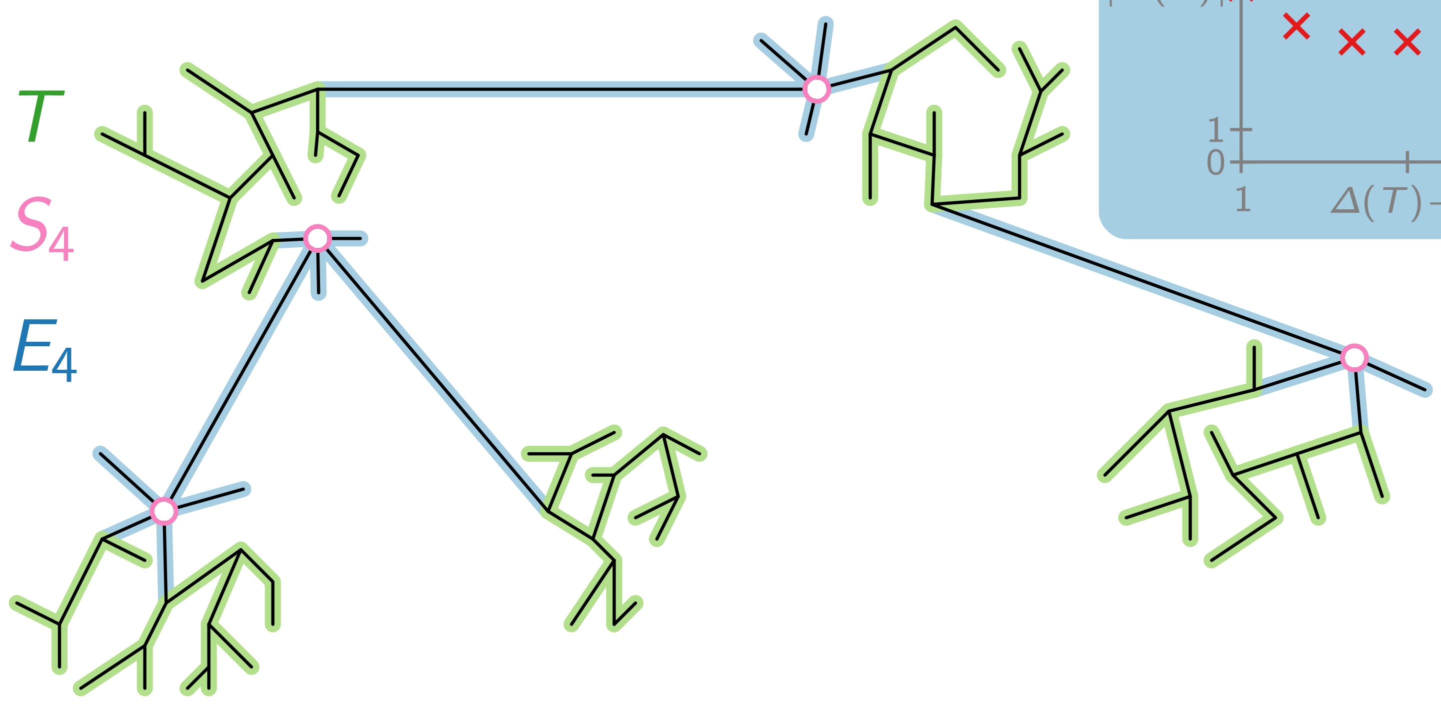


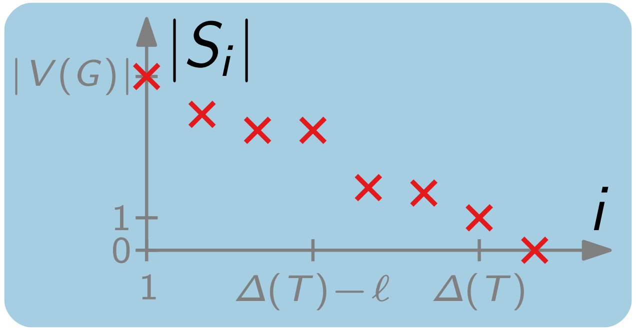
Structure of a Decomposition

$\Rightarrow S_1 \supseteq S_2 \supseteq \ldots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.





Structure of a Decomposition

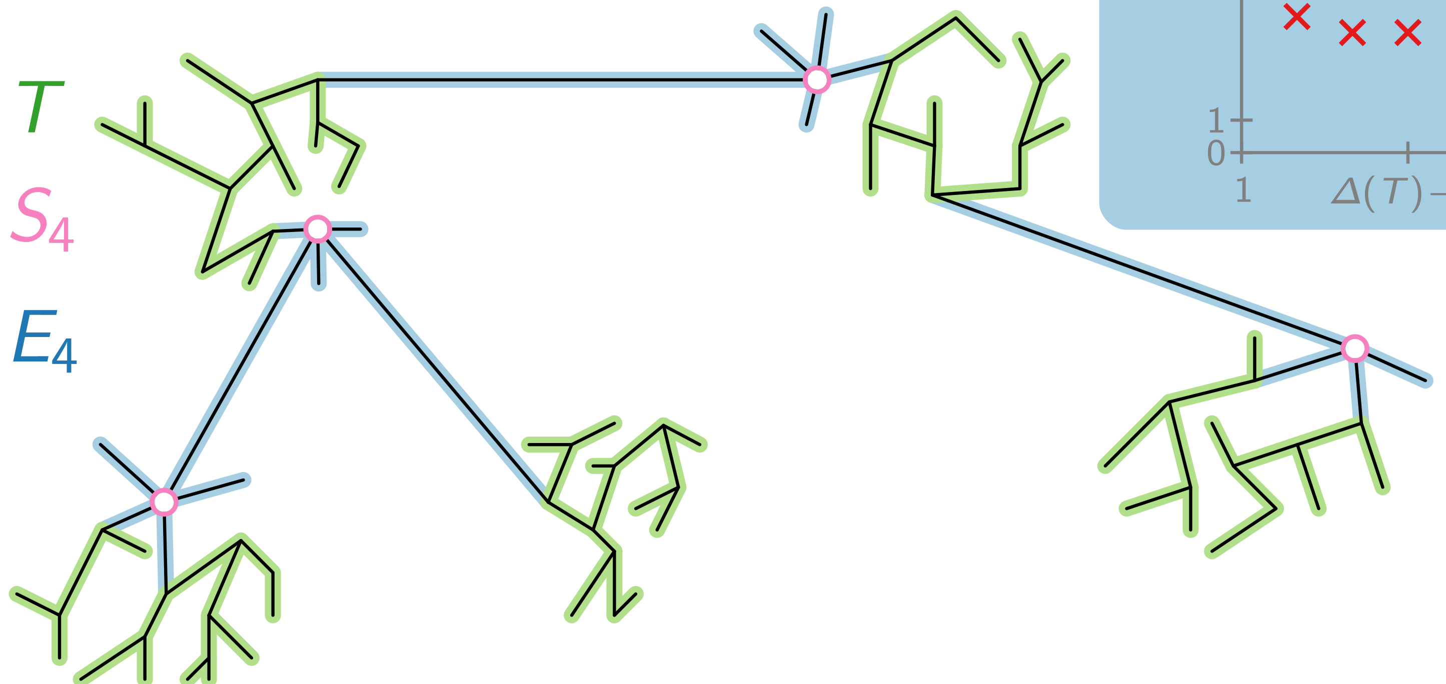
$\Rightarrow S_1 \supseteq S_2 \supseteq \dots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

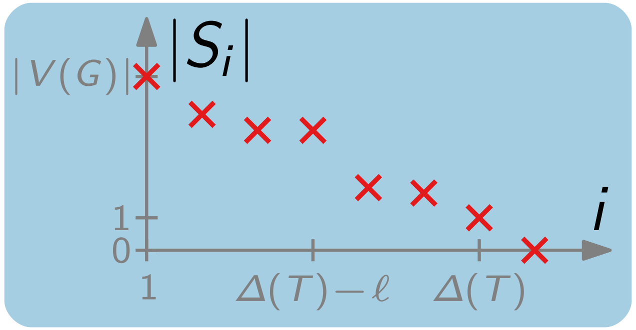
Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Proof. $|S_{\Delta(T)-\ell}| > 2^\ell |S_{\Delta(T)}|$

Otherwise





Structure of a Decomposition

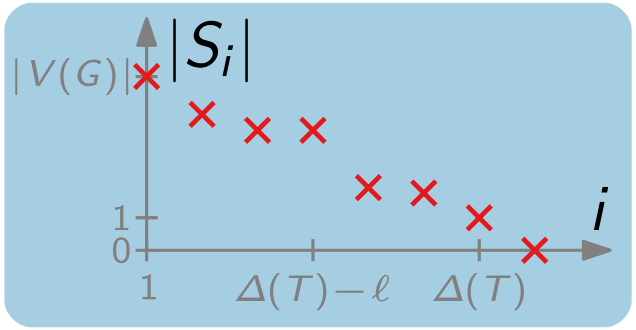
$\Rightarrow S_1 \supseteq S_2 \supseteq \dots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

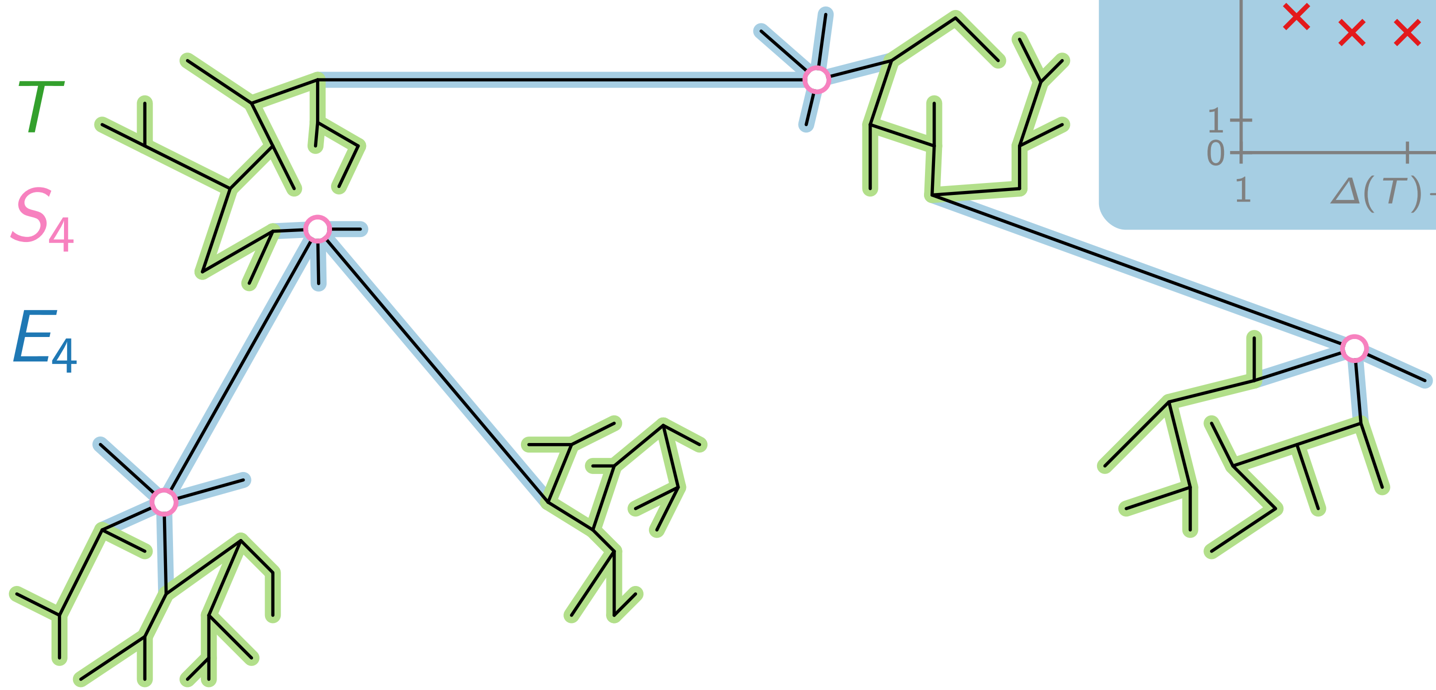
Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Proof. Otherwise $|S_{\Delta(T)-\ell}| > 2^{\ell}|S_{\Delta(T)}| = 2^{\lceil \log_2 n \rceil}|S_{\Delta(T)}| \geq$

$\ell = \lceil \log_2 n \rceil$





Structure of a Decomposition

$\Rightarrow S_1 \supseteq S_2 \supseteq \dots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

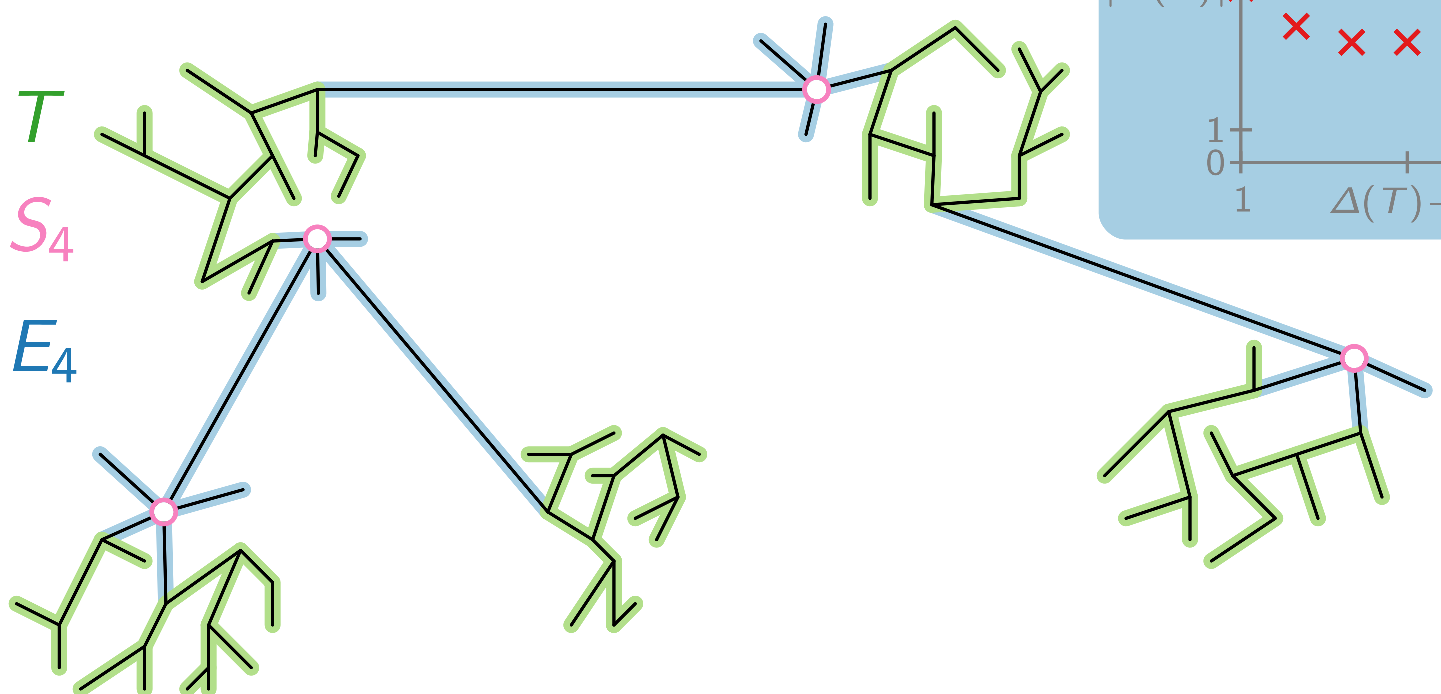
Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

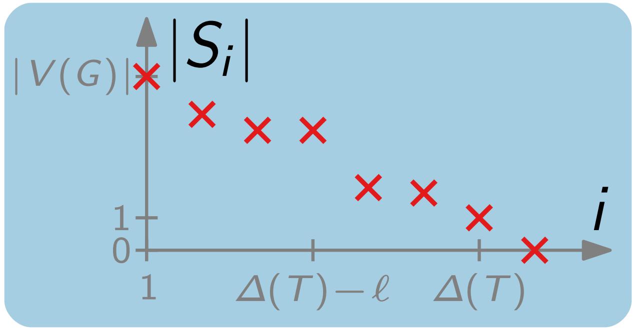
Lemma 2. $\exists i$ s.t. $\Delta(T)-\ell+1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Proof. $|S_{\Delta(T)-\ell}| > 2^\ell |S_{\Delta(T)}| = 2^{\lceil \log_2 n \rceil} |S_{\Delta(T)}| \geq n \cdot |S_{\Delta(T)}|$

$\ell = \lceil \log_2 n \rceil$

Otherwise





Structure of a Decomposition

$\Rightarrow S_1 \supseteq S_2 \supseteq \dots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

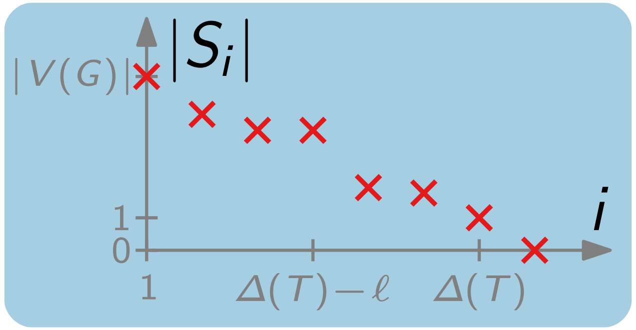
Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

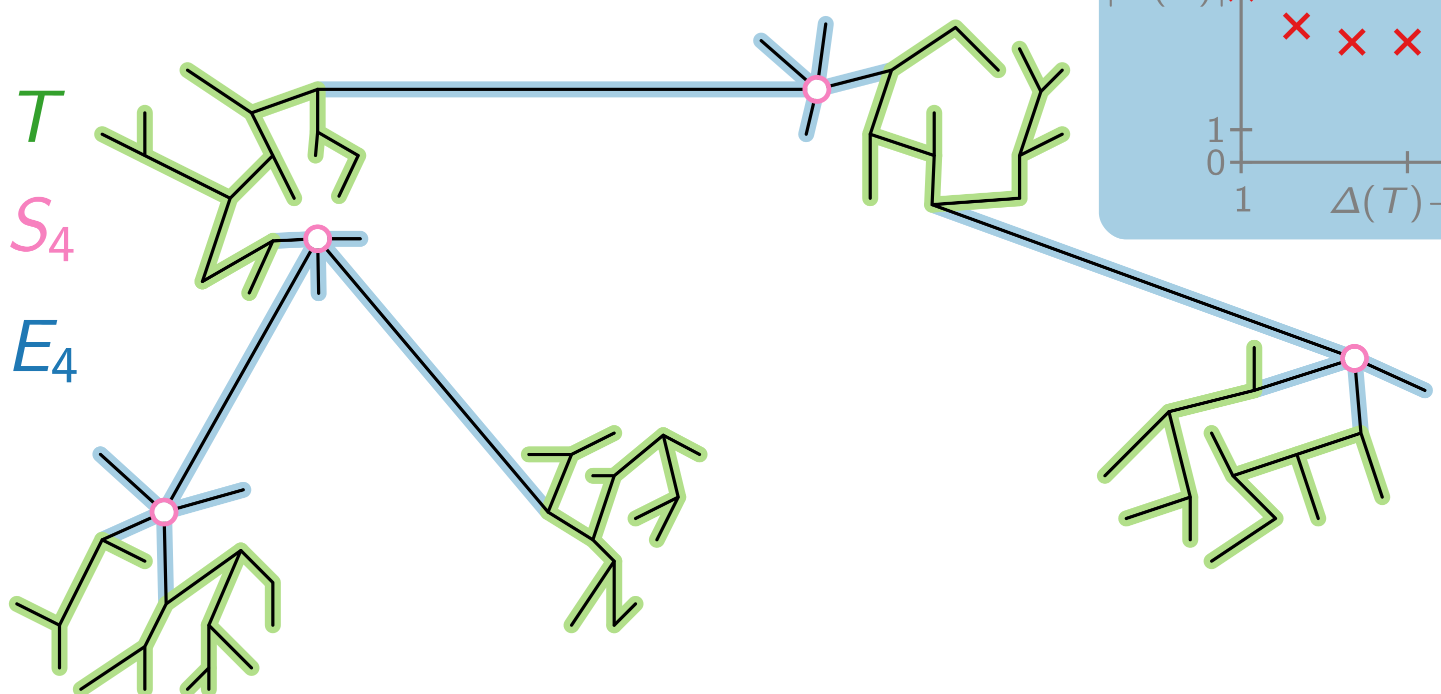
Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Proof. $|S_{\Delta(T)-\ell}| > 2^{\ell}|S_{\Delta(T)}| = 2^{\lceil \log_2 n \rceil}|S_{\Delta(T)}| \geq n \cdot |S_{\Delta(T)}|$ ↯

Otherwise

$\ell = \lceil \log_2 n \rceil$





Structure of a Decomposition

$\Rightarrow S_1 \supseteq S_2 \supseteq \dots$
$\Rightarrow S_1 = V(G)$
$\Rightarrow E_1 = E(T)$

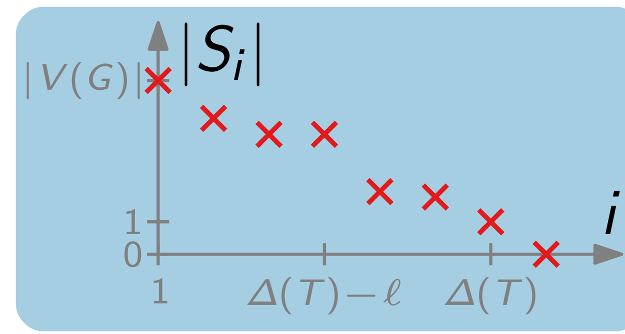
Let S_i be the set of vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the set of edges in T incident to S_i.

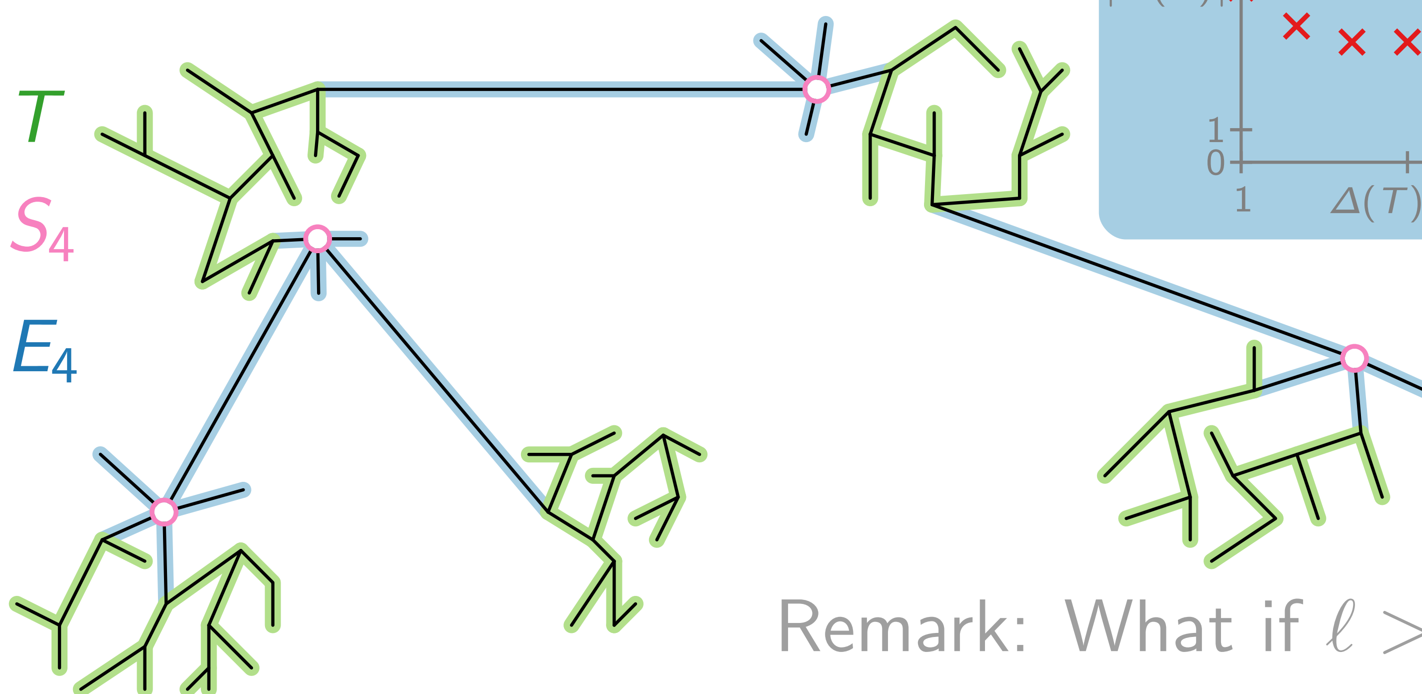
Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Proof. $|S_{\Delta(T)-\ell}| > 2^{\ell} |S_{\Delta(T)}| = 2^{\lceil \log_2 n \rceil} |S_{\Delta(T)}| \geq n \cdot |S_{\Delta(T)}|$ ↯

Otherwise

$\ell = \lceil \log_2 n \rceil$

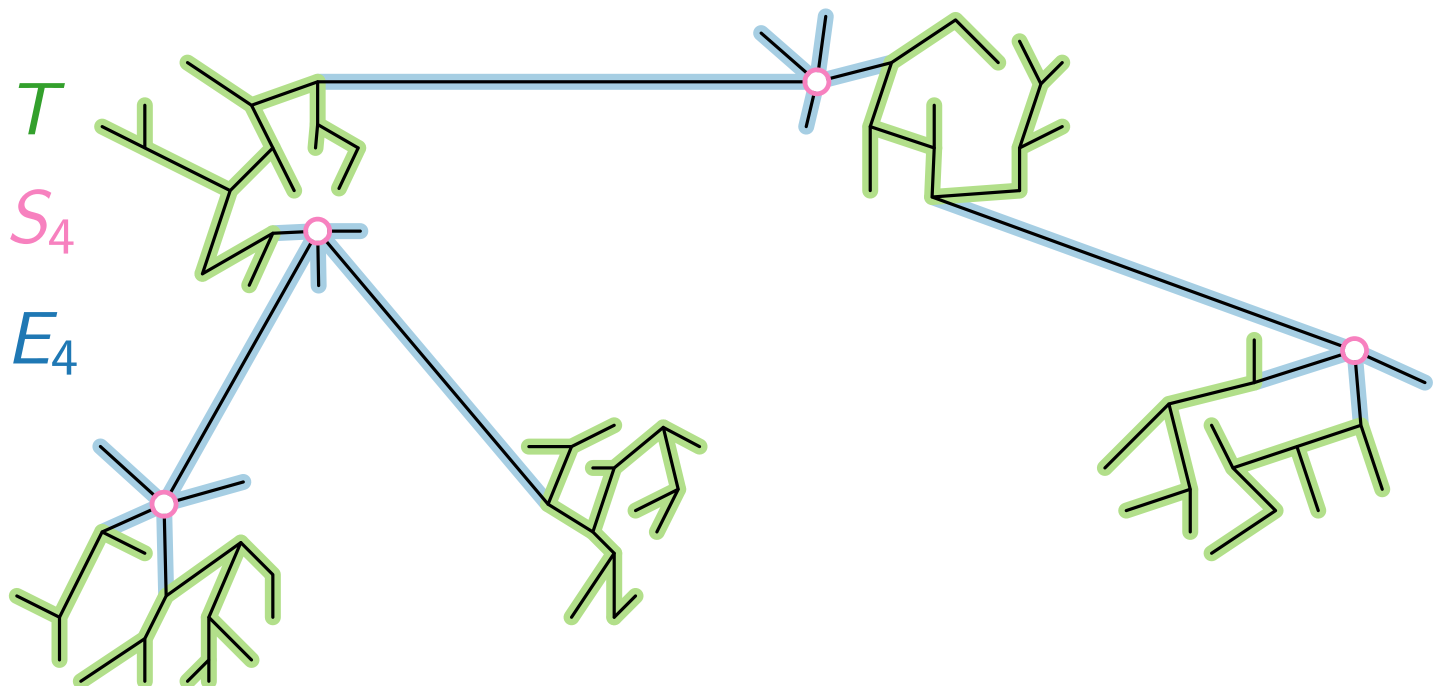




Remark: What if $\ell > \Delta(T)$?

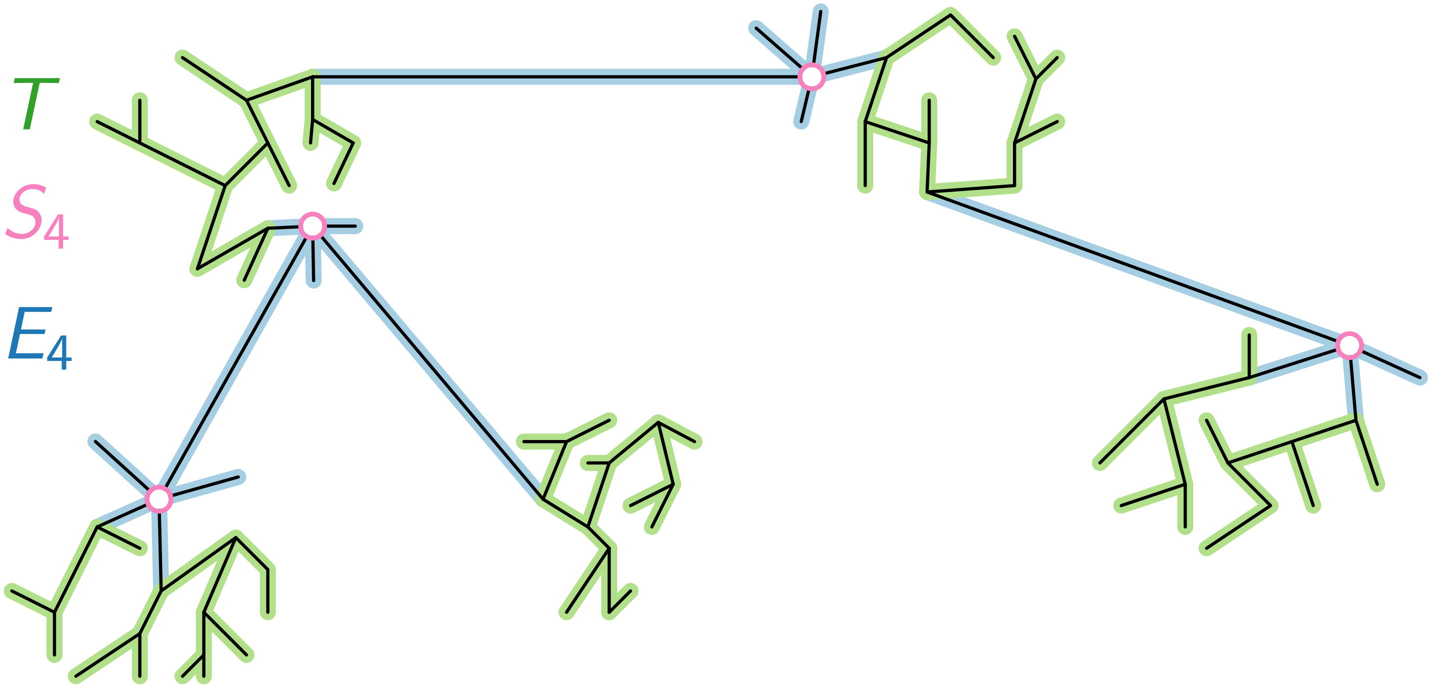
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:



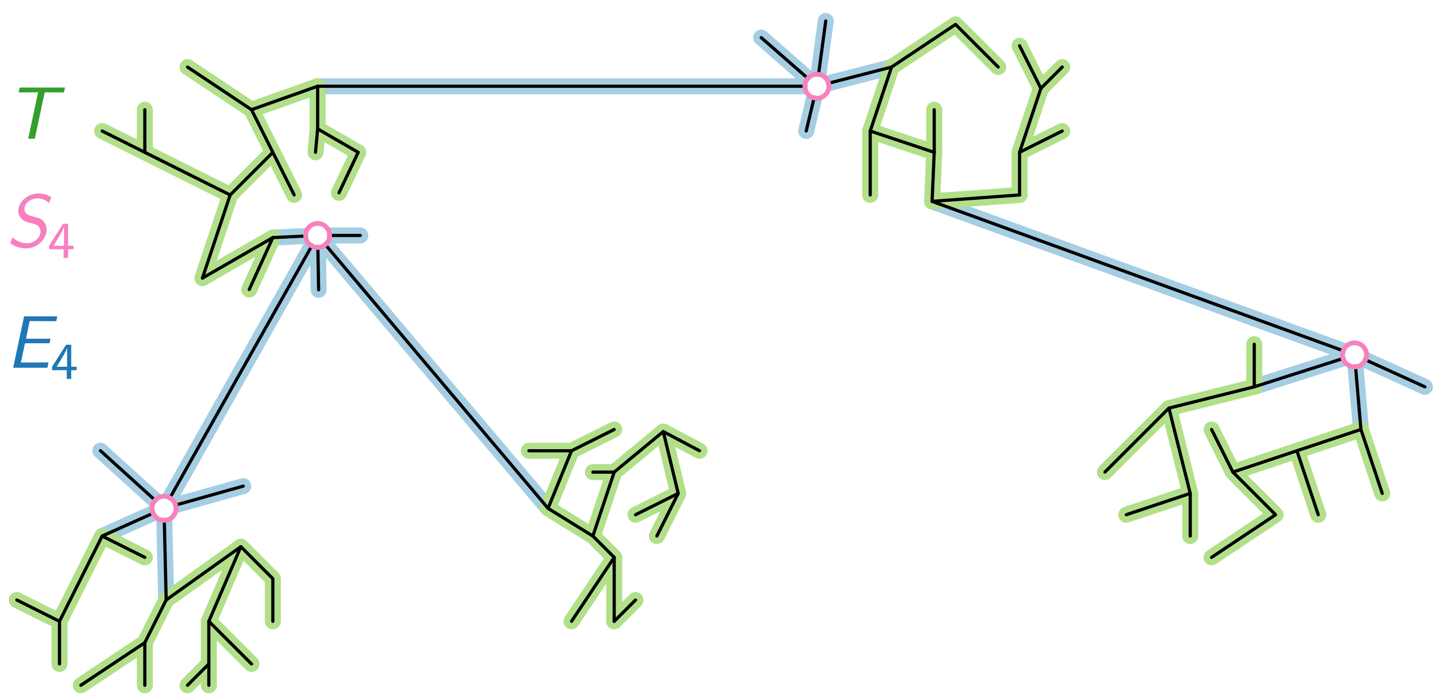
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,



Structure of a Decomposition

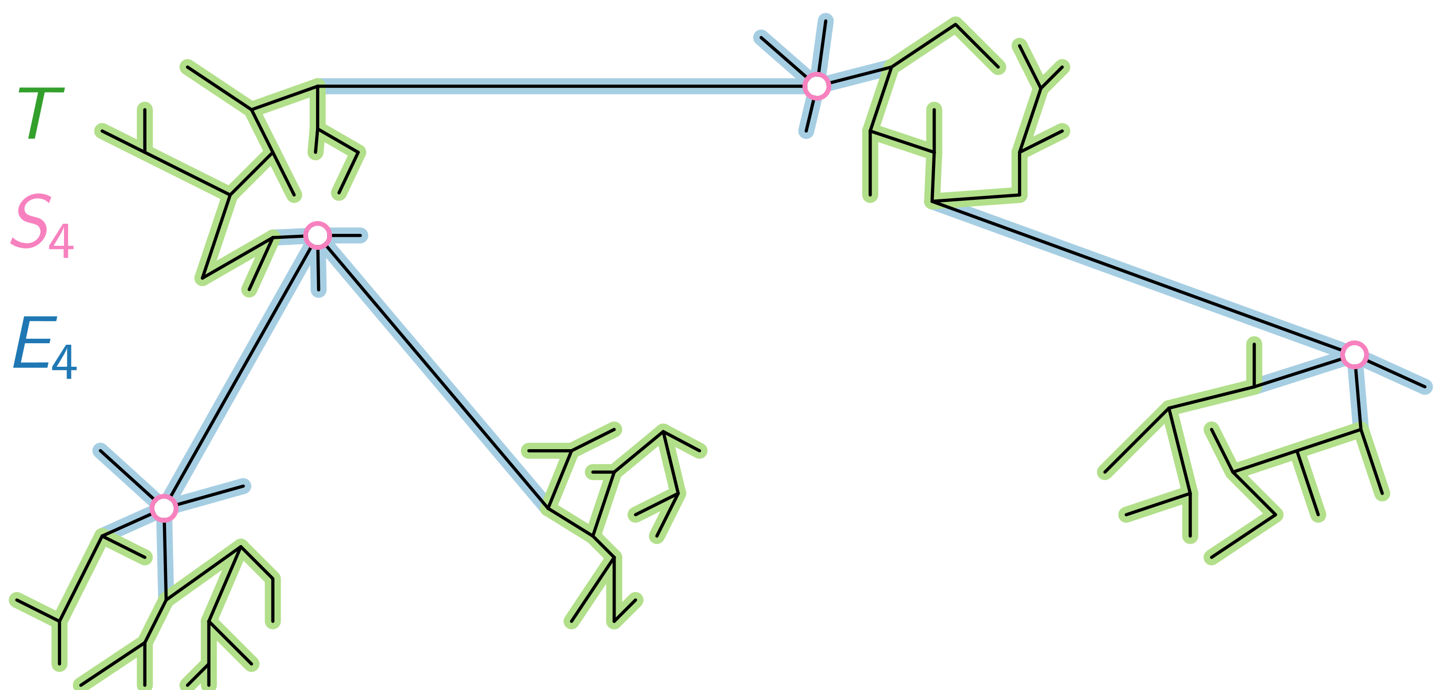
Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.



Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

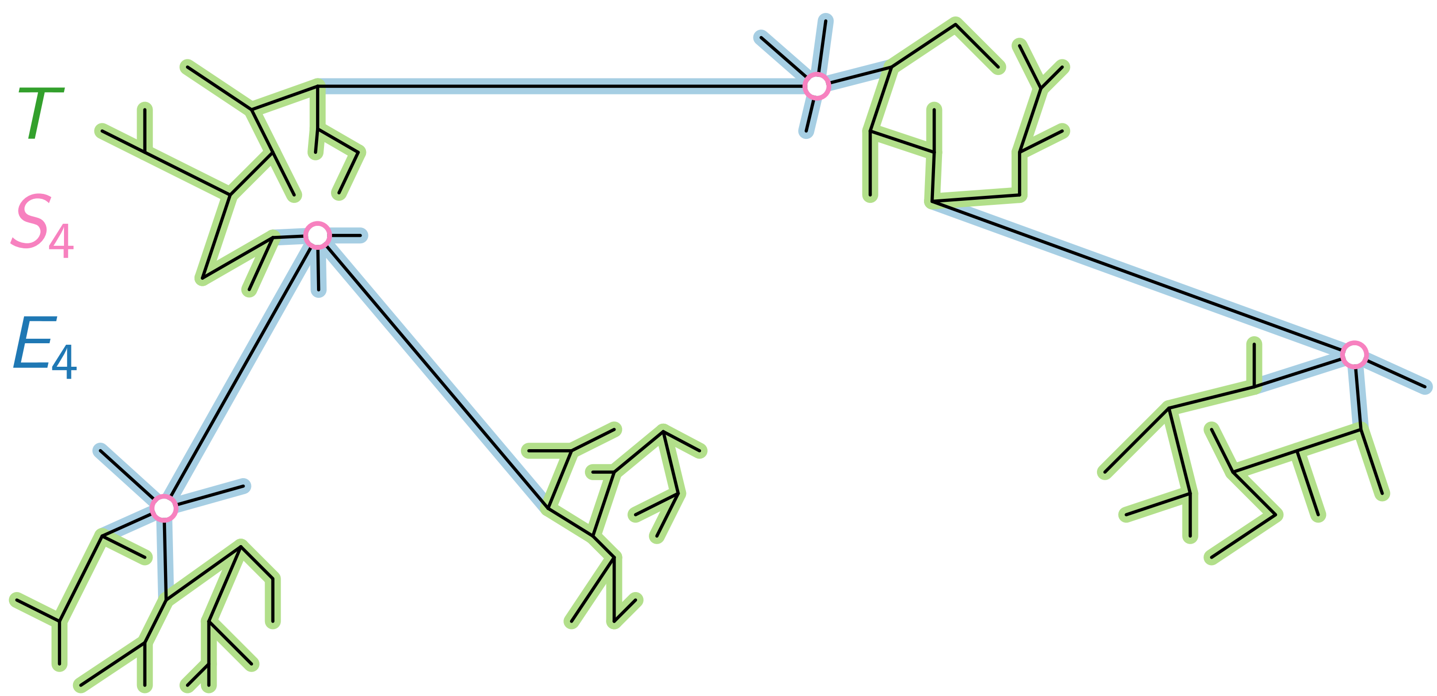
Proof. (i) $|E_i| \geq$



Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \underset{\text{vertex-deg}}{\geq} i|S_i|$

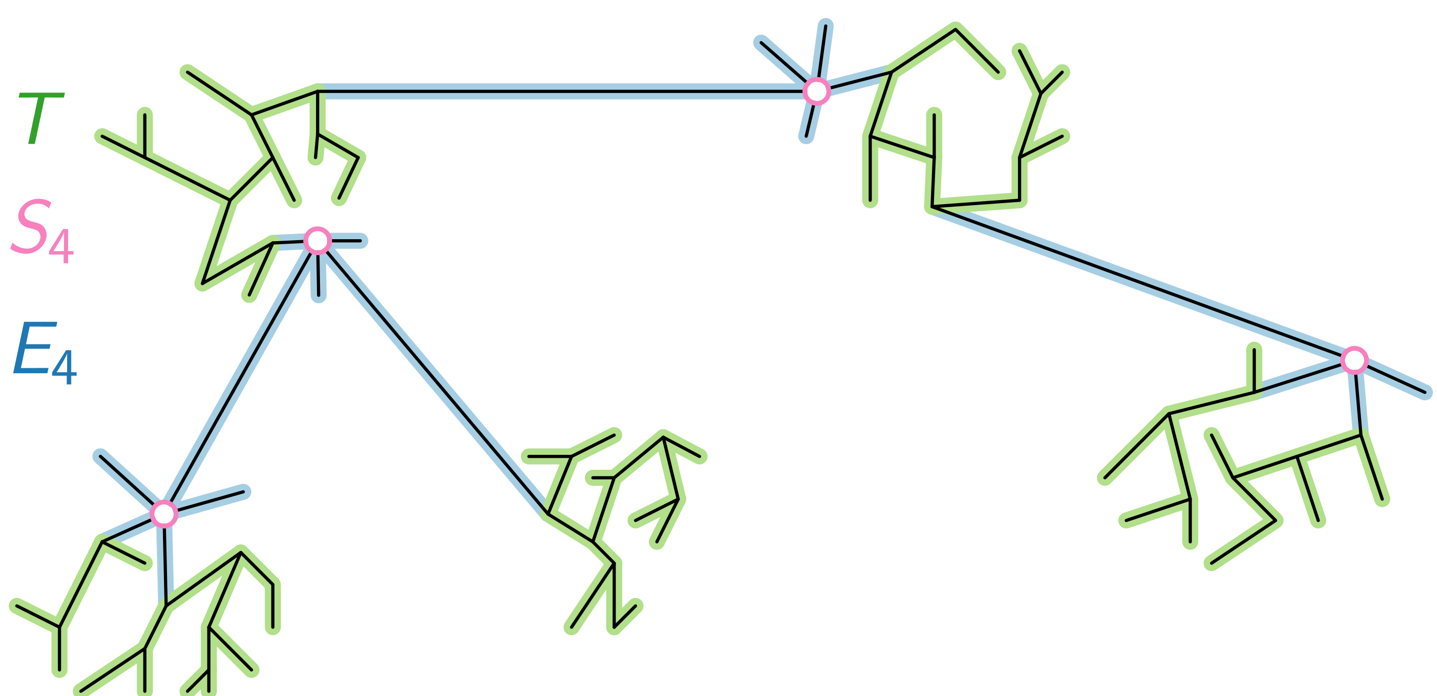


Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) =$

vertex-deg counted twice?

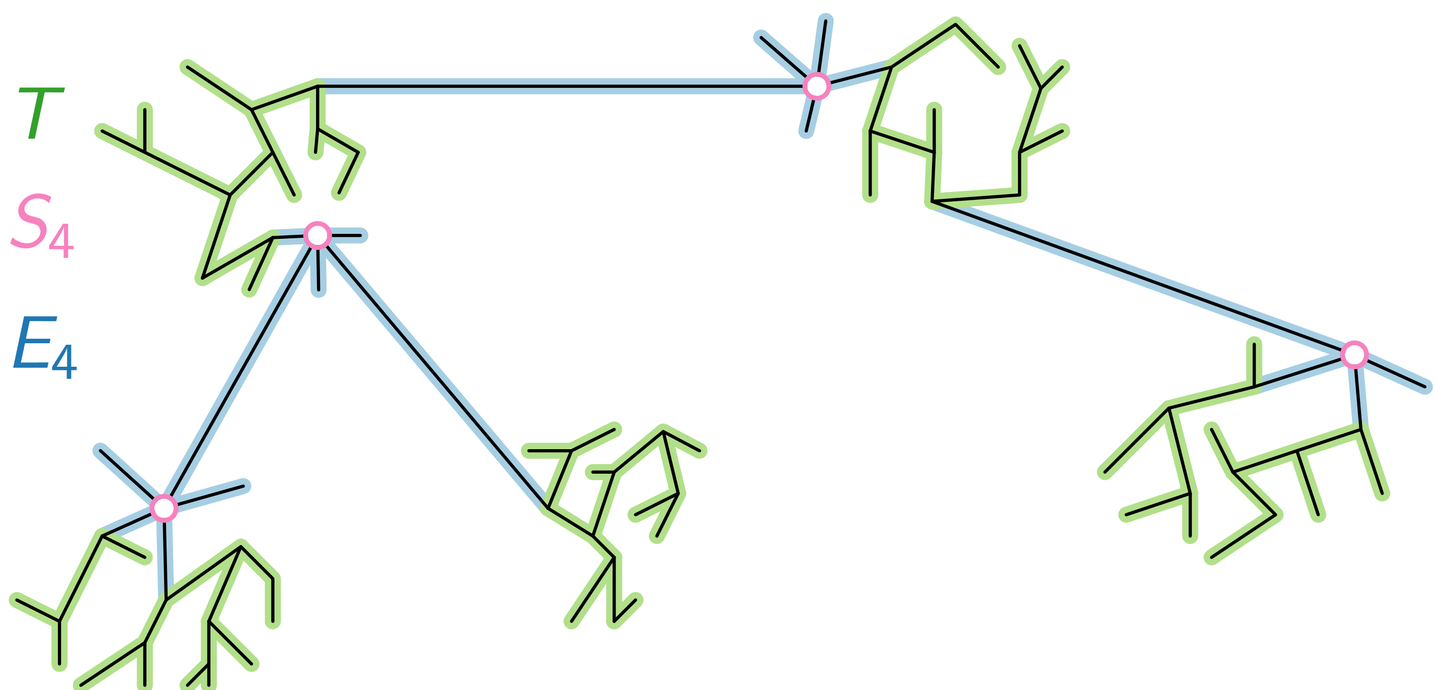


Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$

vertex-deg   counted twice?



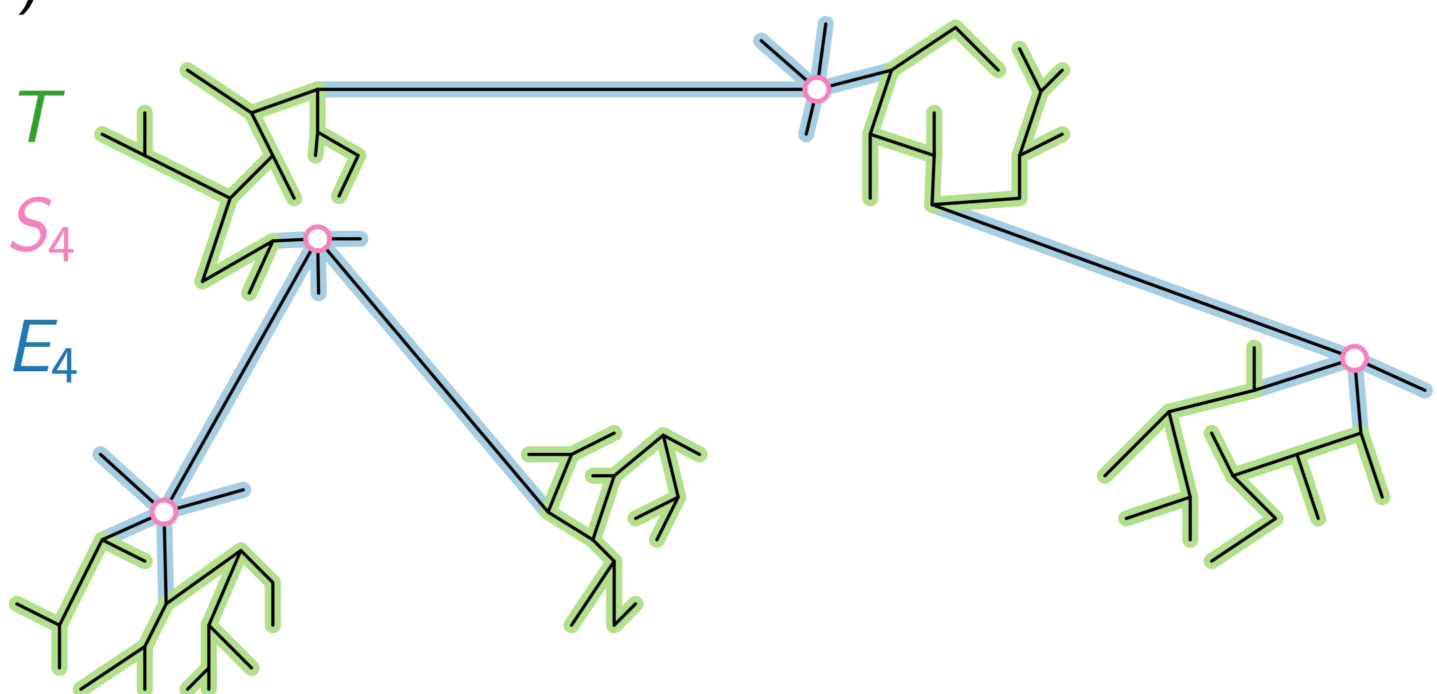
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$

vertex-deg counted twice?

(ii)



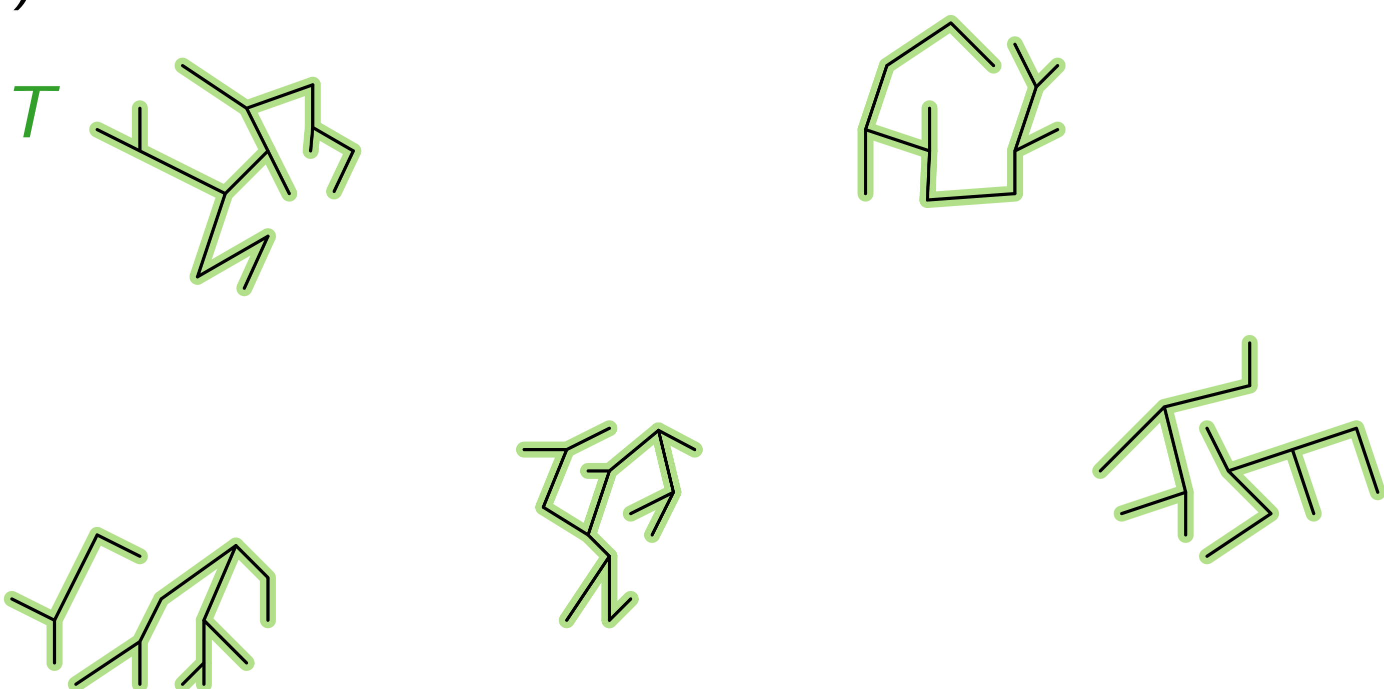
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$

vertex-deg   counted twice?

(ii)



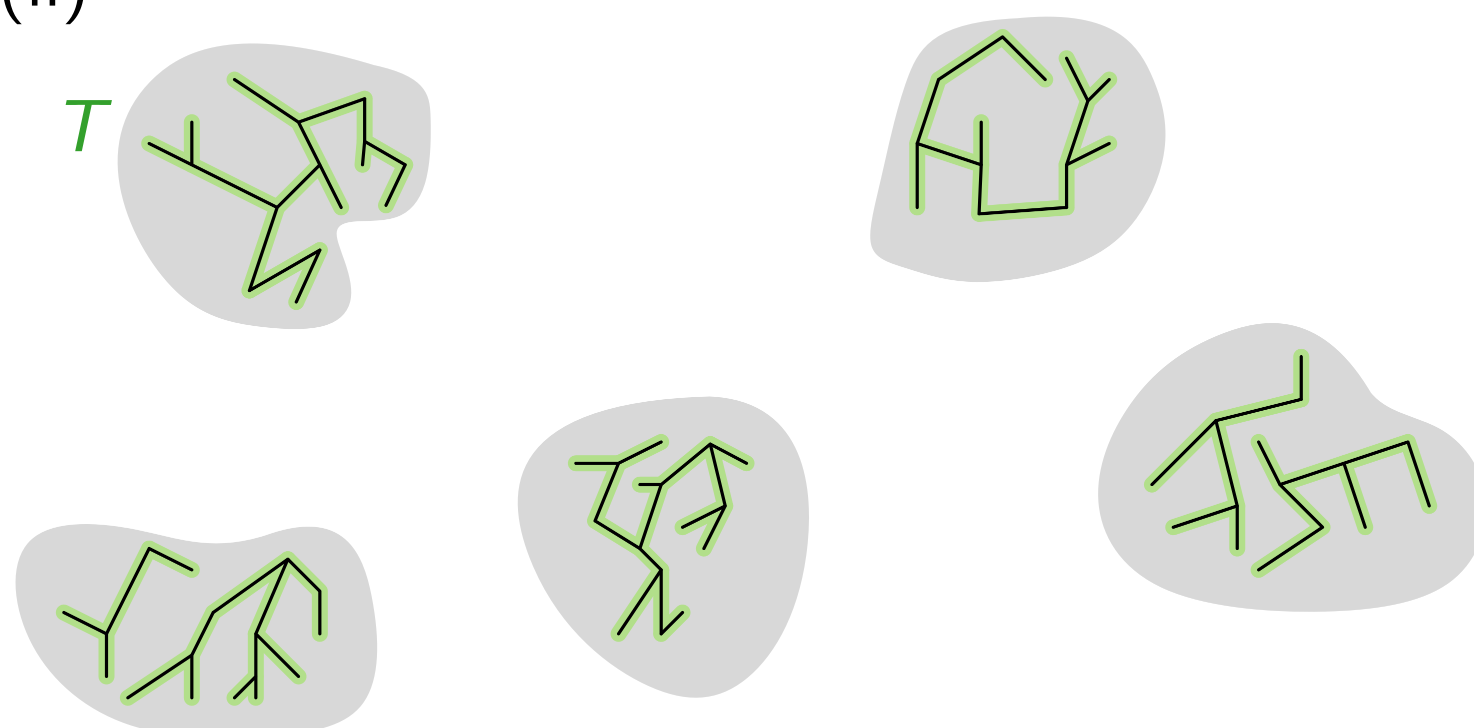
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$

vertex-deg   counted twice?

(ii)



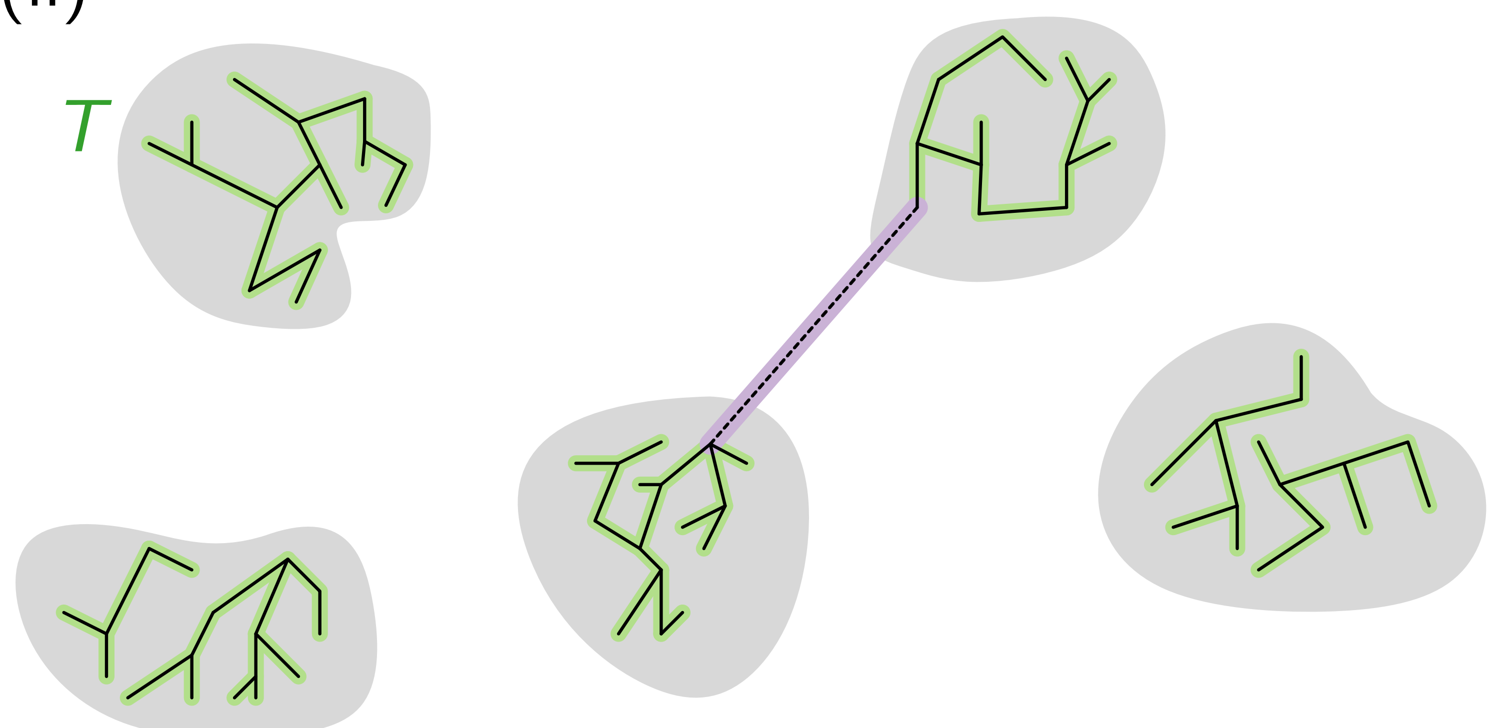
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$

vertex-deg counted twice?

(ii)



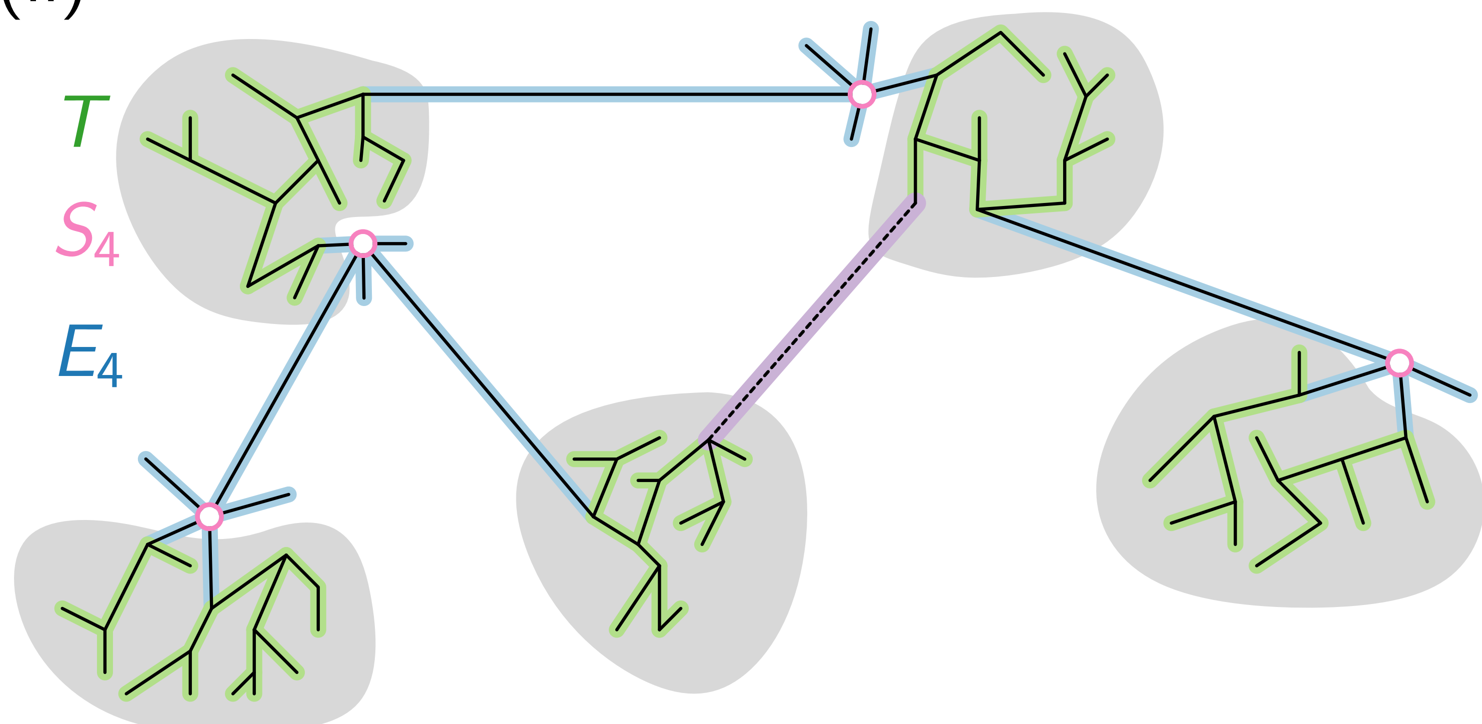
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$

vertex-deg   counted twice?

(ii)



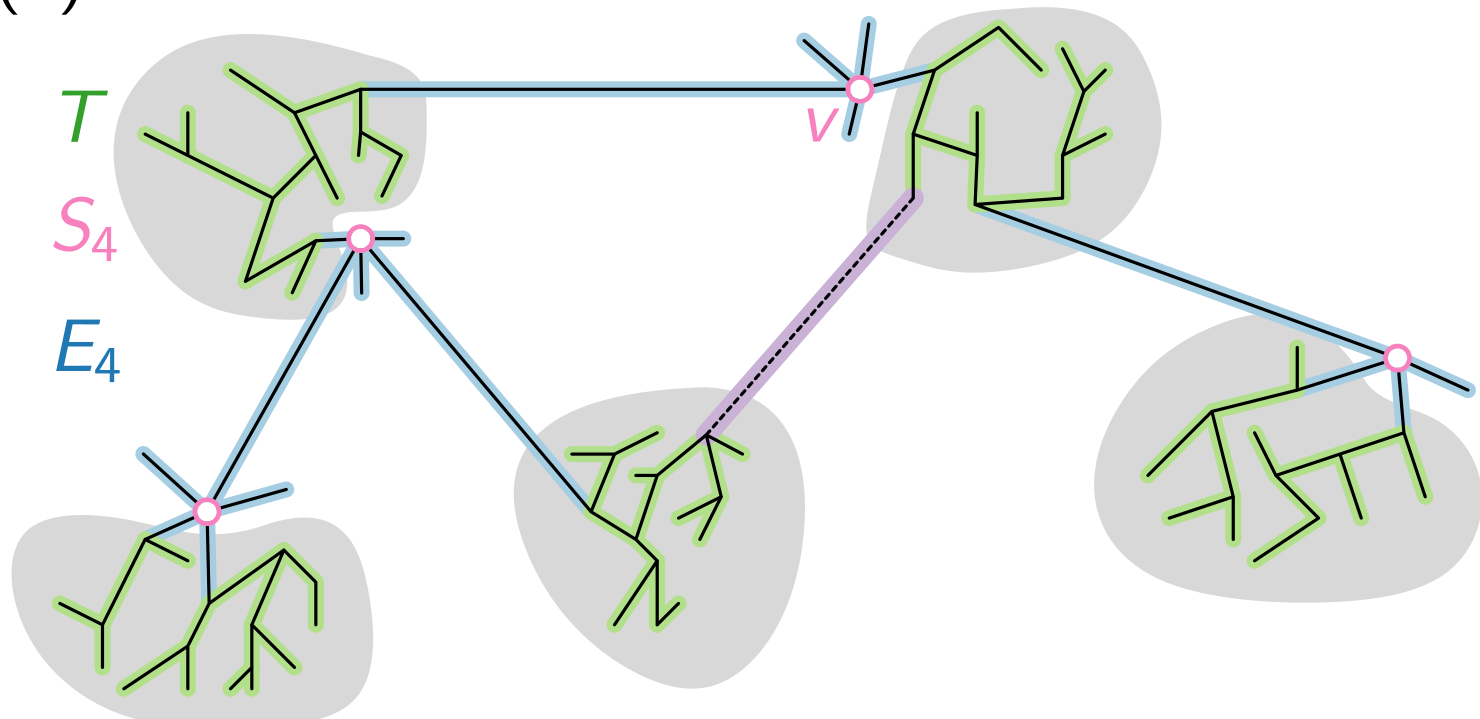
Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$

vertex-deg counted twice?

(ii)

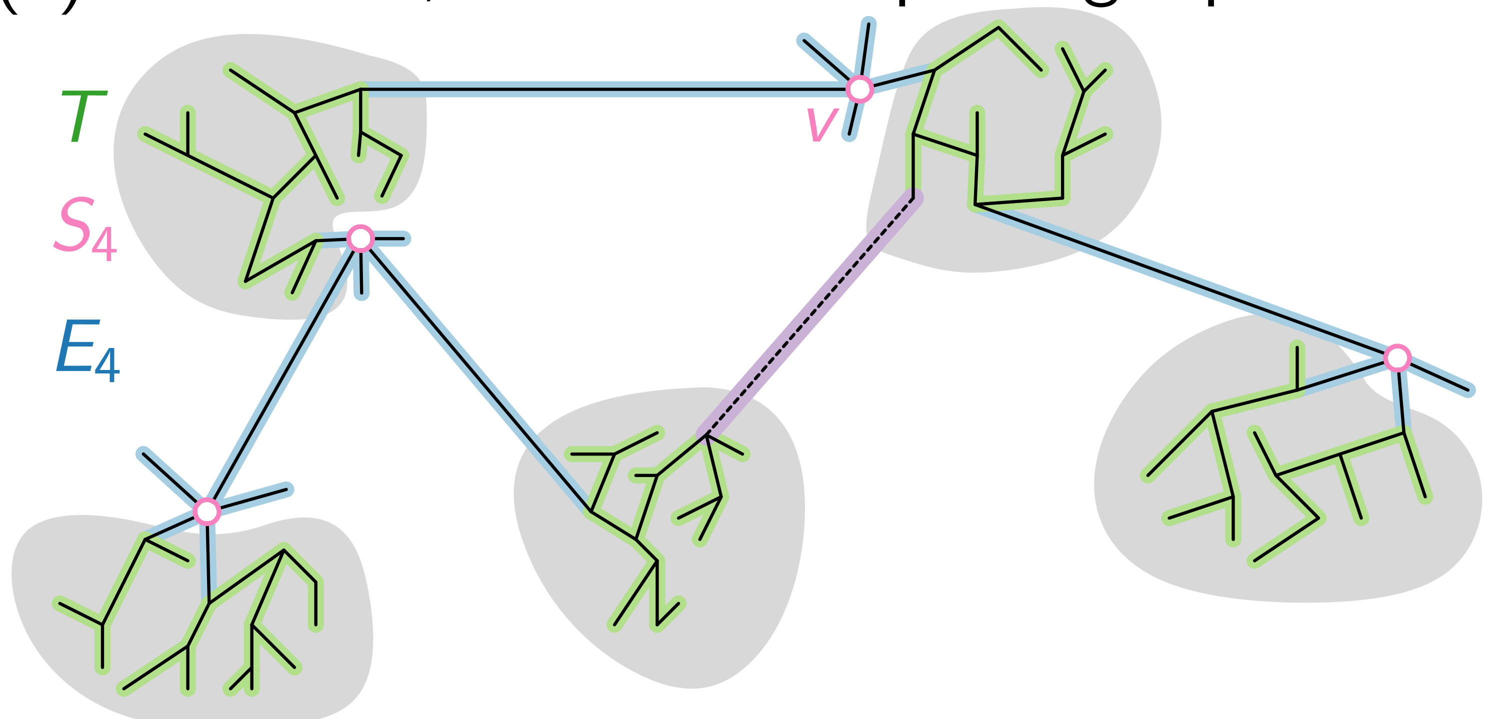


Structure of a Decomposition

Lemma 3. For locally opt. spanning tree T, $i \geq \Delta(T) - \ell + 1$:
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Proof. (i) $|E_i| \geq i|S_i| - (|S_i| - 1) = (i-1)|S_i| + 1$
vertex-deg counted twice?

(ii) Otherwise, there is an improving flip for some $v \in S_i$.



Approximation Algorithms

Lecture 10:
Minimum-Degree Spanning Tree
via Local Search

Part V:
Approximation Factor

Approximation Factor

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

→ Remove E_i for this i!

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree. Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$. Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Remove E_i for this i! $\Rightarrow$ S_{i-1} covers edges between comp.

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree. Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$. Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Remove E_i for this i! $\Rightarrow$ S_{i-1} covers edges between comp.

$\mathrm{OPT} \underset{\text{Lemma 1}}{\geq} \frac{k}{|S|} =$

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Remove E_i for this i! $\Rightarrow$ S_{i-1} covers edges between comp.

$$\mathrm{OPT} \underset{\text{Lemma 1}}{\geq} \frac{k}{|S|} = \frac{|E_i|}{|S_{i-1}|} \geq$$

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Remove E_i for this i! $\Rightarrow$ S_{i-1} covers edges between comp.

$$\mathrm{OPT} \underset{\text{Lemma 1}}{\geq} \frac{k}{|S|} = \frac{|E_i|}{|S_{i-1}|} \underset{\text{Lemma 3}}{\geq} \frac{(i-1)|S_i|+1}{|S_{i-1}|} \geq$$

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree. Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$. Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Remove E_i for this i! $\Rightarrow$ S_{i-1} covers edges between comp.

$$\mathrm{OPT} \underset{\text{Lemma 1}}{\geq} \frac{k}{|S|} = \frac{|E_i|}{|S_{i-1}|} \underset{\text{Lemma 3}}{\geq} \frac{(i-1)|S_i|+1}{|S_{i-1}|} \underset{\text{Lemma 2}}{\geq} \frac{(i-1)|S_i|+1}{2|S_i|} >$$

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Remove E_i for this i! $\Rightarrow$ S_{i-1} covers edges between comp.

$$\mathrm{OPT} \underset{\text{Lemma 1}}{\geq} \frac{k}{|S|} = \frac{|E_i|}{|S_{i-1}|} \underset{\text{Lemma 3}}{\geq} \frac{(i-1)|S_i|+1}{|S_{i-1}|} \underset{\text{Lemma 2}}{\geq} \frac{(i-1)|S_i|+1}{2|S_i|} > \frac{(i-1)}{2} \underset{\text{Lemma 2}}{\geq}$$

Approximation Factor

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. Let T be a locally optimal spanning tree.
Then $\Delta(T) \leq 2 \cdot \mathrm{OPT} + \ell$, where $\ell = \lceil \log_2 n \rceil$.

Proof. Let S_i be the vertices v in T with $\deg_T(v) \geq i$.
Let E_i be the edges in T incident to S_i.

Lemma 1. $\mathrm{OPT} \geq k/|S|$ if $k = |\text{removed edges}|$, S vertex cover.

Lemma 2. $\exists i$ s.t. $\Delta(T) - \ell + 1 \leq i \leq \Delta(T)$ with $|S_{i-1}| \leq 2|S_i|$.

Lemma 3. For $i \geq \Delta(T) - \ell + 1$,
(i) $|E_i| \geq (i-1)|S_i| + 1$,
(ii) Each edge $e \in E(G) \setminus E_i$ connecting distinct components of $T \setminus E_i$ is incident to a node of S_{i-1}.

Remove E_i for this i! $\Rightarrow$ S_{i-1} covers edges between comp.

$$\mathrm{OPT} \underset{\text{Lemma 1}}{\geq} \frac{k}{|S|} = \frac{|E_i|}{|S_{i-1}|} \underset{\text{Lemma 3}}{\geq} \frac{(i-1)|S_i|+1}{|S_{i-1}|} \underset{\text{Lemma 2}}{\geq} \frac{(i-1)|S_i|+1}{2|S_i|} > \frac{(i-1)}{2} \underset{\text{Lemma 2}}{\geq} \frac{\Delta(T)-\ell}{2} \qquad \square$$

Approximation Algorithms

Lecture 10:
Minimum-Degree Spanning Tree
via Local Search

Part VI:
Termination, Running Time & Extensions

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree efficiently.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree efficiently.

Proof.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree efficiently.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully):

- Each iteration decreases the potential of a solution.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree efficiently.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully):

- Each iteration decreases the potential of a solution.
- The function is bounded both from above and below.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree efficiently.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully):

- Each iteration decreases the potential of a solution.
- The function is bounded both from above and below.
- Executing $f(n)$ iterations would exceed the lower bound.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully):

- Each iteration decreases the potential of a solution.
- The function is bounded both from above and below.
- Executing $f(n)$ iterations would exceed the lower bound.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.
- The function is bounded both from above and below.
- Executing $f(n)$ iterations would exceed the lower bound.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

- The function is bounded both from above and below.
- Executing $f(n)$ iterations would exceed the lower bound.

Homework

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

How does $\Phi(T)$ change?

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

How does $\Phi(T)$ change?

$\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq$

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

How does $\Phi(T)$ change?

$\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq$

$1 + x \leq e^x$

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

How does $\Phi(T)$ change?

$\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq (e^{-\frac{2}{27n^3}})^{f(n)} =$

$1 + x \leq e^x$

Homework

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

How does $\Phi(T)$ change?

$\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq (e^{-\frac{2}{27n^3}})^{f(n)} =$

Goal: After $f(n)$ iterations: $\Phi(T) = n < 3n$.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

  Let $f(n) = \frac{27}{2}n^4 \cdot \ln 3$. How does $\Phi(T)$ change?

  $\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq (e^{-\frac{2}{27n^3}})^{f(n)} =$

  Goal: After $f(n)$ iterations: $\Phi(T) = n < 3n$.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

  Let $f(n) = \frac{27}{2}n^4 \cdot \ln 3$. How does $\Phi(T)$ change?

  $\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq (e^{-\frac{2}{27n^3}})^{f(n)} = e^{-n \ln 3} =$

  Goal: After $f(n)$ iterations: $\Phi(T) = n < 3n$.

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $f(n)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

  Let $f(n) = \frac{27}{2} n^4 \cdot \ln 3$. How does $\Phi(T)$ change?

  $\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq (e^{-\frac{2}{27n^3}})^{f(n)} = e^{-n \ln 3} = 3^{-n}$

  Goal: After $f(n)$ iterations: $\Phi(T) = n < 3n$. $\square$

Homework

Termination and Running Time

Theorem. The algorithm finds a locally optimal spanning tree after at most $O(n^4)$ iterations.

Proof. Via potential function $\Phi(T)$ measuring the value of a solution where (hopefully): $\Phi(T) = \sum_{v \in V(G)} 3^{\deg_T(v)}$

- Each iteration decreases the potential of a solution.

Lemma. After each flip $T \to T'$, $\Phi(T') \leq (1 - \frac{2}{27n^3})\Phi(T)$.

Homework

- The function is bounded both from above and below.

Lemma. For every spanning tree T, $\Phi(T) \in [3n, n3^n]$.

- Executing $f(n)$ iterations would exceed the lower bound.

  Let $f(n) = \frac{27}{2}n^4 \cdot \ln 3$. How does $\Phi(T)$ change?

  $\Phi(T)$ decreases by: $(1 - \frac{2}{27n^3})^{f(n)} \leq (e^{-\frac{2}{27n^3}})^{f(n)} = e^{-n \ln 3} = 3^{-n}$

  Goal: After $f(n)$ iterations: $\Phi(T) = n < 3n$. □

Extensions

Corollary. For any constant $b > 1$ and $\ell = \lceil \log_b n \rceil$, the local search algorithm runs in polynomial time and produces a spanning tree T with $\Delta(T) \leq b \cdot \mathrm{OPT} + \ell$.

Extensions

Corollary. For any constant $b > 1$ and $\ell = \lceil \log_b n \rceil$, the local search algorithm runs in polynomial time and produces a spanning tree T with $\Delta(T) \leq b \cdot \mathrm{OPT} + \ell$.

Proof. Similar to previous pages. Homework □

Extensions

Corollary. For any constant $b > 1$ and $\ell = \lceil \log_b n \rceil$, the local search algorithm runs in polynomial time and produces a spanning tree T with $\Delta(T) \leq b \cdot \mathrm{OPT} + \ell$.

Proof. Similar to previous pages. Homework $\square$

- A variant of this algorithm yields the following result:

Extensions

Corollary. For any constant $b > 1$ and $\ell = \lceil \log_b n \rceil$, the local search algorithm runs in polynomial time and produces a spanning tree T with $\Delta(T) \leq b \cdot \mathrm{OPT} + \ell$.

Proof. Similar to previous pages. Homework □

- A variant of this algorithm yields the following result:

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. There is a local search algorithm that runs in $O(EV\alpha(E, V) \log V)$ time and produces a spanning tree T with $\Delta(T) \leq \mathrm{OPT} + 1$.

Extensions

Corollary. For any constant $b > 1$ and $\ell = \lceil \log_b n \rceil$, the local search algorithm runs in polynomial time and produces a spanning tree T with $\Delta(T) \leq b \cdot \mathrm{OPT} + \ell$.

Proof. Similar to previous pages. Homework □

- A variant of this algorithm yields the following result:

[Fürer & Raghavachari: SODA'92, JA'94]

Theorem. There is a local search algorithm that runs in $O(EV\alpha(E,V)\log V)$ time and produces a spanning tree T with $\Delta(T) \leq \mathrm{OPT} + 1$.

- Further variants for directed graphs and Steiner tree.