Approximation Algorithms

Lecture 12:
SteinerForest via Primal-Dual

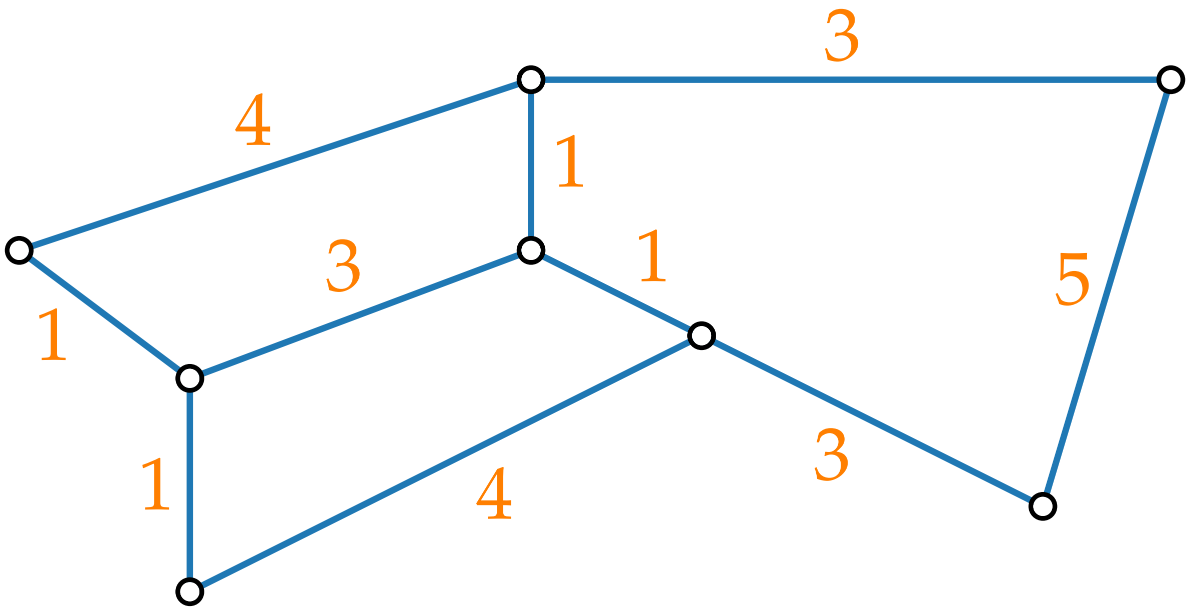
Part I:
SteinerForest

Philipp Kindermann

Summer Semester 2020

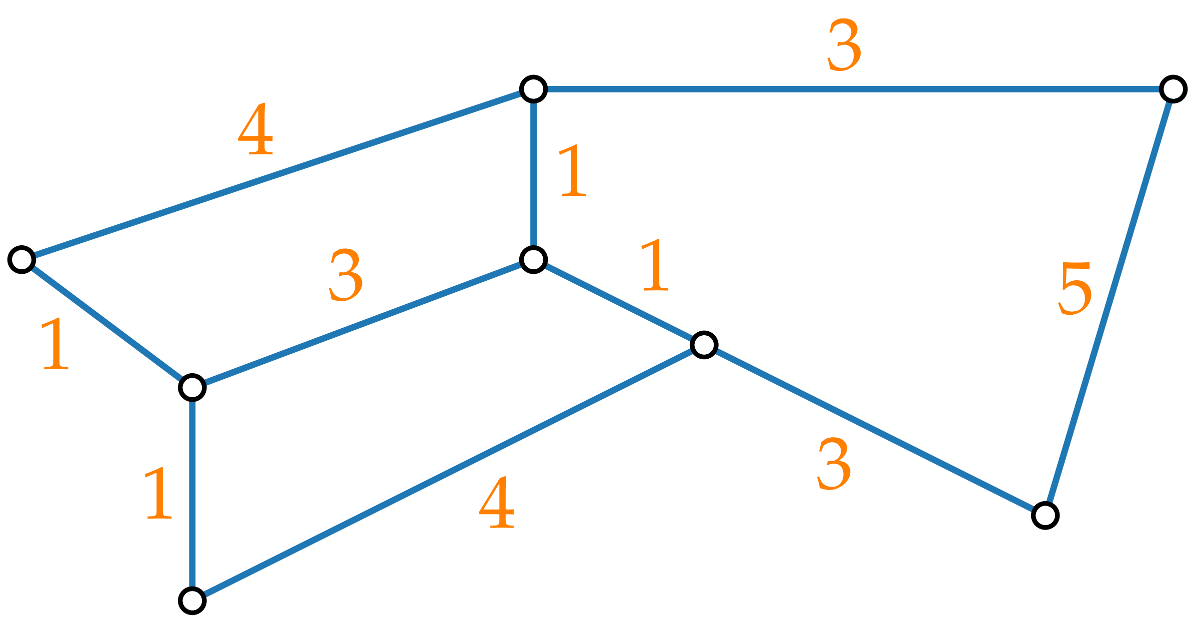
SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{N}$



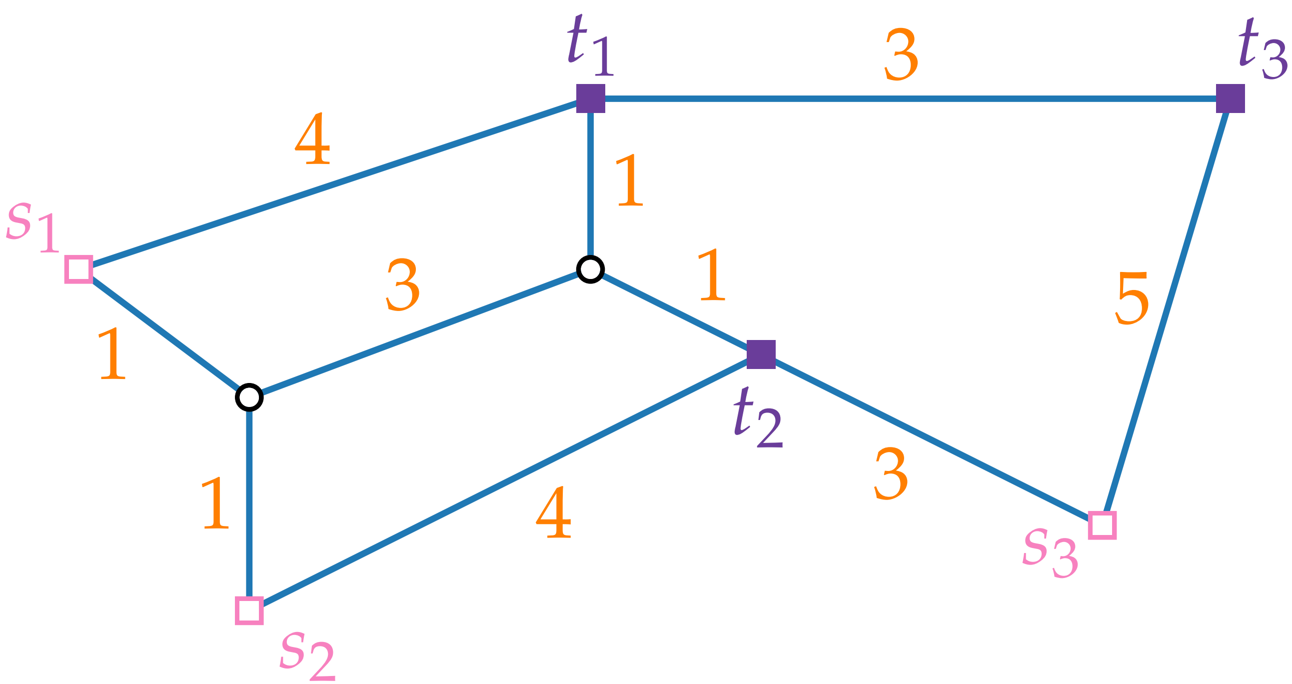
SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of k pairs of vertices



SteinerForest

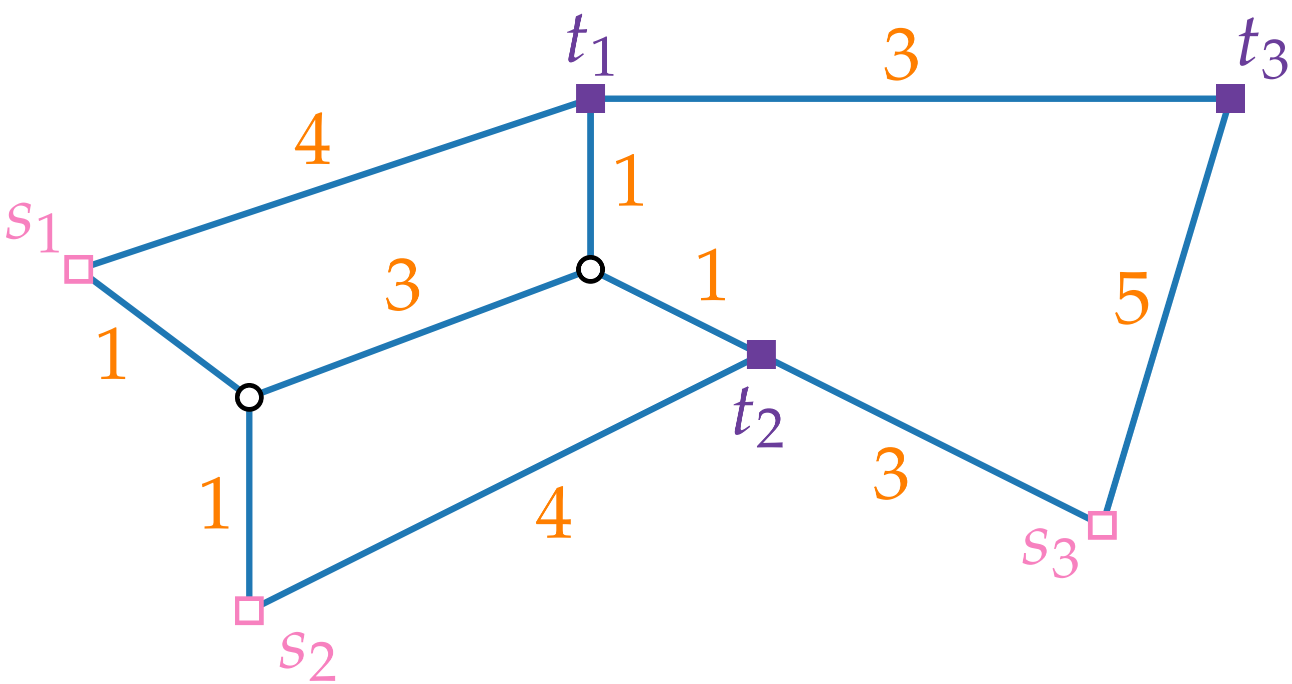
Given: A graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \dots, (s_k, t_k)\}$ of k pairs of vertices



SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of k pairs of vertices

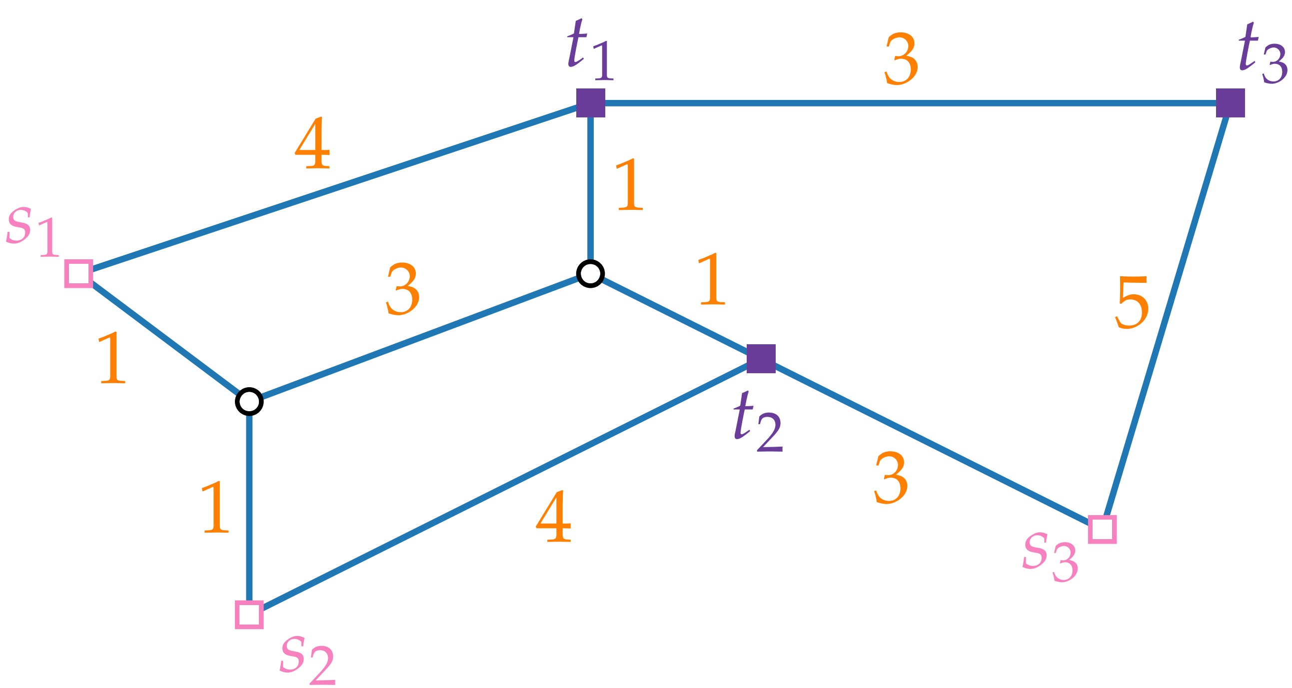
Task: Find an edge set $F \subseteq E$ with min. total cost $c(F)$



SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \dots, (s_k, t_k)\}$ of k pairs of vertices

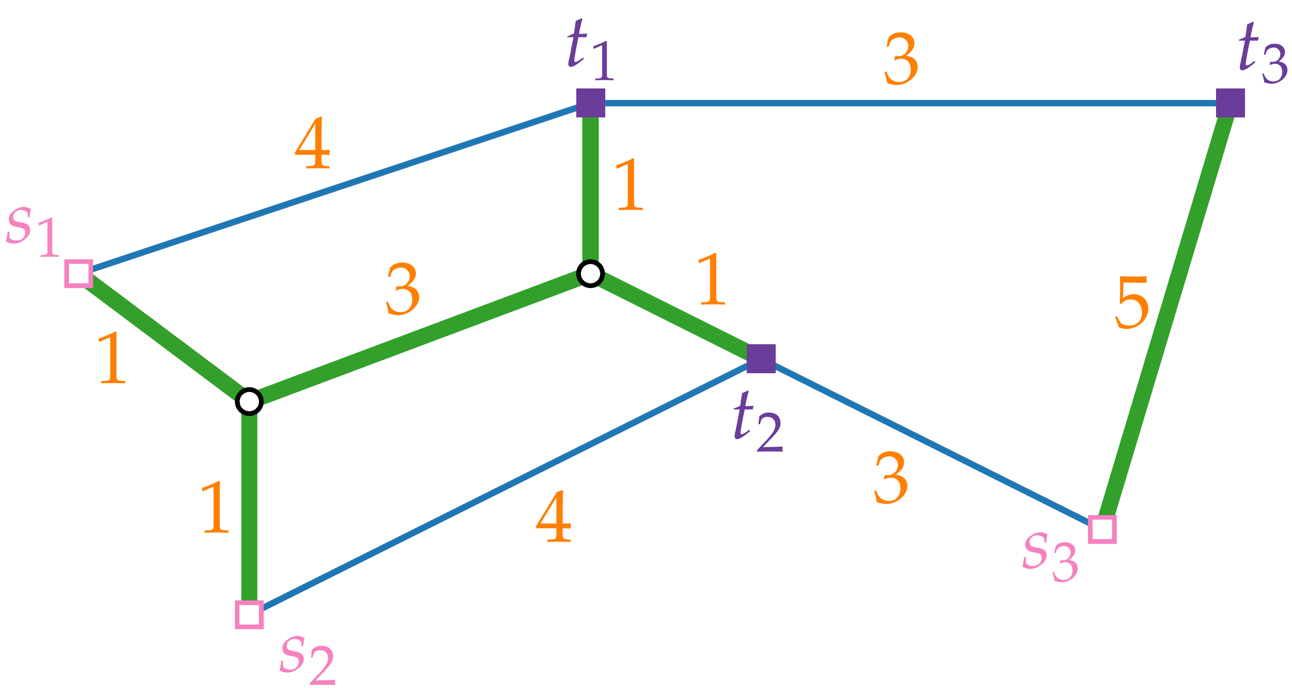
Task: Find an edge set $F \subseteq E$ with min. total cost $c(F)$ such that in the subgraph (V, F) each pair (s_i, t_i), $i = 1, \dots, k$ is connected.



SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \dots, (s_k, t_k)\}$ of k pairs of vertices

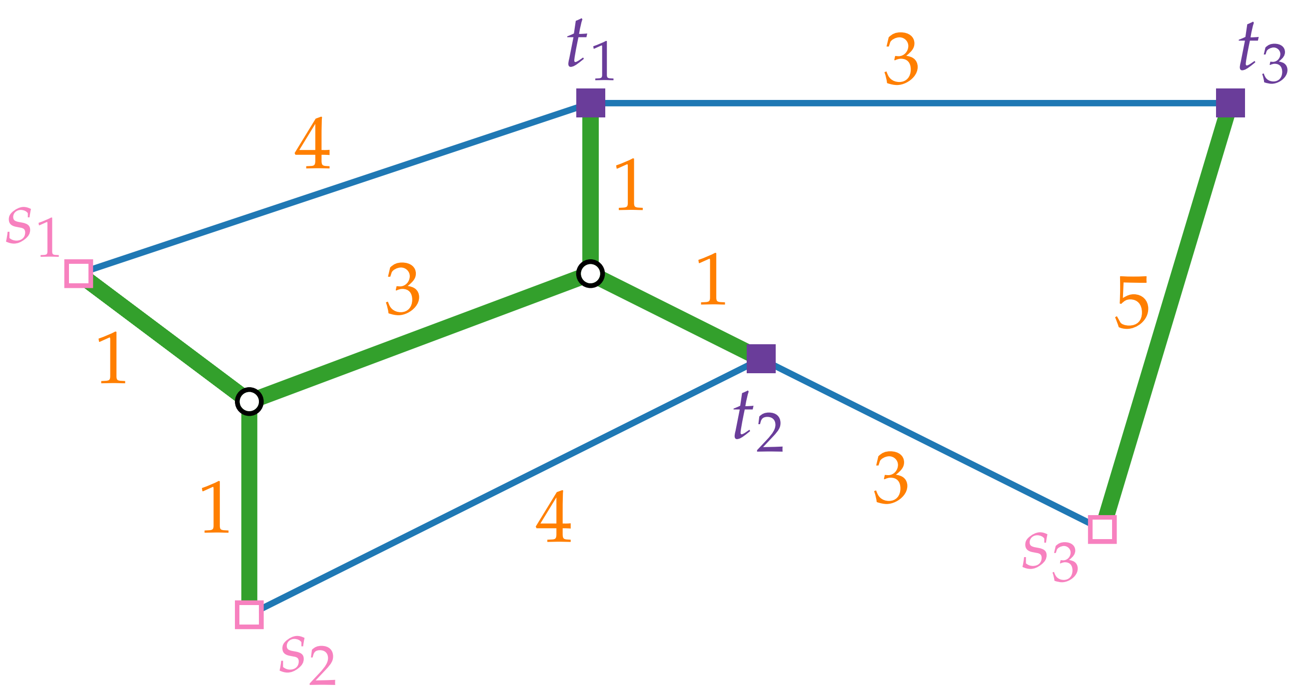
Task: Find an edge set $F \subseteq E$ with min. total cost $c(F)$ such that in the subgraph (V, F) each pair (s_i, t_i), $i = 1, \dots, k$ is connected.



SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of k pairs of vertices

Task: Find an edge set $F \subseteq E$ with min. total cost $c(F)$ such that in the subgraph (V, F) each pair (s_i, t_i), $i = 1, \ldots, k$ is connected.

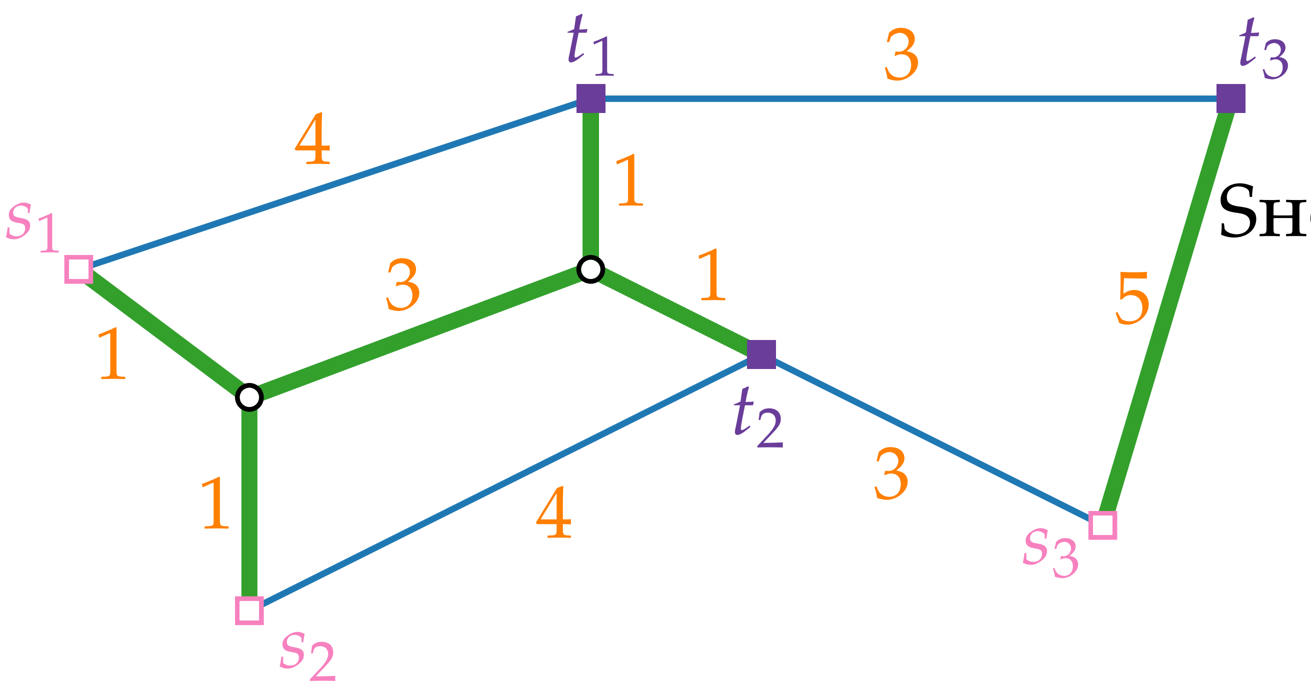


Special cases?

SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \dots, (s_k, t_k)\}$ of k pairs of vertices

Task: Find an edge set $F \subseteq E$ with min. total cost $c(F)$ such that in the subgraph (V, F) each pair (s_i, t_i), $i = 1, \dots, k$ is connected.



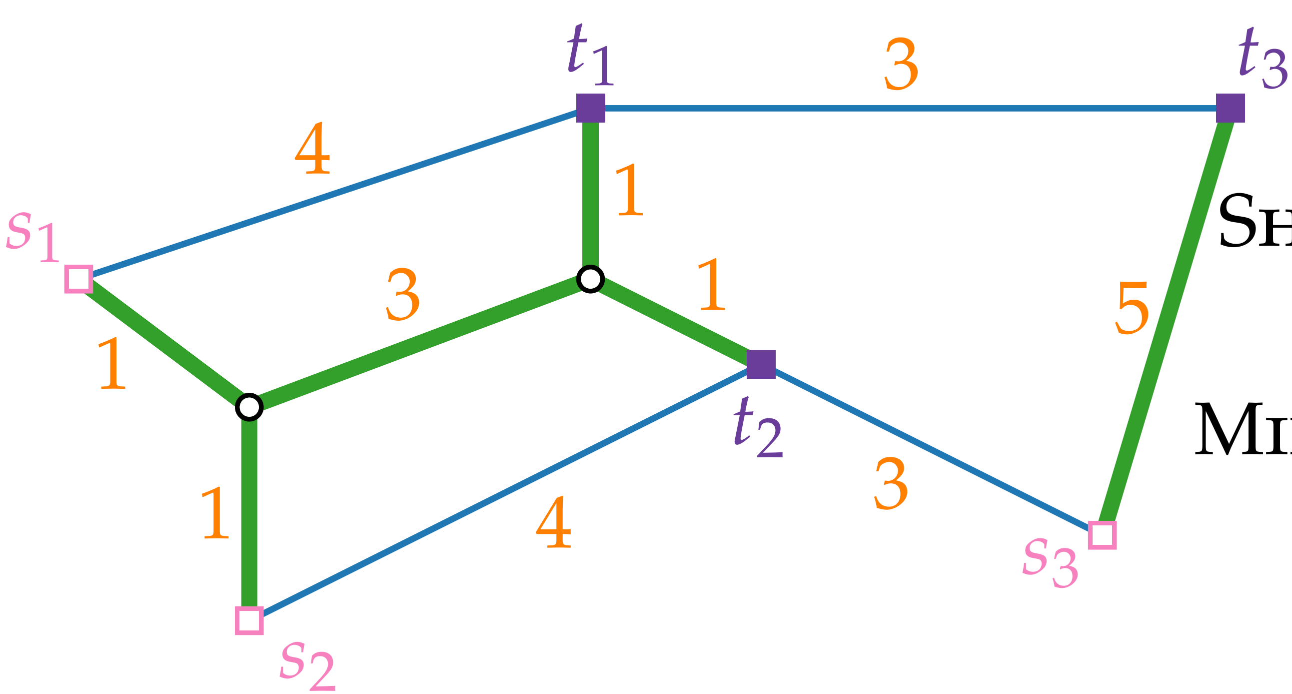
Special cases?

ShortestPath ($R = \{s, t\}$)

SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \dots, (s_k, t_k)\}$ of k pairs of vertices

Task: Find an edge set $F \subseteq E$ with min. total cost $c(F)$ such that in the subgraph (V, F) each pair (s_i, t_i), $i = 1, \dots, k$ is connected.



Special cases?

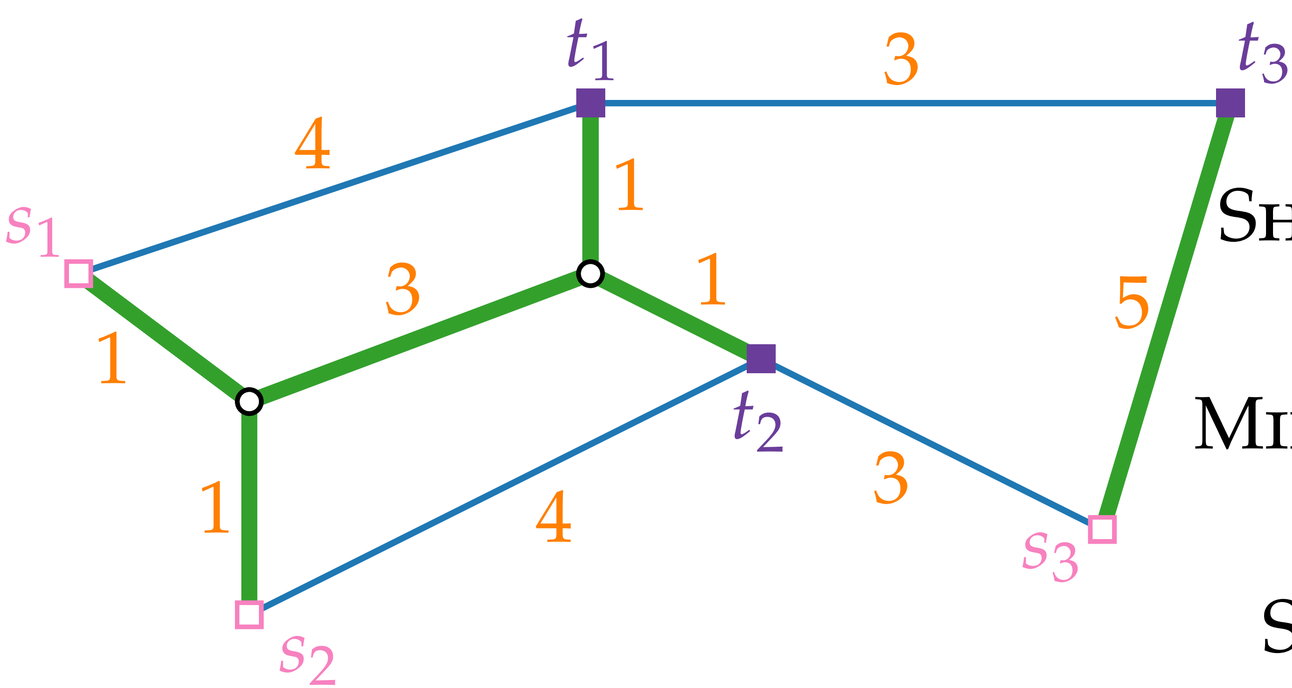
ShortestPath ($R = \{s, t\}$)

MinSpanningTree ($R = E$)

SteinerForest

Given: A graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \dots, (s_k, t_k)\}$ of k pairs of vertices

Task: Find an edge set $F \subseteq E$ with min. total cost $c(F)$ such that in the subgraph (V, F) each pair (s_i, t_i), $i = 1, \dots, k$ is connected.



Special cases?

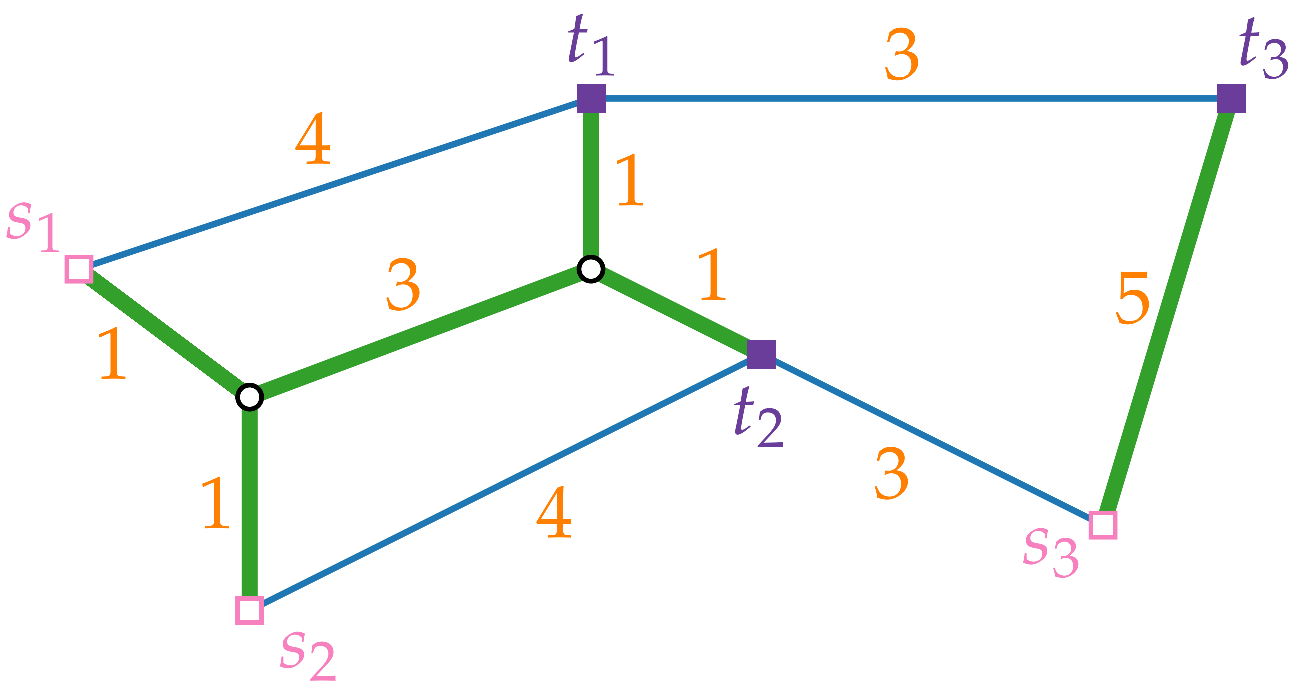
ShortestPath $(R = \{s, t\})$

MinSpanningTree $(R = E)$

SteinerTree $(R = T \times T)$

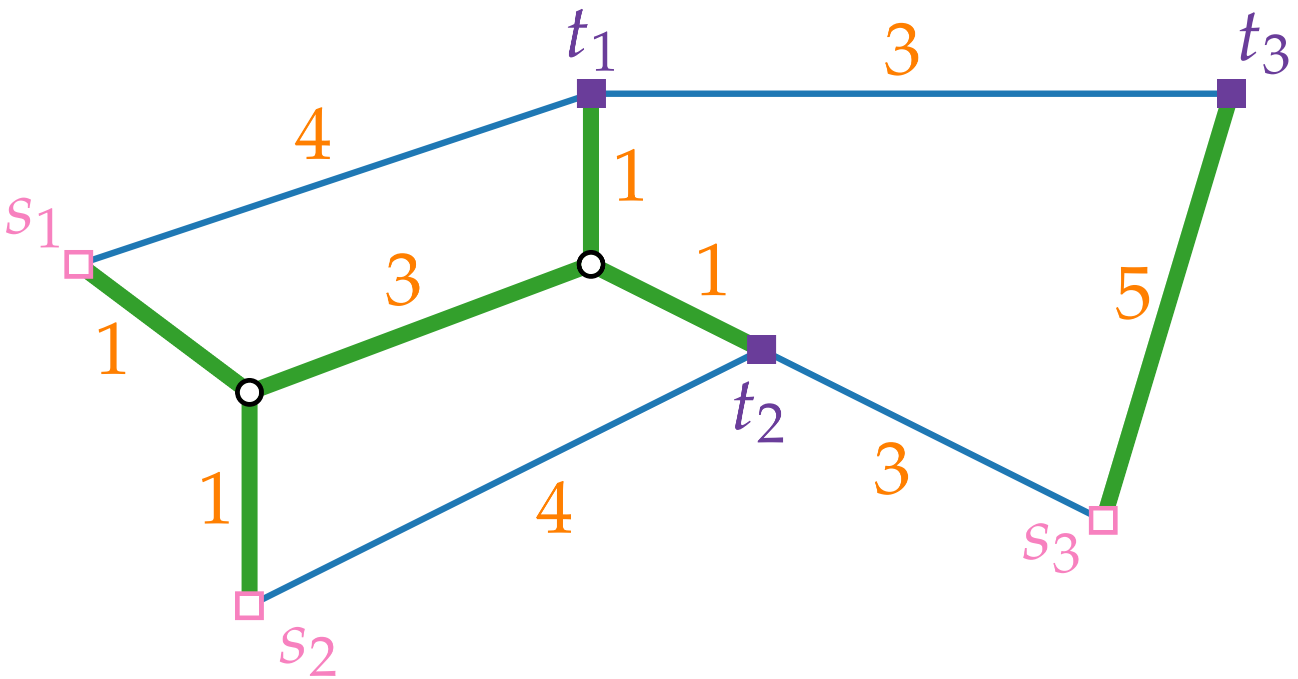
Approaches?

- Merge k shortest s_i-t_i-paths



Approaches?

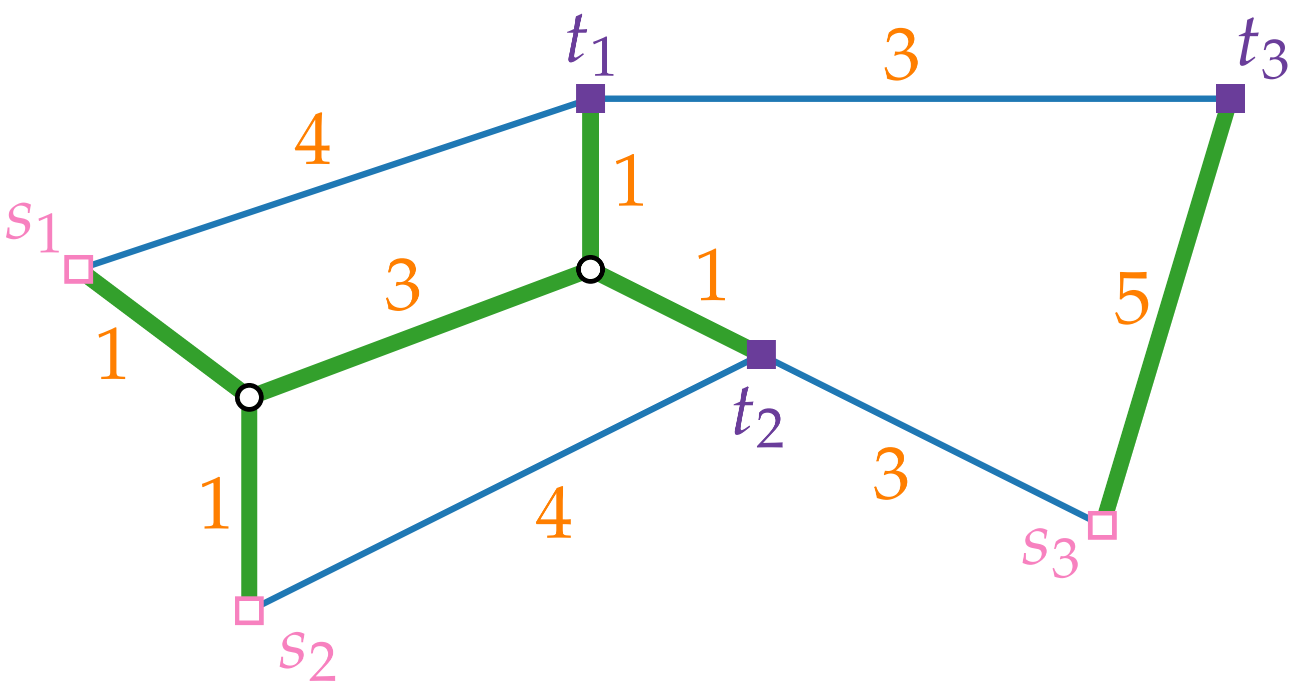
- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals



Approaches?

- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals

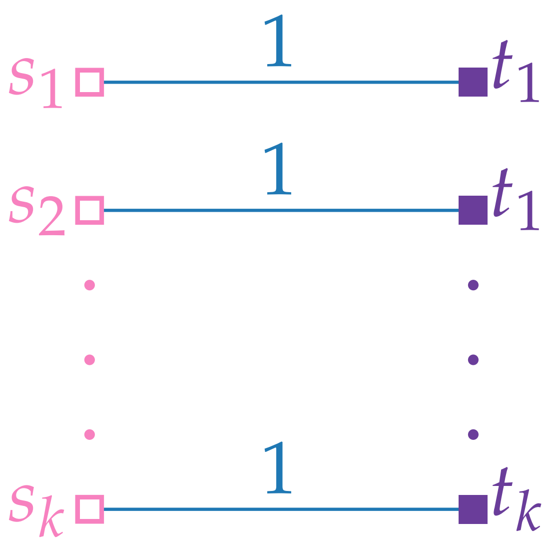
Above approaches perform poorly :-(



Approaches?

- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals

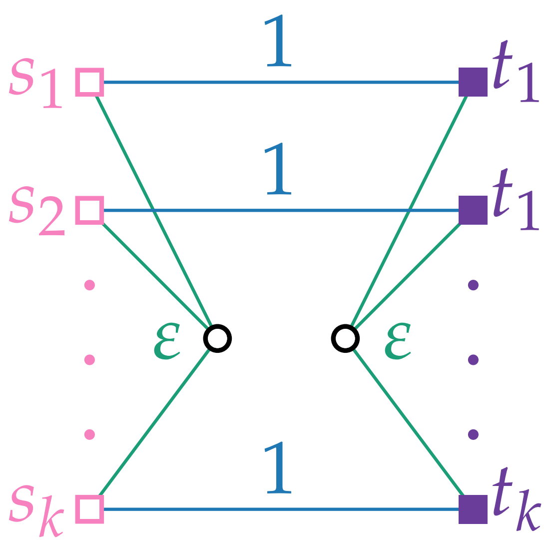
Above approaches perform poorly :-(



Approaches?

- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals

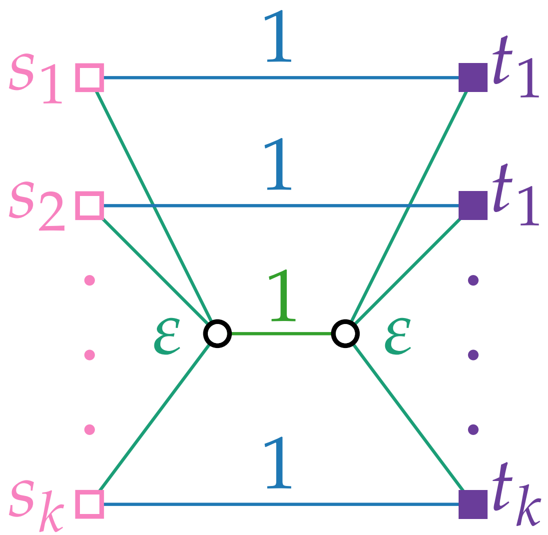
Above approaches perform poorly :-(



Approaches?

- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals

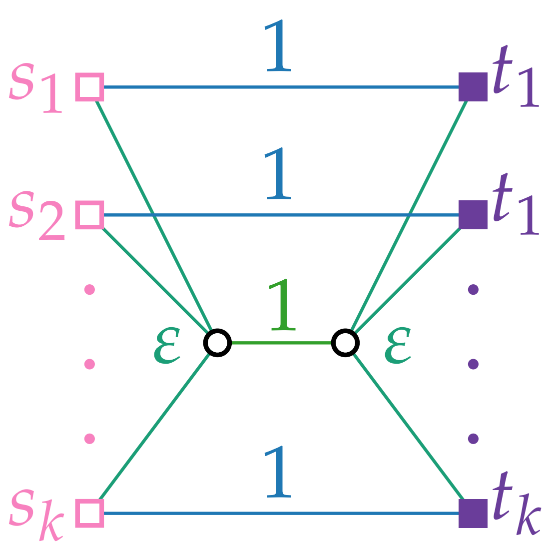
Above approaches perform poorly :-(

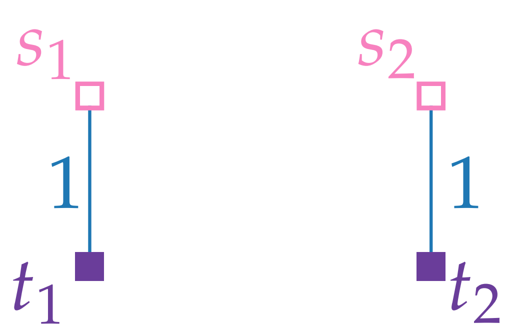


Approaches?

- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals

Above approaches perform poorly :-(

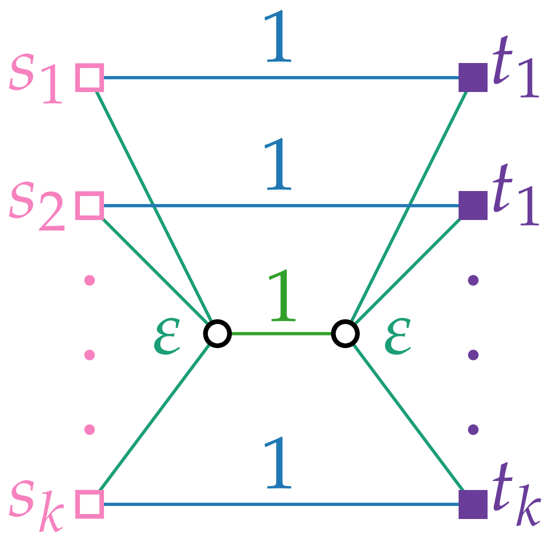


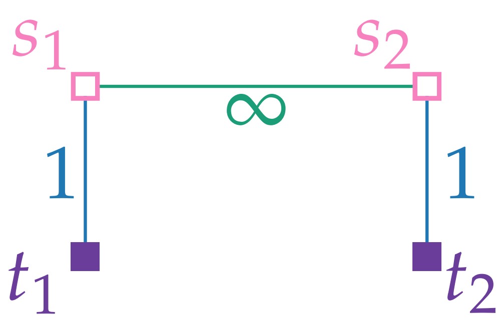


Approaches?

- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals

Above approaches perform poorly :-(



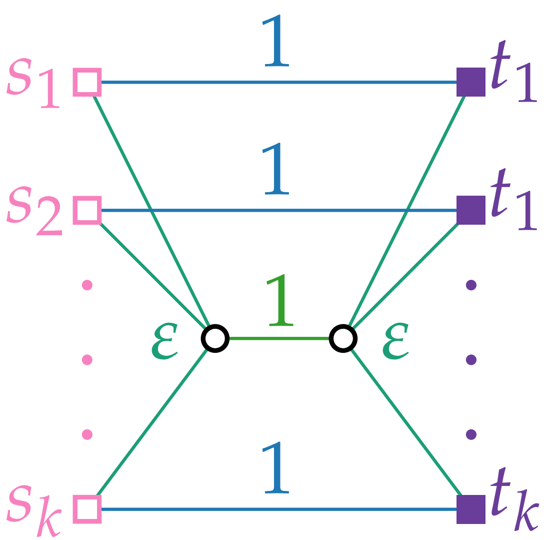


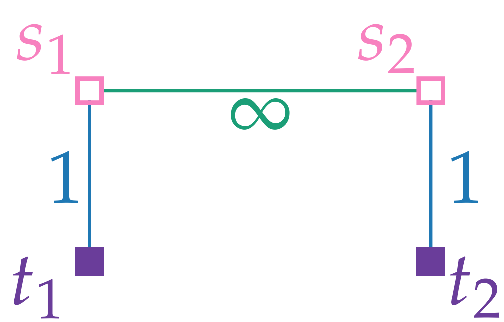
Approaches?

- Merge k shortest s_i-t_i-paths
- SteinerTree on the set of terminals

Above approaches perform poorly :-(

Difficulty: which terminals belong to the same tree of the forest?





Approximation Algorithms

Lecture 12:
SteinerForest via Primal-Dual

Part II:
Primal and Dual LP

Philipp Kindermann

Summer Semester 2020

An ILP

minimize

subject to

An ILP

minimize

subject to

$$x_e \in \{0, 1\} \qquad e \in E$$

An ILP

minimize $\sum_{e \in E} c_e x_e$

subject to

$x_e \in \{0, 1\} \qquad e \in E$

An ILP

minimize $\sum_{e \in E} c_e x_e$

subject to

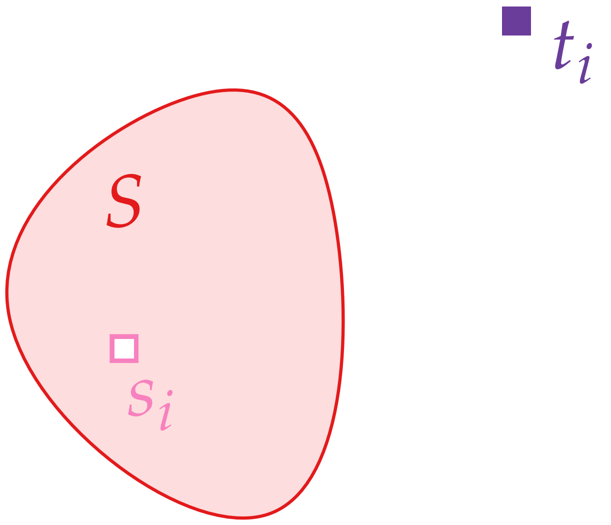
$$x_e \in \{0,1\} \qquad e \in E$$

An ILP

$$\textbf{minimize} \quad \sum_{e \in E} c_e x_e$$

$$\textbf{subject to}$$

$$x_e \in \{0, 1\} \qquad e \in E$$

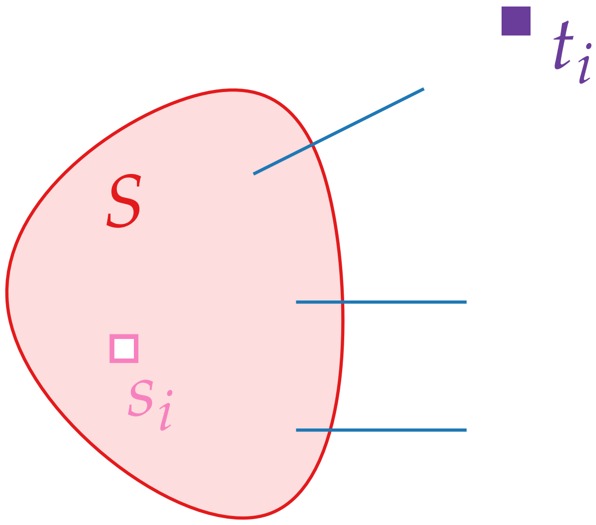


An ILP

minimize $\sum_{e \in E} c_e x_e$

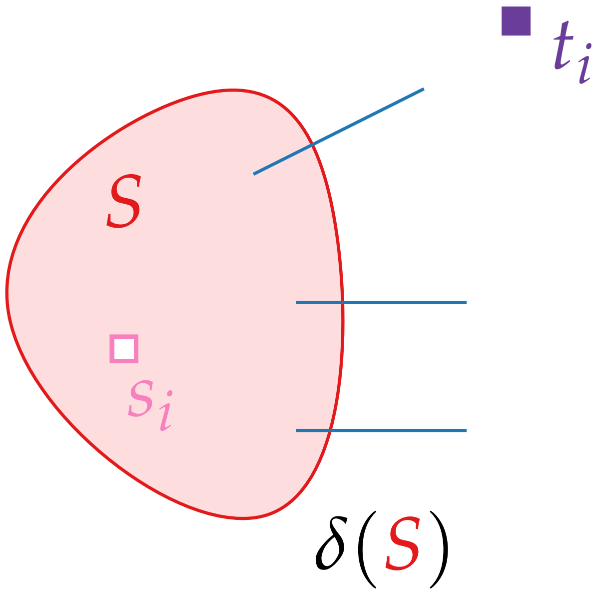
subject to

$x_e \in \{0,1\} \qquad e \in E$



An ILP

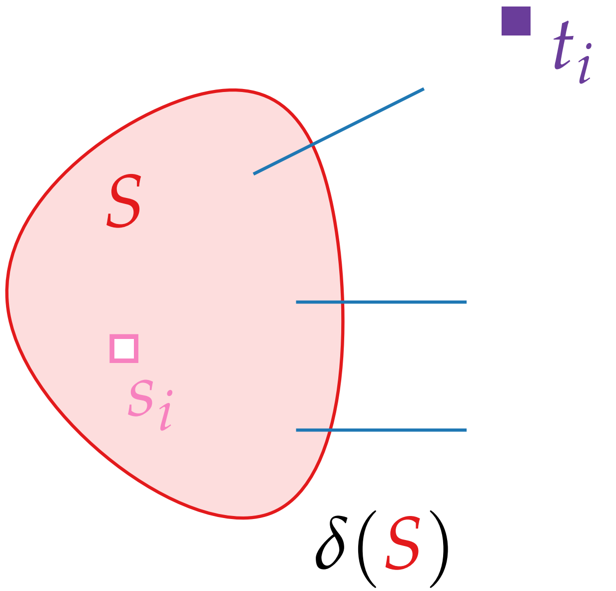
minimize	$\sum_{e \in E} c_e x_e$
subject to	
	$x_e \in \{0,1\} \qquad e \in E$



An ILP

$$\begin{array}{ll} \textbf{minimize} & \sum_{e \in E} c_e x_e \\ \textbf{subject to} & \\ & x_e \in \{0,1\} \qquad e \in E \end{array}$$

$$\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$$



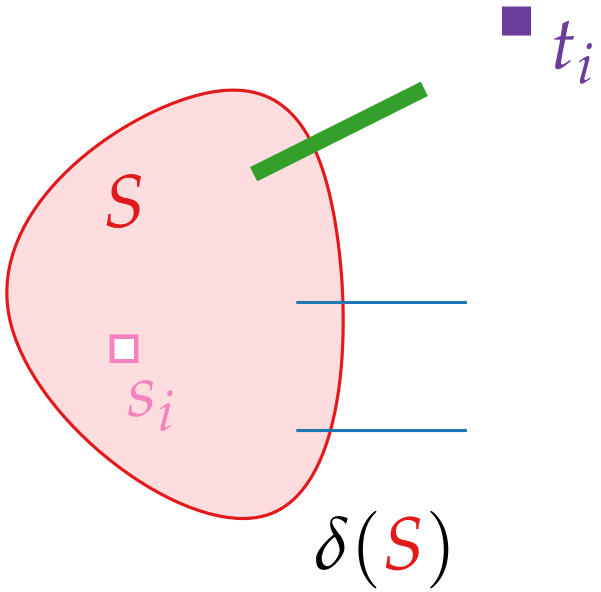
An ILP

minimize $\sum_{e \in E} c_e x_e$

subject to

$x_e \in \{0,1\} \qquad e \in E$

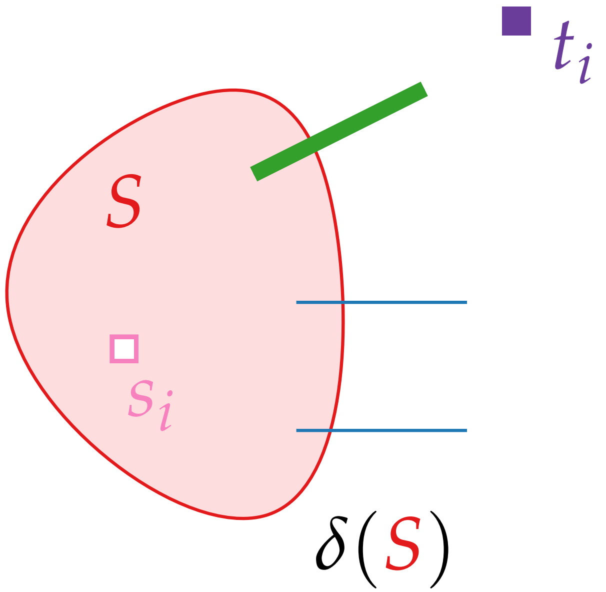
$$\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$$



An ILP

$$
\begin{array}{ll}
\textbf{minimize} & \displaystyle\sum_{e \in E} c_e x_e \\
\textbf{subject to} & \displaystyle\sum_{e \in \delta(S)} x_e \geq 1 \\
 & x_e \in \{0,1\} \qquad\qquad e \in E
\end{array}
$$

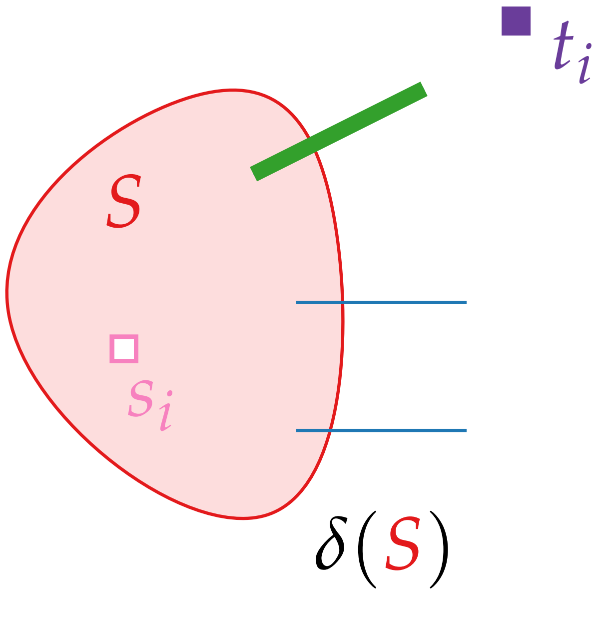
$$\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$$



An ILP

$$\begin{array}{lll} \textbf{minimize} & \sum_{e \in E} c_e x_e & \\ \textbf{subject to} & \sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \ldots, k \\ & x_e \in \{0,1\} & e \in E \end{array}$$

$$\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$$

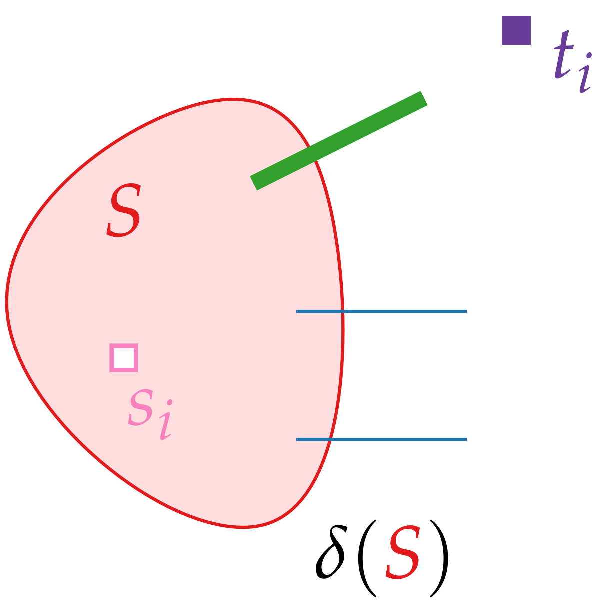


An ILP

$$\begin{array}{lll} \textbf{minimize} & \sum_{e \in E} c_e x_e & \\ \textbf{subject to} & \sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \dots, k \\ & x_e \in \{0,1\} & e \in E \end{array}$$

where $\mathcal{S}_i := \{S \subseteq V : |S \cap \{s_i, t_i\}| = 1\}$

and $\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$



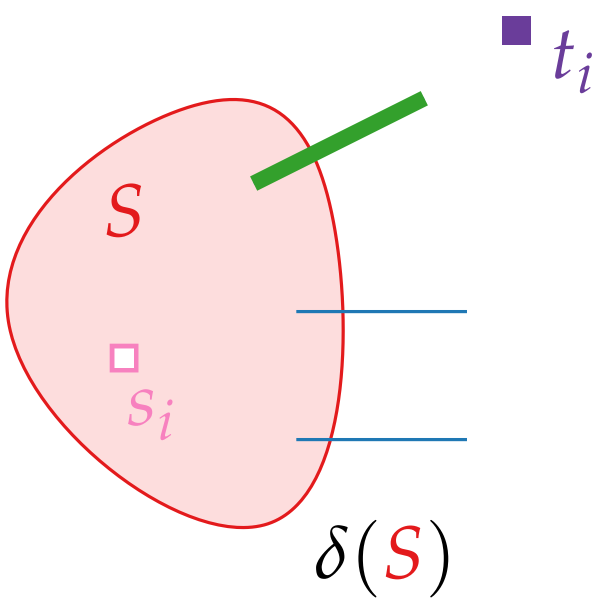
An ILP

$$\begin{array}{lll} \textbf{minimize} & \sum_{e \in E} c_e x_e & \\ \textbf{subject to} & \sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \dots, k \\ & x_e \in \{0,1\} & e \in E \end{array}$$

where $\mathcal{S}_i := \{S \subseteq V : |S \cap \{s_i, t_i\}| = 1\}$

and $\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$

$\rightsquigarrow$ exponentially many constraints!



LP-Relaxation and Dual LP

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{e \in E} c_e x_e & \\ \textbf{subject to} & \displaystyle\sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \ldots, k \\ & x_e \geq 0 & e \in E \end{array}$$

LP-Relaxation and Dual LP

$$
\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{e \in E} c_e x_e & \\
\textbf{subject to} & \displaystyle\sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \ldots, k \quad (y_S) \\
& x_e \geq 0 & e \in E
\end{array}
$$

LP-Relaxation and Dual LP

$$\begin{array}{lll} \textbf{minimize} & \sum_{e \in E} c_e x_e & \\ \textbf{subject to} & \sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \ldots, k \quad (y_S) \\ & x_e \geq 0 & e \in E \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & & \\ \textbf{subject to} & & \\ & y_S \geq 0 & S \in \mathcal{S}_i, i = 1, \ldots, k \end{array}$$

LP-Relaxation and Dual LP

$$
\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{e \in E} c_e x_e & \\
\textbf{subject to} & \displaystyle\sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \ldots, k \quad (y_S) \\
& x_e \geq 0 & e \in E
\end{array}
$$

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S & \\
\textbf{subject to} & & \\
& y_S \geq 0 & S \in \mathcal{S}_i, i = 1, \ldots, k
\end{array}
$$

LP-Relaxation and Dual LP

$$\begin{array}{lll} \textbf{minimize} & \sum_{e \in E} c_e x_e & \\ \textbf{subject to} & \sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S}_i, i = 1, \ldots, k \quad (y_S) \\ & x_e \geq 0 & e \in E \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S & \\ \textbf{subject to} & \sum_{S:\, e \in \delta(S)} y_S \leq c_e & e \in E \\ & y_S \geq 0 & S \in \mathcal{S}_i, i = 1, \ldots, k \end{array}$$

Intuition for the Dual

$$\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S & \\
\textbf{subject to} & \displaystyle\sum_{S:\, e \in \delta(S)} y_S \le c_e & e \in E \\
& y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \ldots, k
\end{array}$$

Intuition for the Dual

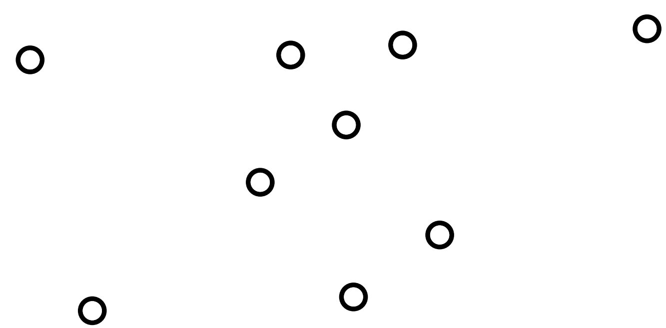
$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S & \\
\textbf{subject to} & \displaystyle\sum_{S:\, e \in \delta(S)} y_S \le c_e & e \in E \\
& y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \ldots, k
\end{array}
$$

The graph is a network of **bridges**, spanning the **moats**.

Intuition for the Dual

$$\begin{aligned} \textbf{maximize} \quad & \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\dots,k}} y_S \\ \textbf{subject to} \quad & \sum_{S:\, e \in \delta(S)} y_S \le c_e && e \in E \\ & y_S \ge 0 && S \in \mathcal{S}_i, i = 1, \dots, k \end{aligned}$$

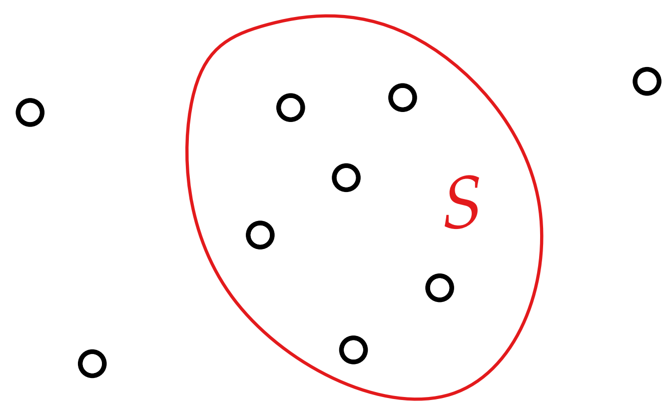
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{\substack{S\in\mathcal{S}_i\\ i=1,\ldots,k}} y_S & \\ \textbf{subject to} & \displaystyle\sum_{S:\, e\in\delta(S)} y_S \le c_e & e \in E \\ & y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \ldots, k \end{array}$$

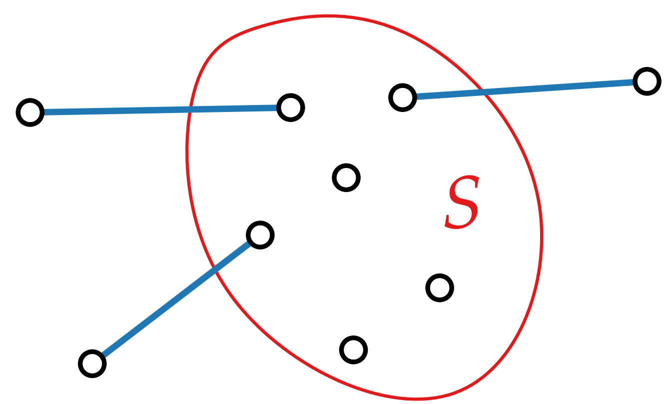
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

$$\begin{aligned}
\textbf{maximize} \quad & \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S \\
\textbf{subject to} \quad & \sum_{S:\, e \in \delta(S)} y_S \le c_e && e \in E \\
& y_S \ge 0 && S \in \mathcal{S}_i, i = 1, \ldots, k
\end{aligned}$$

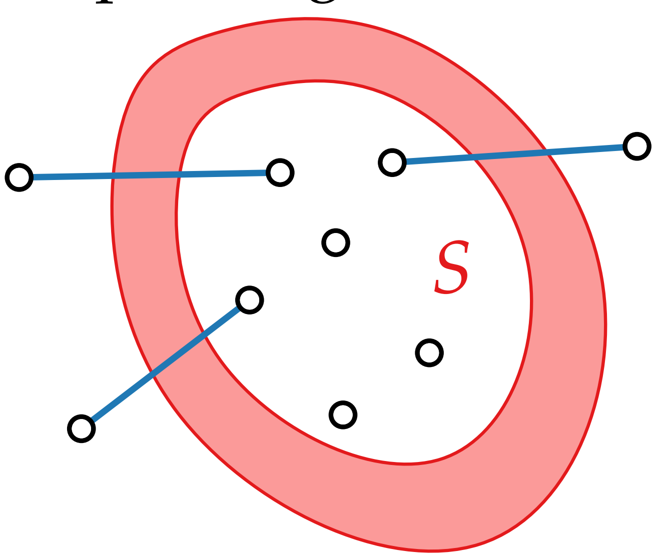
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

maximize	$\displaystyle\sum_{\substack{S\in\mathcal{S}_i \\ i=1,\dots,k}} y_S$	
subject to	$\displaystyle\sum_{S:\, e\in\delta(S)} y_S \le c_e$	$e \in E$
	$y_S \ge 0$	$S \in \mathcal{S}_i, i = 1, \dots, k$

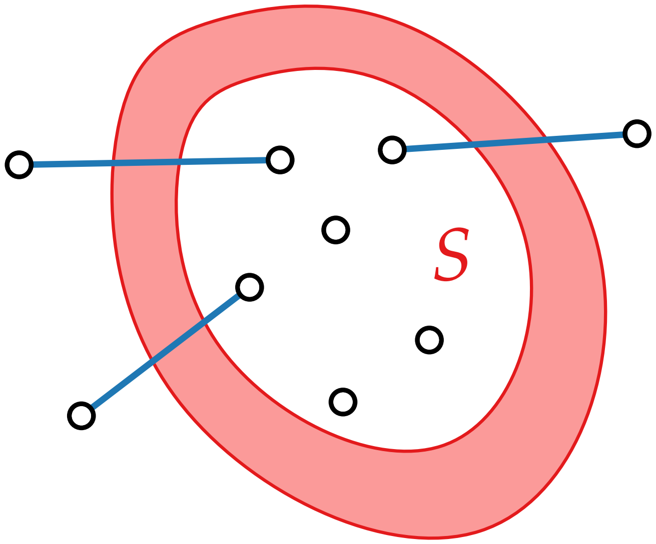
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S & \\ \textbf{subject to} & \displaystyle\sum_{S:\, e \in \delta(S)} y_S \le c_e & e \in E \\ & y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \ldots, k \end{array}$$

The graph is a network of **bridges**, spanning the **moats**.

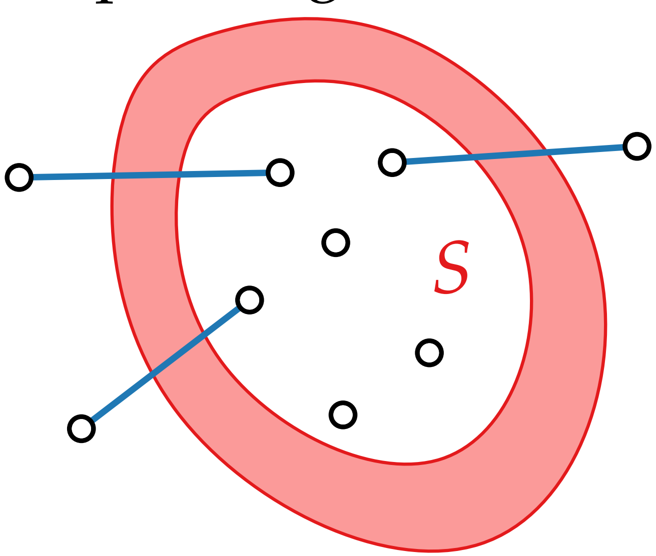


$\delta(S)$ = set of edges / bridges over the moat around S

Intuition for the Dual

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S & \\
\textbf{subject to} & \displaystyle\sum_{S:\, e \in \delta(S)} y_S \le c_e & e \in E \\
& y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \ldots, k
\end{array}
$$

The graph is a network of **bridges**, spanning the **moats**.



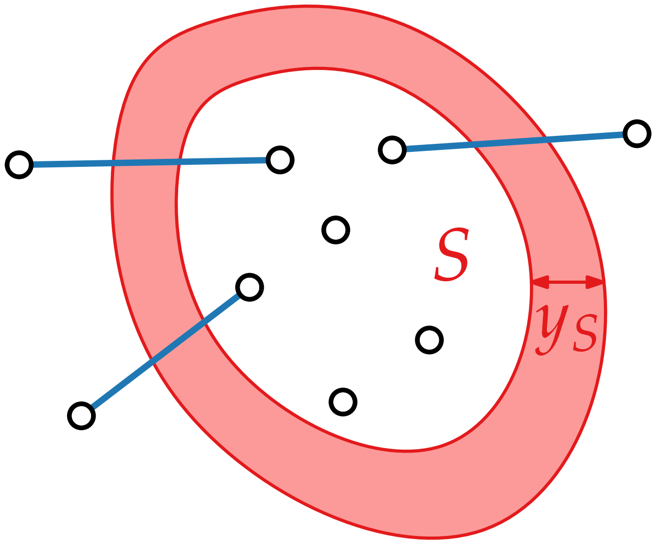
$\delta(S)$ = set of edges / bridges over the moat around S

y_S = width of the **moat** around S

Intuition for the Dual

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{\substack{S\in\mathcal{S}_i\\ i=1,\dots,k}} y_S & \\ \textbf{subject to} & \displaystyle\sum_{S:\, e\in\delta(S)} y_S \le c_e & e \in E \\ & y_S \ge 0 & S \in \mathcal{S}_i, i = 1,\dots,k \end{array}$$

The graph is a network of **bridges**, spanning the **moats**.



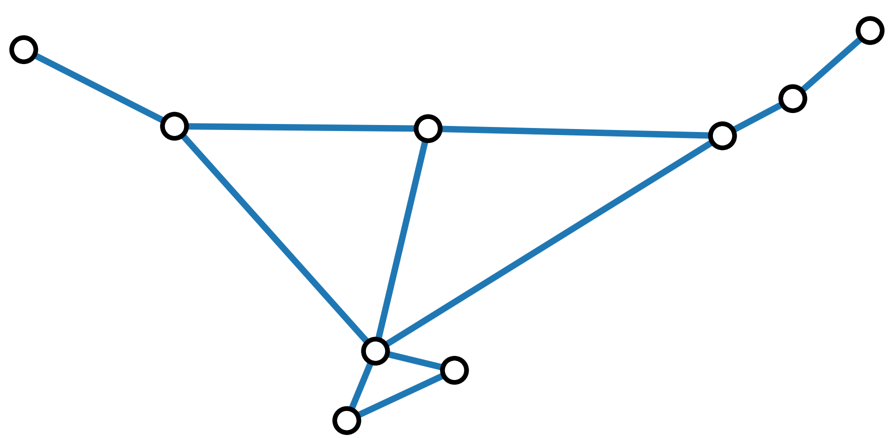
$\delta(S)$ = set of edges / bridges over the moat around S

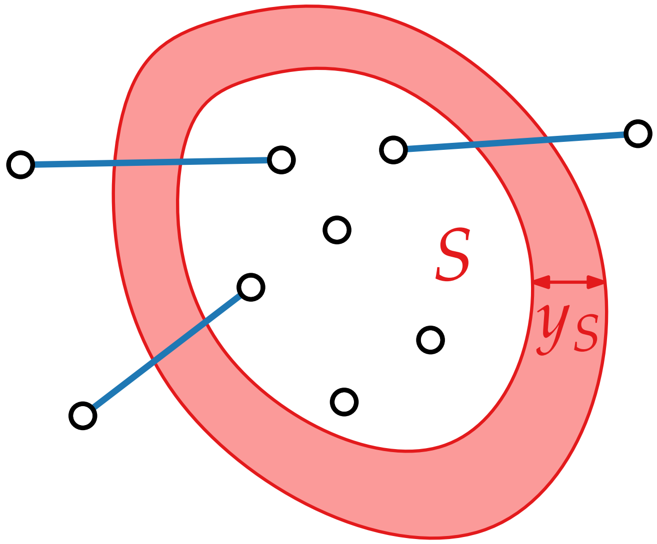
y_S = width of the **moat** around S

Intuition for the Dual

$$\begin{aligned} \textbf{maximize} \quad & \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S \\ \textbf{subject to} \quad & \sum_{S:\, e \in \delta(S)} y_S \le c_e \qquad e \in E \\ & y_S \ge 0 \qquad S \in \mathcal{S}_i, i = 1, \ldots, k \end{aligned}$$

The graph is a network of **bridges**, spanning the **moats**.





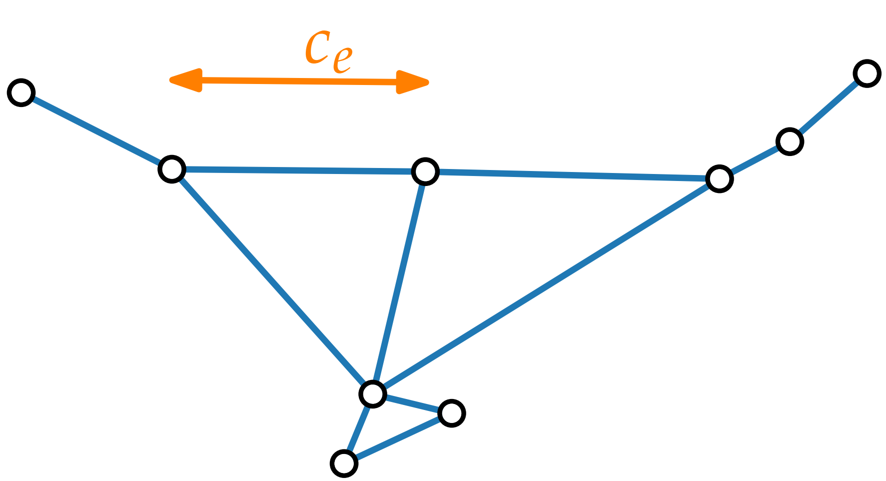
$\delta(S)$ = set of edges / bridges over the moat around S

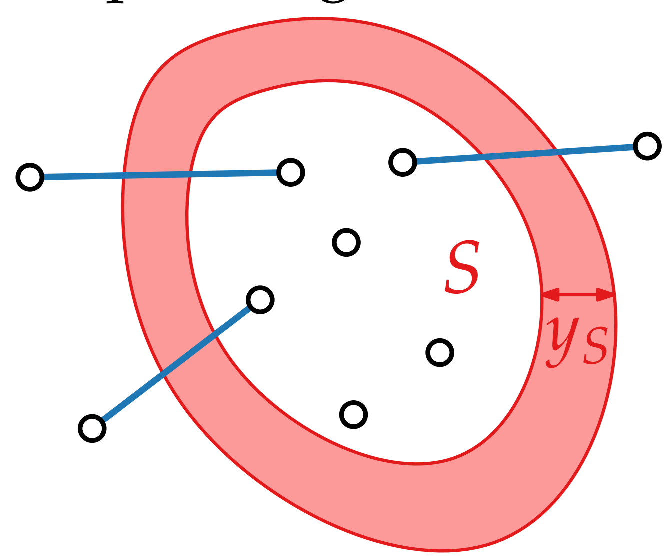
y_S = width of the **moat** around S

Intuition for the Dual

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{\substack{S\in\mathcal{S}_i\\ i=1,\ldots,k}} y_S & \\ \textbf{subject to} & \displaystyle\sum_{S:\, e\in\delta(S)} y_S \le c_e & e \in E \\ & y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \ldots, k \end{array}$$

The graph is a network of **bridges**, spanning the **moats**.





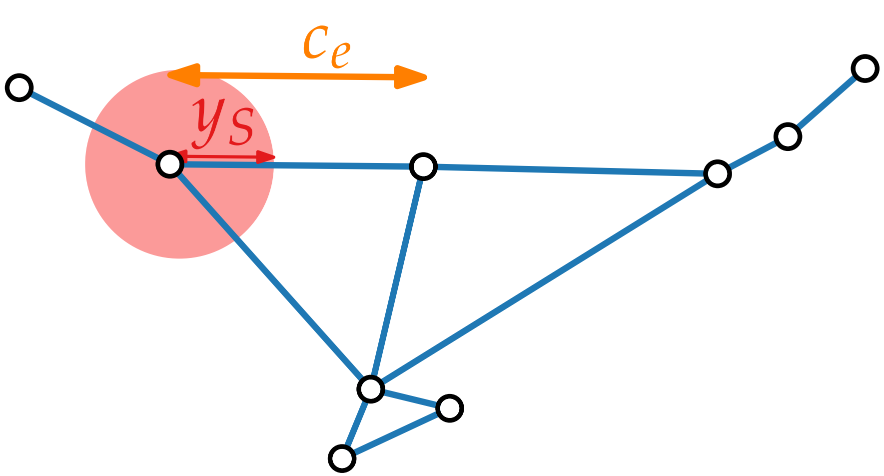
$\delta(S)$ = set of edges / bridges over the moat around S

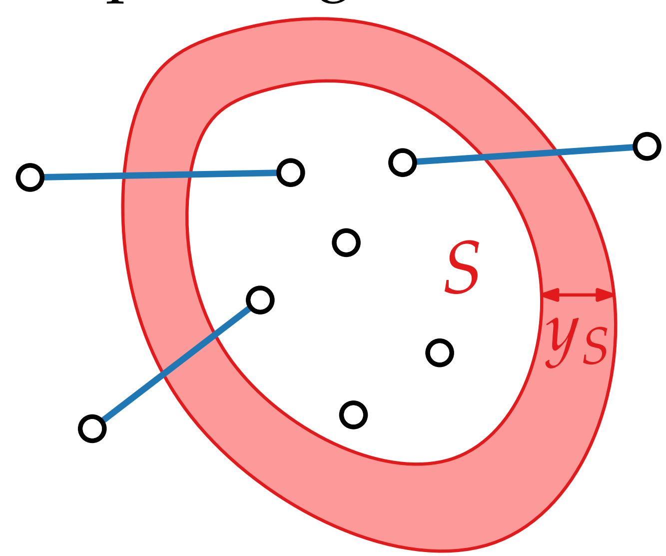
y_S = width of the **moat** around S

Intuition for the Dual

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\dots,k}} y_S & \\ \textbf{subject to} & \displaystyle\sum_{S:\, e \in \delta(S)} y_S \le c_e & e \in E \\ & y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \dots, k \end{array}$$

The graph is a network of **bridges**, spanning the **moats**.





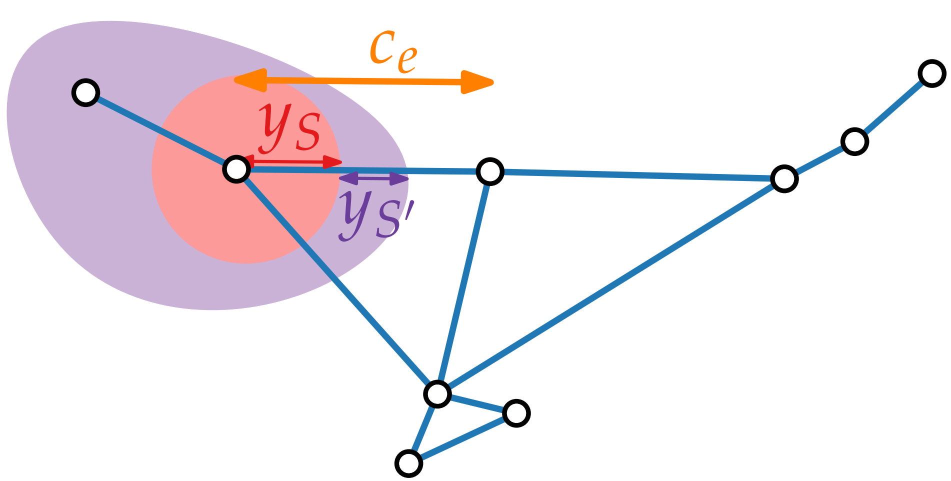
$\delta(S)$ = set of edges / bridges over the moat around S

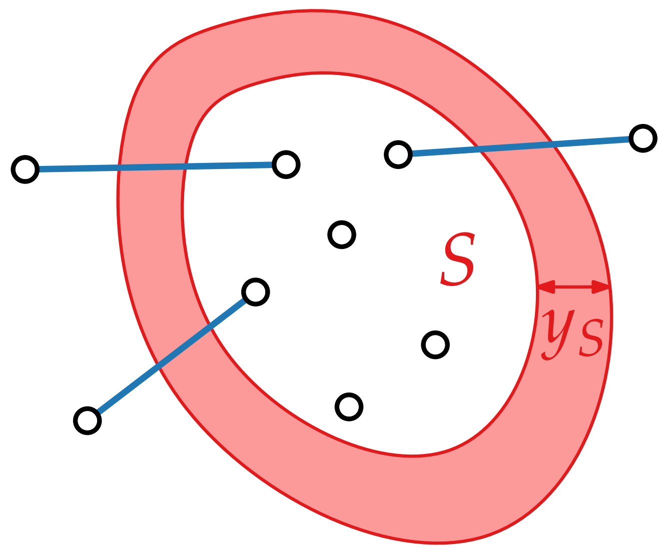
y_S = width of the **moat** around S

Intuition for the Dual

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\dots,k}} y_S & \\
\textbf{subject to} & \displaystyle\sum_{S:\, e \in \delta(S)} y_S \le c_e & e \in E \\
& y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \dots, k
\end{array}
$$

The graph is a network of **bridges**, spanning the **moats**.





$\delta(S)$ = set of edges / bridges over the moat around S

y_S = width of the **moat** around S

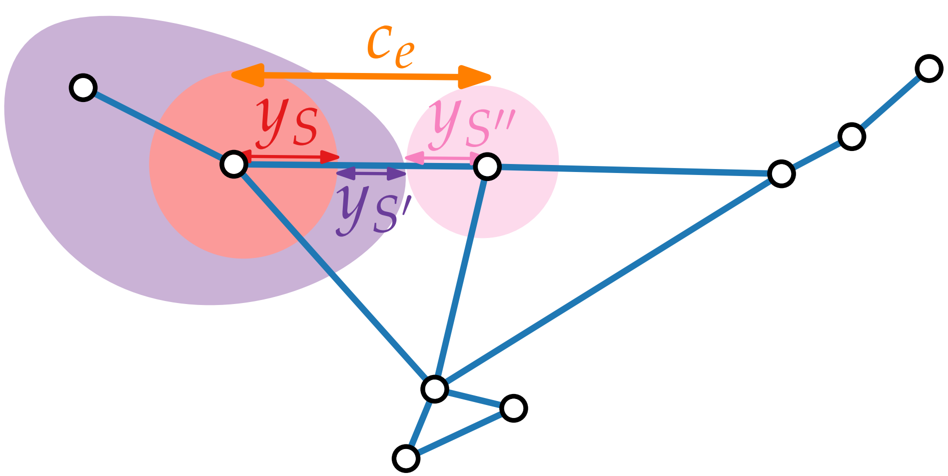
Intuition for the Dual

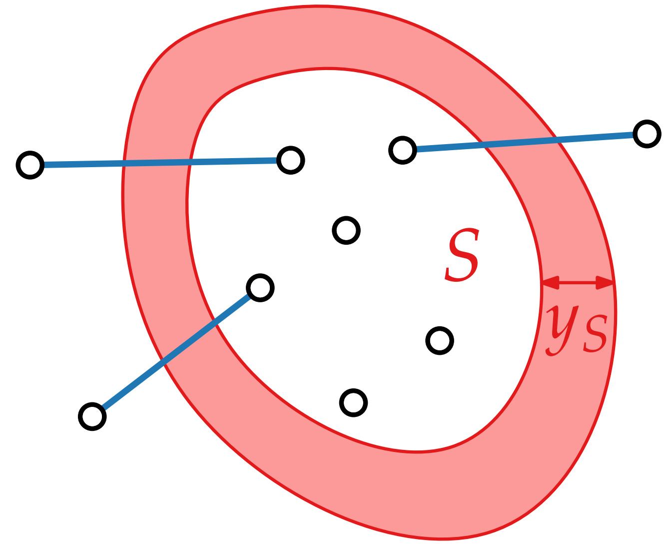
maximize $\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$

subject to $\sum_{S:\, e \in \delta(S)} y_S \le c_e \qquad e \in E$

$y_S \ge 0 \qquad S \in \mathcal{S}_i, i = 1, \ldots, k$

The graph is a network of **bridges**, spanning the **moats**.





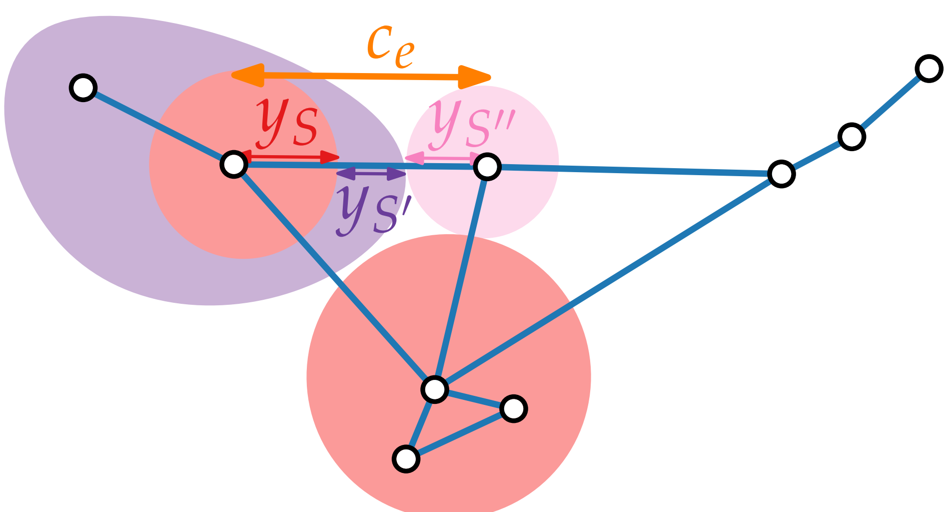
$\delta(S)$ = set of edges / bridges over the moat around S

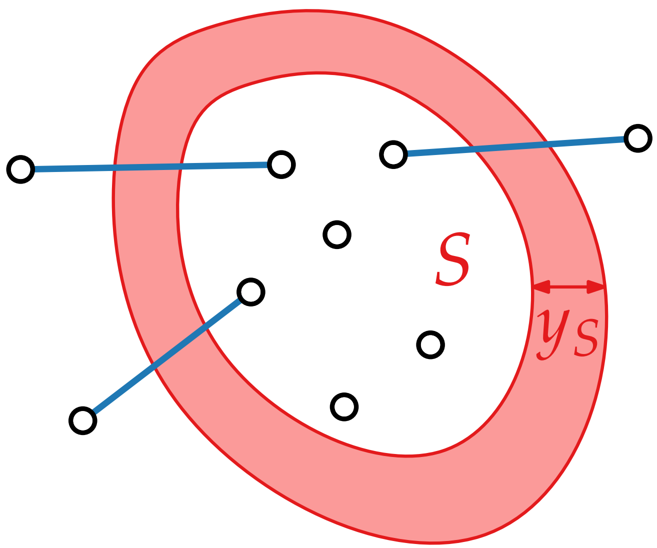
y_S = width of the **moat** around S

Intuition for the Dual

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{\substack{S\in\mathcal{S}_i\\ i=1,\ldots,k}} y_S & \\
\textbf{subject to} & \displaystyle\sum_{S:\, e\in\delta(S)} y_S \le c_e & e\in E \\
 & y_S \ge 0 & S\in\mathcal{S}_i, i=1,\ldots,k
\end{array}
$$

The graph is a network of **bridges**, spanning the **moats**.





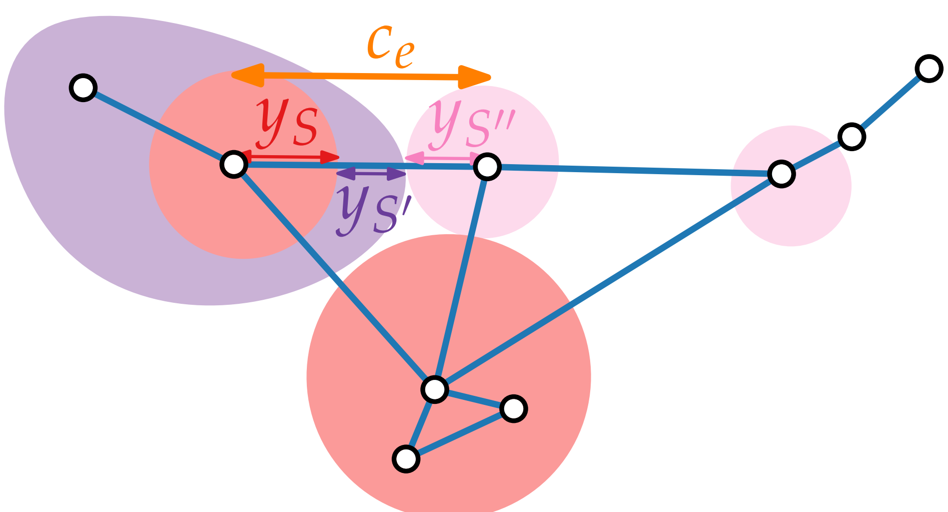
$\delta(S)$ = set of edges / bridges over the moat around S

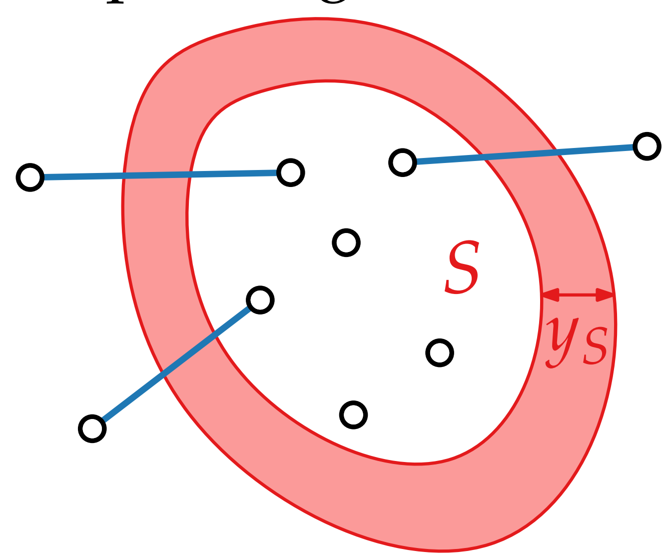
y_S = width of the **moat** around S

Intuition for the Dual

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{\substack{S\in\mathcal{S}_i \\ i=1,\ldots,k}} y_S & \\
\textbf{subject to} & \displaystyle\sum_{S:\, e\in\delta(S)} y_S \le c_e & e \in E \\
 & y_S \ge 0 & S \in \mathcal{S}_i, i = 1,\ldots,k
\end{array}
$$

The graph is a network of **bridges**, spanning the **moats**.





$\delta(S)$ = set of edges / bridges over the moat around S

y_S = width of the **moat** around S

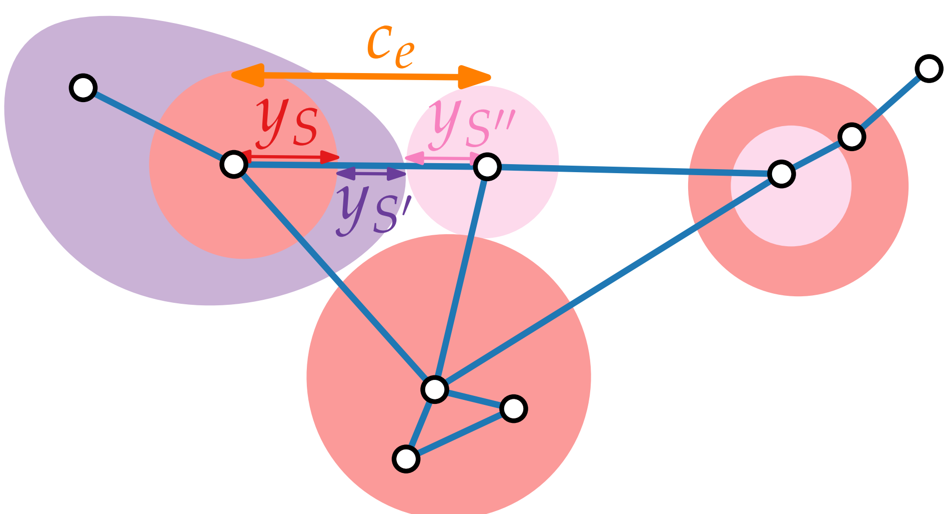
Intuition for the Dual

$$\text{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\dots,k}} y_S$$

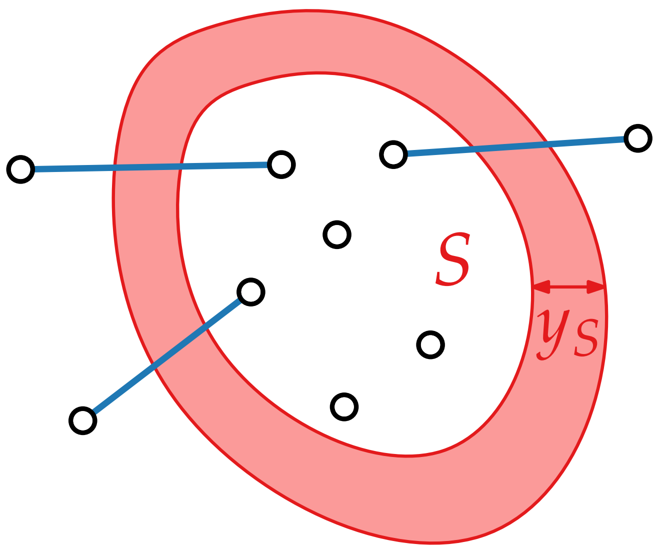
$$\text{subject to} \quad \sum_{S:\, e \in \delta(S)} y_S \le c_e \qquad e \in E$$

$$y_S \ge 0 \qquad S \in \mathcal{S}_i, i = 1, \dots, k$$

The graph is a network of **bridges**, spanning the **moats**.



y_S = width of the **moat** around S

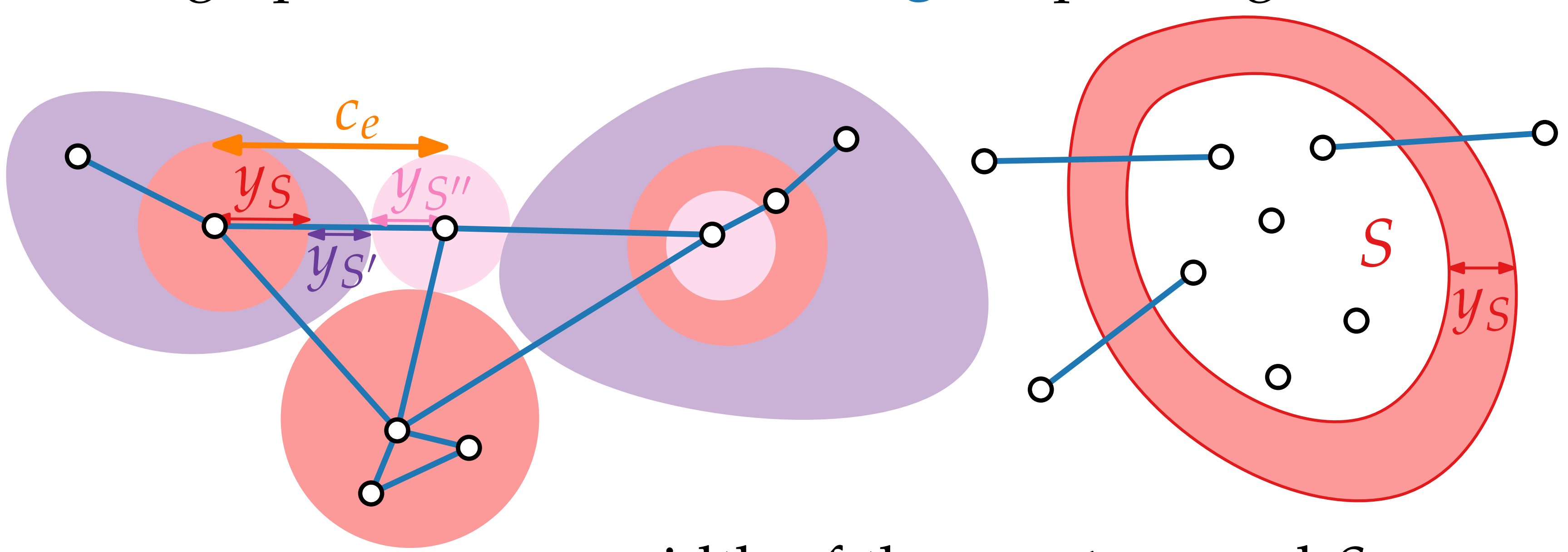


$\delta(S)$ = set of edges / bridges over the moat around S

Intuition for the Dual

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{\substack{S \in \mathcal{S}_i \\ i=1,\dots,k}} y_S & \\ \textbf{subject to} & \displaystyle\sum_{S:\, e \in \delta(S)} y_S \le c_e & e \in E \\ & y_S \ge 0 & S \in \mathcal{S}_i, i = 1, \dots, k \end{array}$$

The graph is a network of **bridges**, spanning the **moats**.



$\delta(S)$ = set of edges / bridges over the moat around S

y_S = width of the **moat** around S

Approximation Algorithms

Lecture 12:
SteinerForest via Primal-Dual

Part III:
A First Primal-Dual Approach

Philipp Kindermann

Summer Semester 2020

Complementary Slackness (Rep.)

$$\begin{array}{lrcl} \textbf{minimize} & c^\intercal x & & \\ \textbf{subject to} & Ax & \geq & b \\ & x & \geq & 0 \end{array}$$

$$\begin{array}{lrcl} \textbf{maximize} & b^\intercal y & & \\ \textbf{subject to} & A^\intercal y & \leq & c \\ & y & \geq & 0 \end{array}$$

Theorem. Let $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_m)$ be valid solutions for the primal and dual program (resp.). Then x and y are optimal if and only if the following conditions are met:

Primal CS:
For each $j = 1, \ldots, n$: either $x_j = 0$ or $\sum_{i=1}^{m} a_{ij} y_i = c_j$

Dual CS:
For each $i = 1, \ldots, m$: either $y_i = 0$ or $\sum_{j=1}^{n} a_{ij} x_j = b_i$

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \quad \Rightarrow$

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \;\Rightarrow\; \sum_{S:\, e \in \delta(S)} y_S = c_e$.

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \;\Rightarrow\; \sum_{S\,:\, e \in \delta(S)} y_S = c_e$.

$\Rightarrow$ pick "critical" edges (and only those)

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \Rightarrow \sum_{S:\, e \in \delta(S)} y_S = c_e$.

$\Rightarrow$ pick "critical" edges (and only those)

Idea: iteratively build a feasible integral Primal-Solution.

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \;\Rightarrow\; \sum_{S\,:\; e\in\delta(S)} y_S = c_e$.

$\Rightarrow$ pick "critical" edges (and only those)

Idea: iteratively build a feasible integral Primal-Solution.

How to find a violated primal constraint? $\left(\sum_{e\in\delta(S)} x_e < 1\right)$

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \Rightarrow \sum_{S:\, e \in \delta(S)} y_S = c_e$.

$\Rightarrow$ pick "critical" edges (and only those)

Idea: iteratively build a feasible integral Primal-Solution.

How to find a violated primal constraint? $(\sum_{e \in \delta(S)} x_e < 1)$

$\rightsquigarrow$ Consider related component C!

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \;\Rightarrow\; \sum_{S\colon e \in \delta(S)} y_S = c_e$.

$\Rightarrow$ pick "critical" edges (and only those)

Idea: iteratively build a feasible integral Primal-Solution.

How to find a violated primal constraint? $\left(\sum_{e \in \delta(S)} x_e < 1\right)$

$\rightsquigarrow$ Consider related component C!

How do we iteratively improve the Dual-Solution?

A First Primal-Dual Approach

Complementary slackness: $x_e > 0 \;\Rightarrow\; \sum_{S\,:\, e\in\delta(S)} y_S = c_e$.

$\Rightarrow$ pick "critical" edges (and only those)

Idea: iteratively build a feasible integral Primal-Solution.

How to find a violated primal constraint? $\left(\sum_{e\in\delta(S)} x_e < 1\right)$

$\rightsquigarrow$ Consider related component C!

How do we iteratively improve the Dual-Solution?

$\rightsquigarrow$ increase y_C! (until some edge in $\delta(C)$ becomes critical)

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$

return F

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$

while some $(s_i, t_i) \in R$ not connected in (V, F) **do**

return F

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $C \leftarrow$ comp. in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i

return F

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $C \leftarrow$ comp. in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i
  Increase y_C
return F

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$

while some $(s_i, t_i) \in R$ not connected in (V, F) **do**

  $C \leftarrow$ comp. in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i

  Increase y_C

    until $\sum_{S\colon e' \in \delta(S)} y_S = c_{e'}$ for some $e' \in \delta(C)$.

return F

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$

while some $(s_i, t_i) \in R$ not connected in (V, F) **do**

- $C \leftarrow$ comp. in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i
- Increase y_C
  until $\sum_{S:\, e' \in \delta(S)} y_S = c_{e'}$ for some $e' \in \delta(C)$.
- $F \leftarrow F \cup \{e'\}$

return F

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$

while some $(s_i, t_i) \in R$ not connected in (V, F) **do**

  $C \leftarrow$ comp. in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i

  Increase y_C

    until $\sum_{S:\, e' \in \delta(S)} y_S = c_{e'}$ for some $e' \in \delta(C)$.

  $F \leftarrow F \cup \{e'\}$

return F

Running Time?

A First Primal-Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $C \leftarrow$ comp. in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i
  Increase y_C
    until $\sum_{S:\, e' \in \delta(S)} y_S = c_{e'}$ for some $e' \in \delta(C)$.
  $F \leftarrow F \cup \{e'\}$
return F

Running Time?

Trick: Handle all y_S with $y_S = 0$ implicitly

Analysis

The cost of the solution F can be written as

Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e =$$

Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F}$$

Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S \,:\, e \in \delta(S)} y_S =$$

Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S:\, e \in \delta(S)} y_S = \sum_{S} |\delta(S) \cap F| \cdot y_S.$$

Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S \,:\, e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S:\, e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

There are examples with $|\delta(S) \cap F| = k$ for each $y_S > 0$:

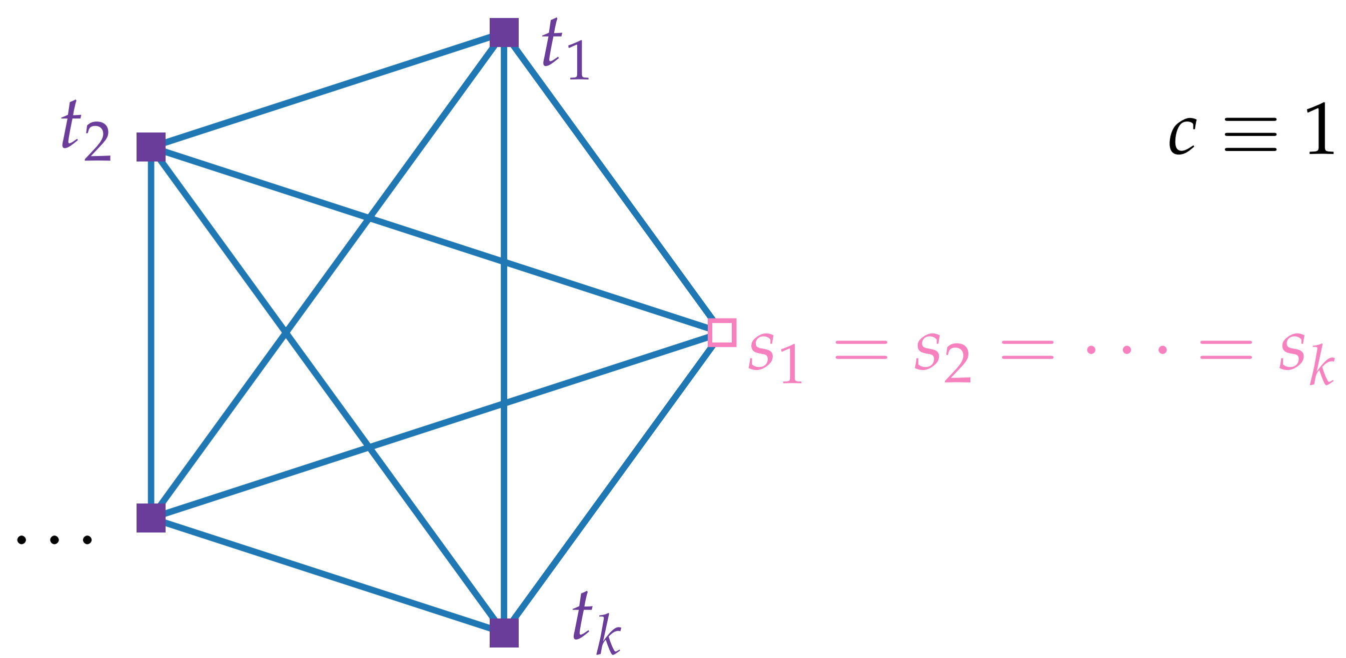
Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \stackrel{\text{CS}}{=} \sum_{e \in F} \sum_{S \colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

There are examples with $|\delta(S) \cap F| = k$ for each $y_S > 0$:



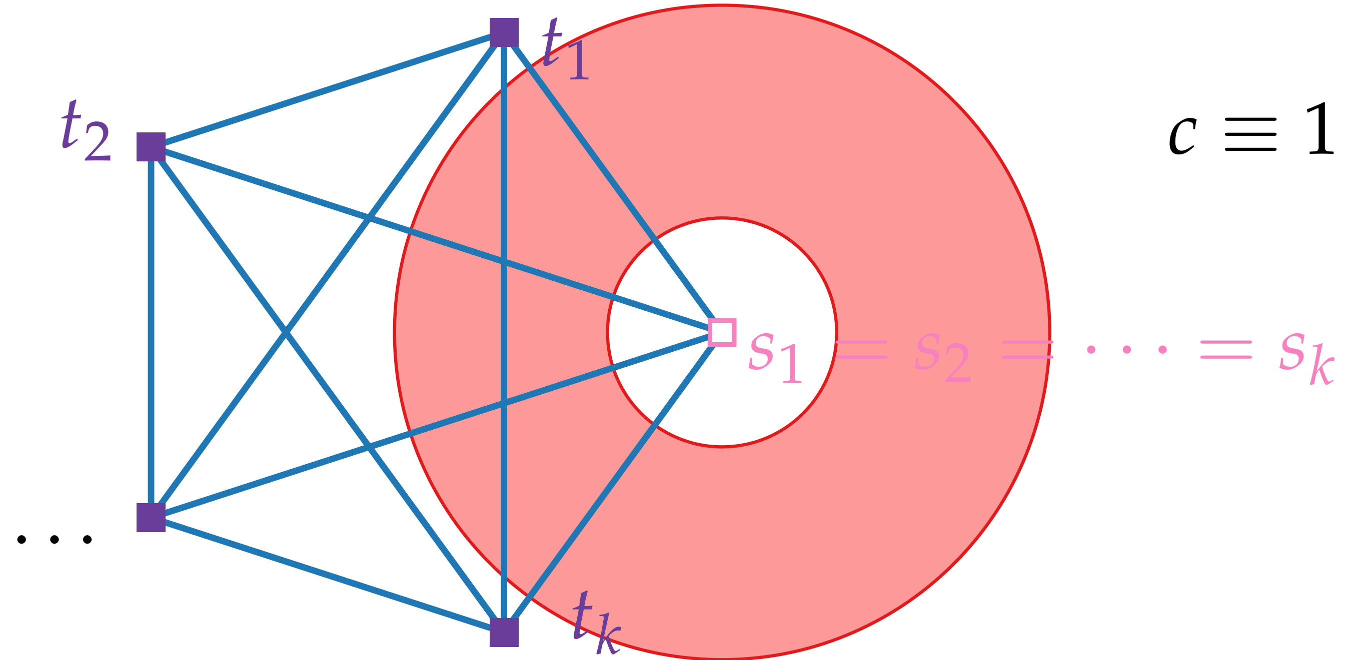
Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S\,:\, e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

There are examples with $|\delta(S) \cap F| = k$ for each $y_S > 0$:



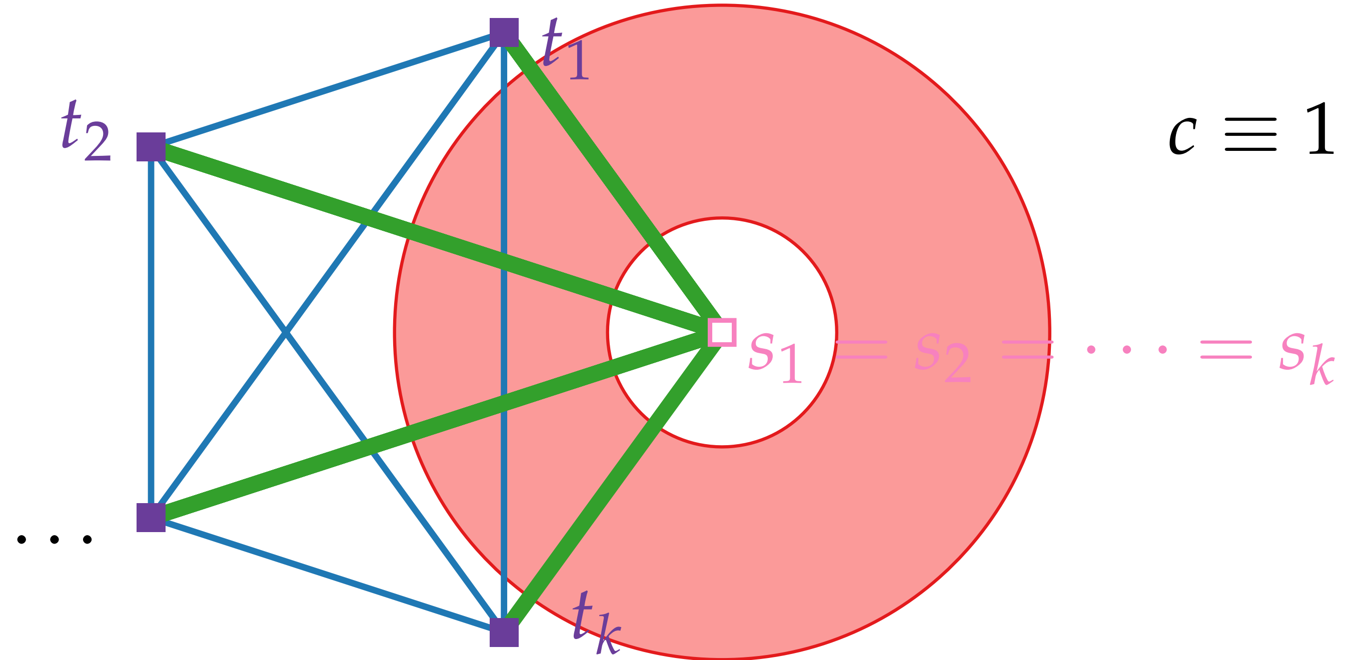
Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \stackrel{\text{CS}}{=} \sum_{e \in F} \sum_{S \colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

There are examples with $|\delta(S) \cap F| = k$ for each $y_S > 0$:



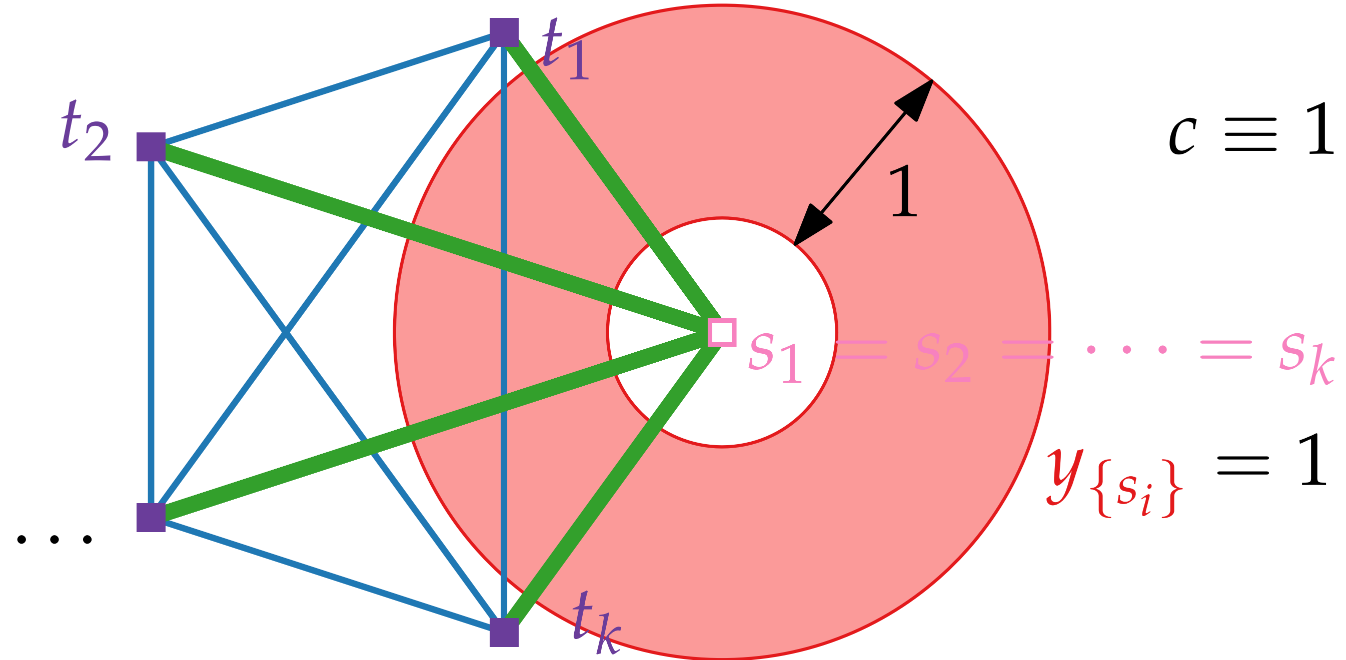
Analysis

The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S\colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

There are examples with $|\delta(S) \cap F| = k$ for each $y_S > 0$:



Analysis

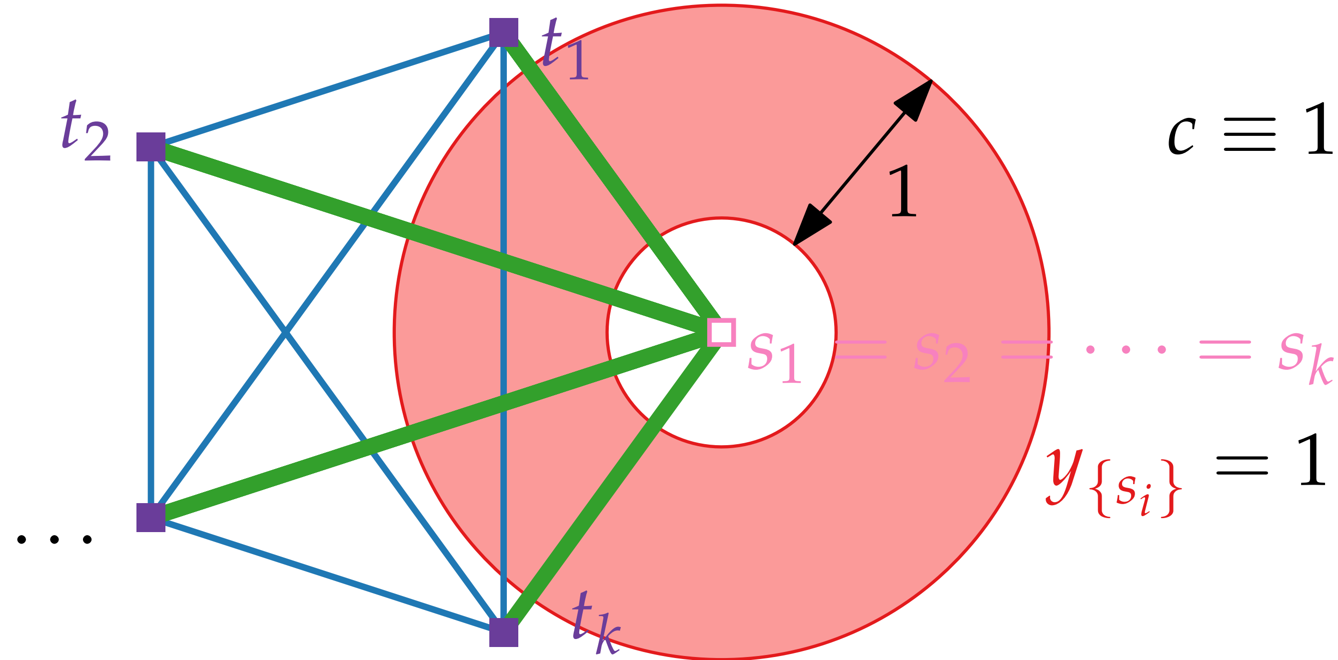
The cost of the solution F can be written as

$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S\colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

There are examples with $|\delta(S) \cap F| = k$ for each $y_S > 0$:

But: Average degree of component is 2!



Analysis

The cost of the solution F can be written as

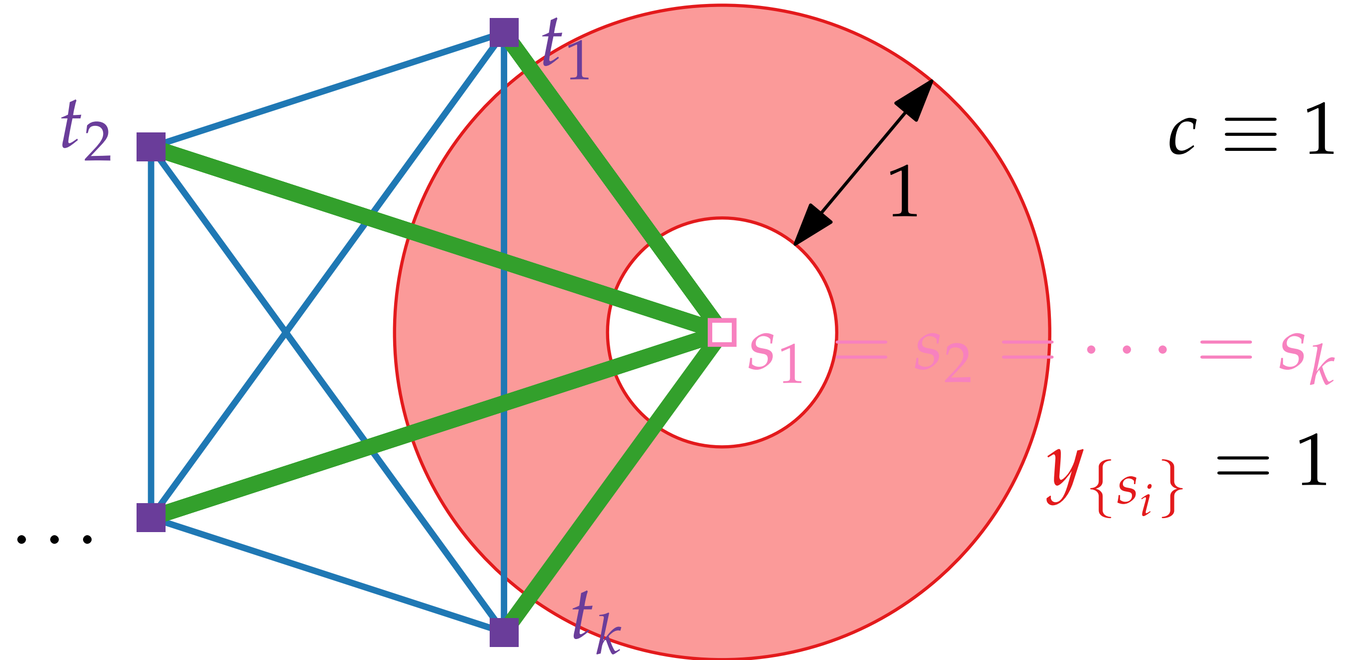
$$\sum_{e \in F} c_e \stackrel{\text{CS}}{=} \sum_{e \in F} \sum_{S\colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Compare to the value of the dual objective function $\sum_S y_S$

There are examples with $|\delta(S) \cap F| = k$ for each $y_S > 0$:

But: Average degree of component is 2!

$\Rightarrow$ Increase y_C for all components C simultaneously!



Approximation Algorithms

Lecture 12:
SteinerForest via Primal-Dual

Part IV:
Primal-Dual with Synchronized Increases

Philipp Kindermann

Summer Semester 2020

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$

  $F \leftarrow F \cup \{e_\ell\}$

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$
  $\mathcal{C} \leftarrow \{$comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some $i\}$

  $F \leftarrow F \cup \{e_\ell\}$

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$
  $\mathcal{C} \leftarrow \{$comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some $i\}$
  Increase y_C for all $C \in \mathcal{C}$ simultaneously

  $F \leftarrow F \cup \{e_\ell\}$

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
- $\ell \leftarrow \ell + 1$
- $\mathcal{C} \leftarrow \{$comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some $i\}$
- Increase y_C for all $C \in \mathcal{C}$ simultaneously
  until $\sum_{S\,:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.
- $F \leftarrow F \cup \{e_\ell\}$

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$

while some $(s_i, t_i) \in R$ not connected in (V, F) **do**

$\ell \leftarrow \ell + 1$

$\mathcal{C} \leftarrow \{\text{comp. } C \text{ in } (V, F) \text{ with } |C \cap \{s_i, t_i\}| = 1 \text{ for some } i\}$

Increase y_C for all $C \in \mathcal{C}$ simultaneously

until $\sum_{S \colon e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.

$F \leftarrow F \cup \{e_\ell\}$

$F' \leftarrow F$

return F'

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$
  $\mathcal{C} \leftarrow \{\text{comp. } C \text{ in } (V, F) \text{ with } |C \cap \{s_i, t_i\}| = 1 \text{ for some } i\}$
  Increase y_C for all $C \in \mathcal{C}$ simultaneously
    until $\sum_{S:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.
  $F \leftarrow F \cup \{e_\ell\}$
$F' \leftarrow F$
// Pruning

return F'

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$
  $\mathcal{C} \leftarrow \{$comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some $i\}$
  Increase y_C for all $C \in \mathcal{C}$ simultaneously
    until $\displaystyle\sum_{S\,:\,e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C), C \in \mathcal{C}$.
  $F \leftarrow F \cup \{e_\ell\}$
$F' \leftarrow F$
// Pruning
for $j \leftarrow \ell$ **down to** 1 **do**

return F'

Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$
  $\mathcal{C} \leftarrow \{$comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some $i\}$
  Increase y_C for all $C \in \mathcal{C}$ simultaneously
    until $\sum_{S \colon e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C), C \in \mathcal{C}$.
  $F \leftarrow F \cup \{e_\ell\}$
$F' \leftarrow F$
// Pruning
for $j \leftarrow \ell$ **down to** 1 **do**
  if $F' \setminus \{e_j\}$ is feasible solution **then**

return F'

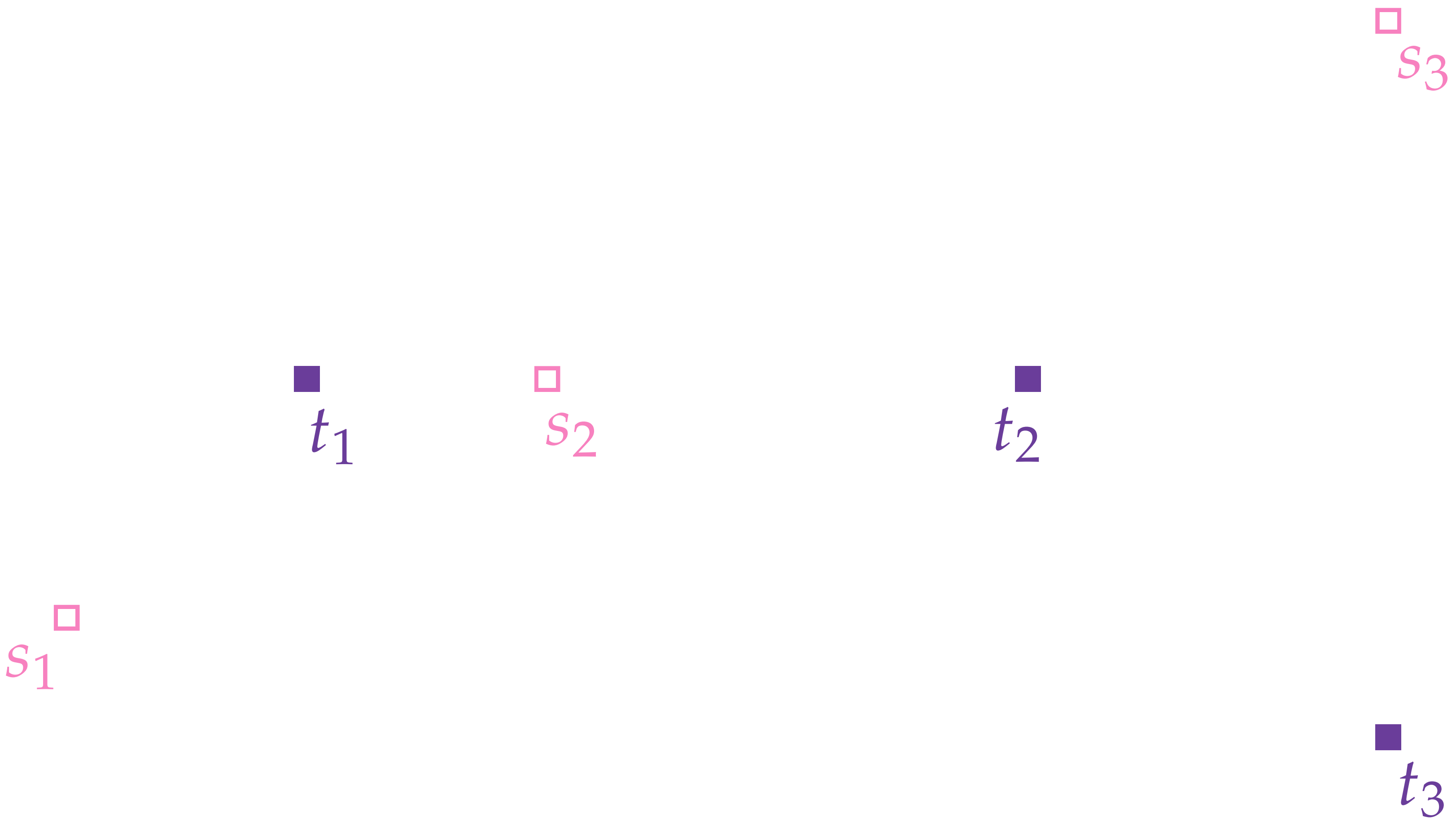
Primal-Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

$y \leftarrow 0, F \leftarrow \emptyset, \ell \leftarrow 0$
while some $(s_i, t_i) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$
  $\mathcal{C} \leftarrow \{$comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some $i\}$
  Increase y_C for all $C \in \mathcal{C}$ simultaneously
    until $\displaystyle\sum_{S \colon e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C), C \in \mathcal{C}$.
  $F \leftarrow F \cup \{e_\ell\}$
$F' \leftarrow F$
// Pruning
for $j \leftarrow \ell$ **down to** 1 **do**
  if $F' \setminus \{e_j\}$ is feasible solution **then**
    $F' \leftarrow F' \setminus \{e_j\}$
return F'

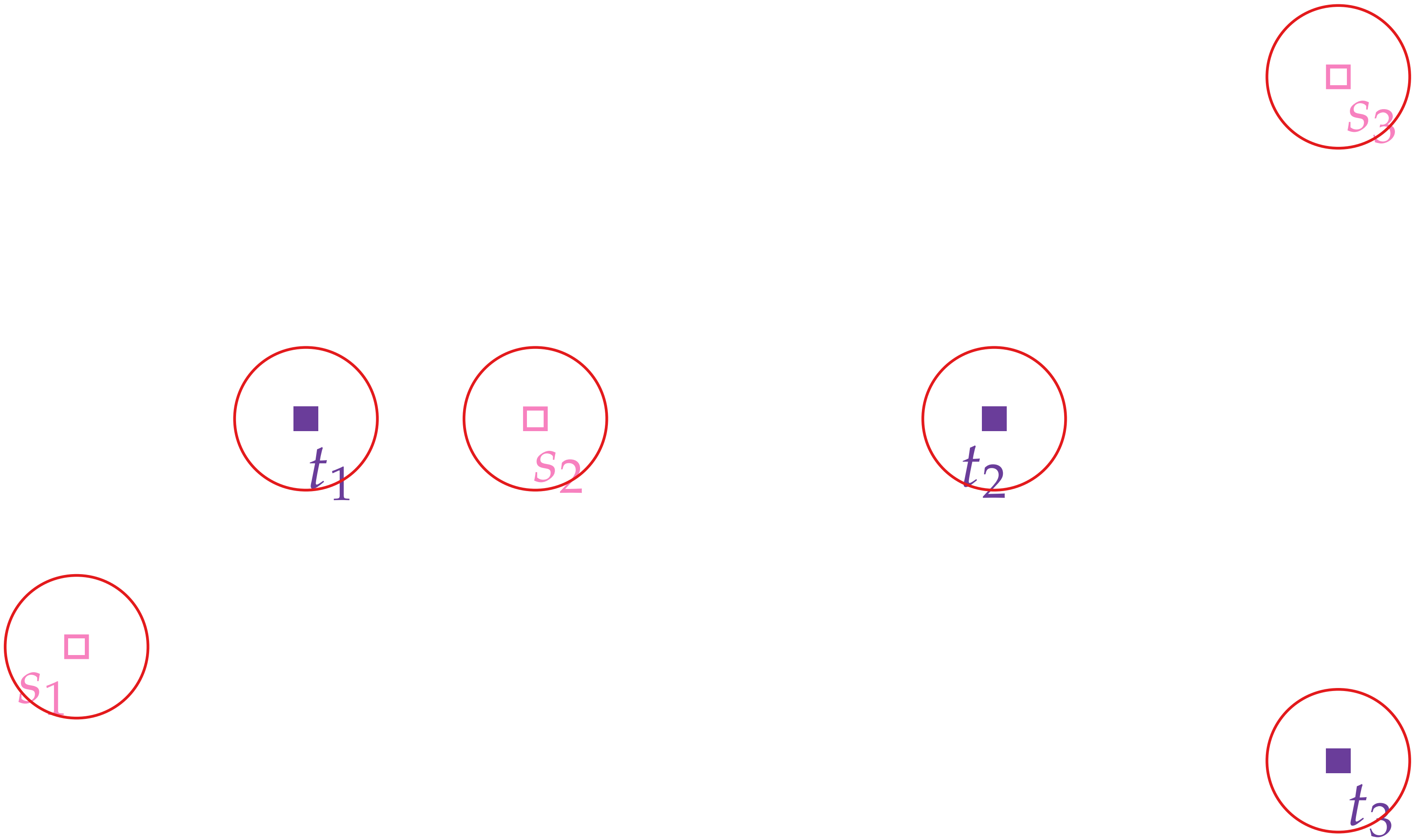
Illustration

$G = K_6$ with Euclidean edge costs



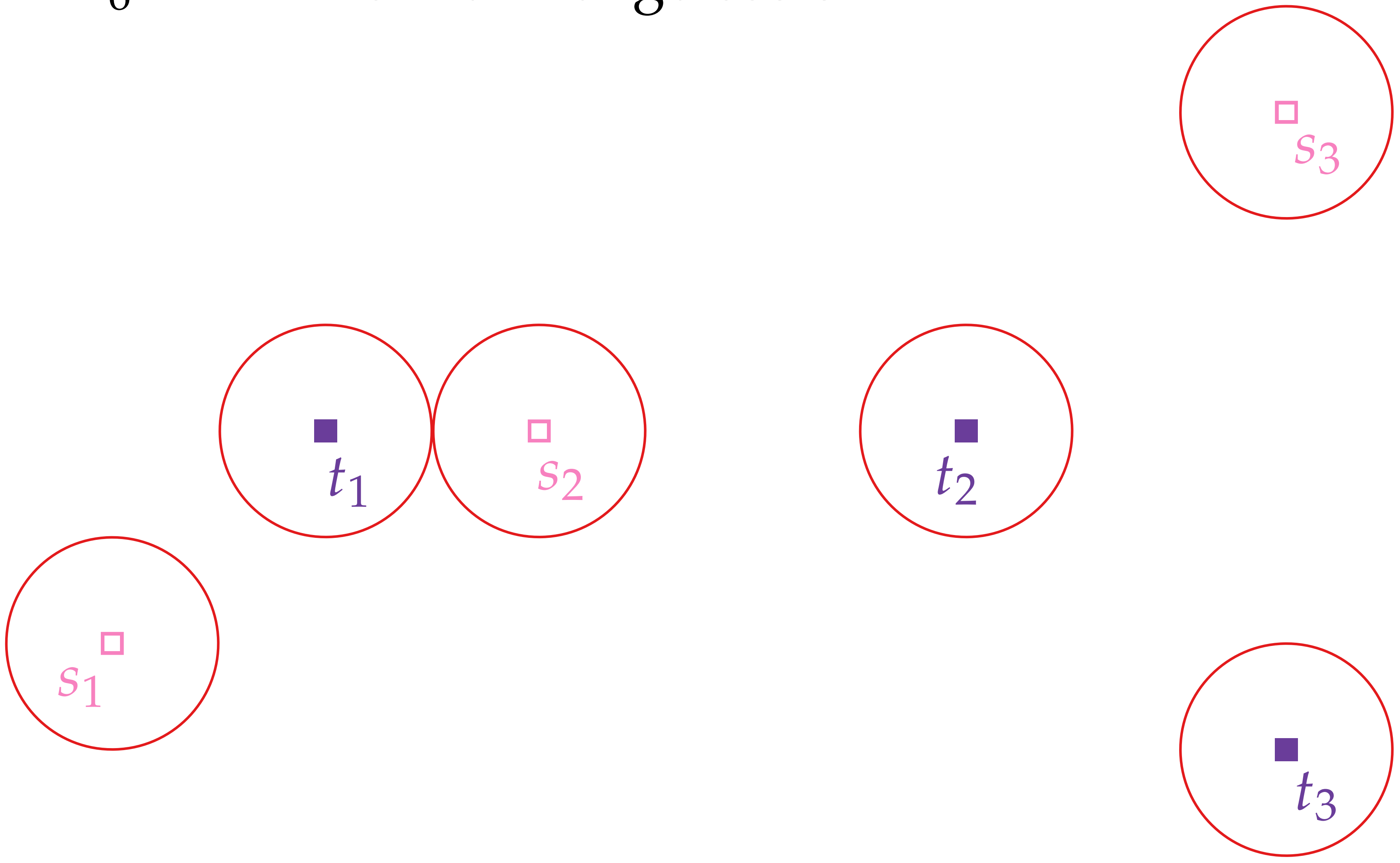
Illustration

$G = K_6$ with Euclidean edge costs



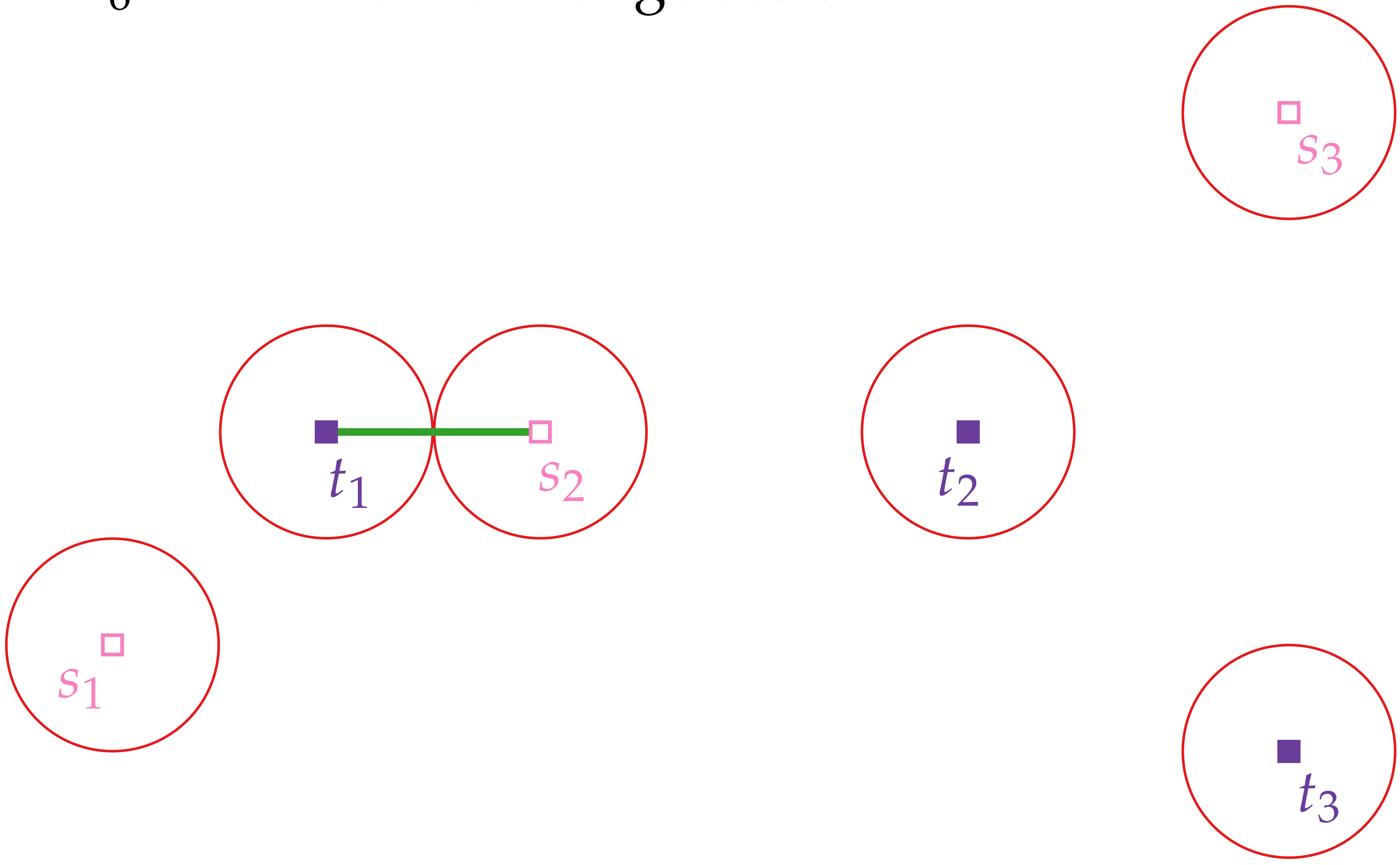
Illustration

$G = K_6$ with Euclidean edge costs



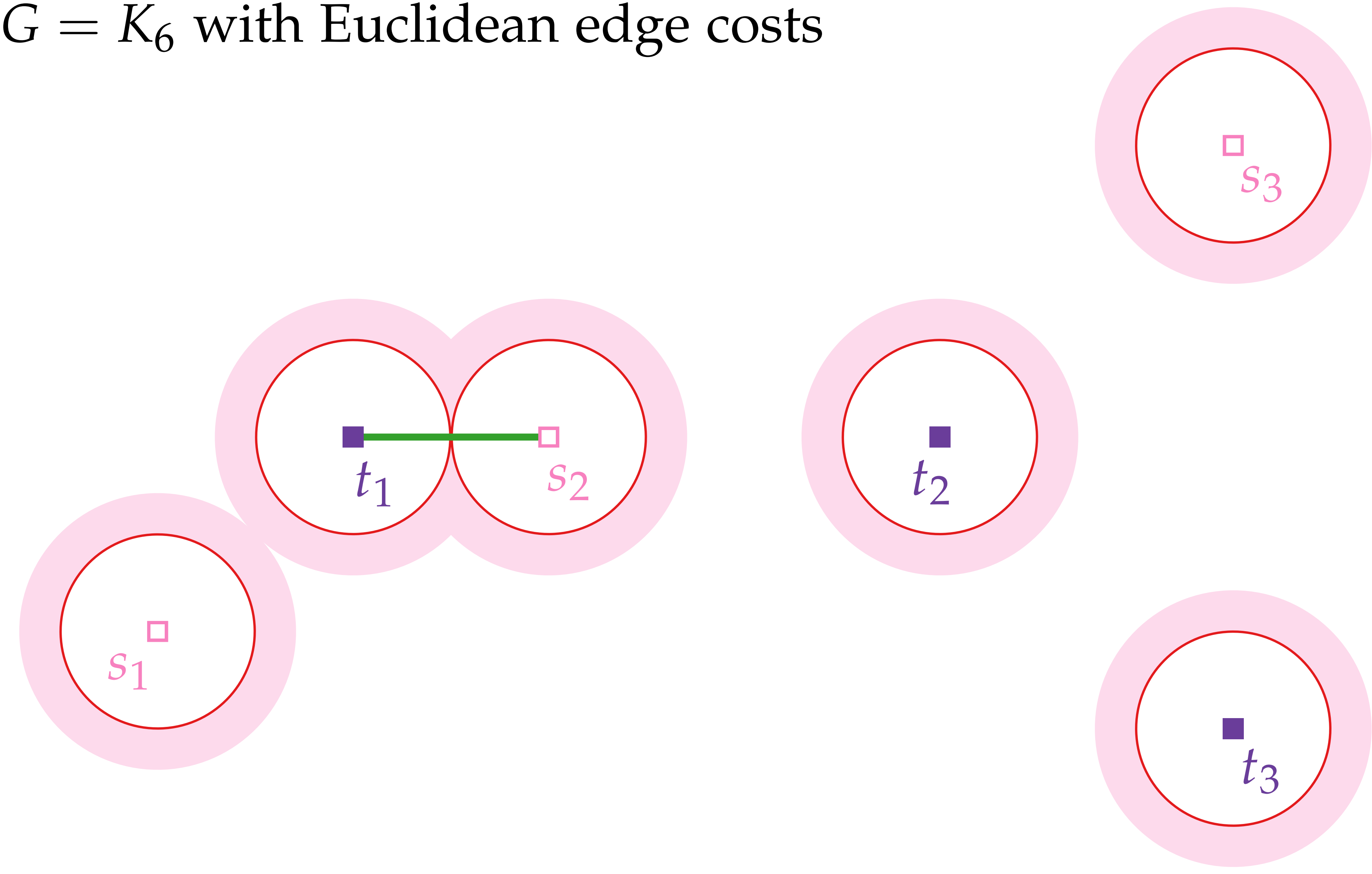
Illustration

$G = K_6$ with Euclidean edge costs



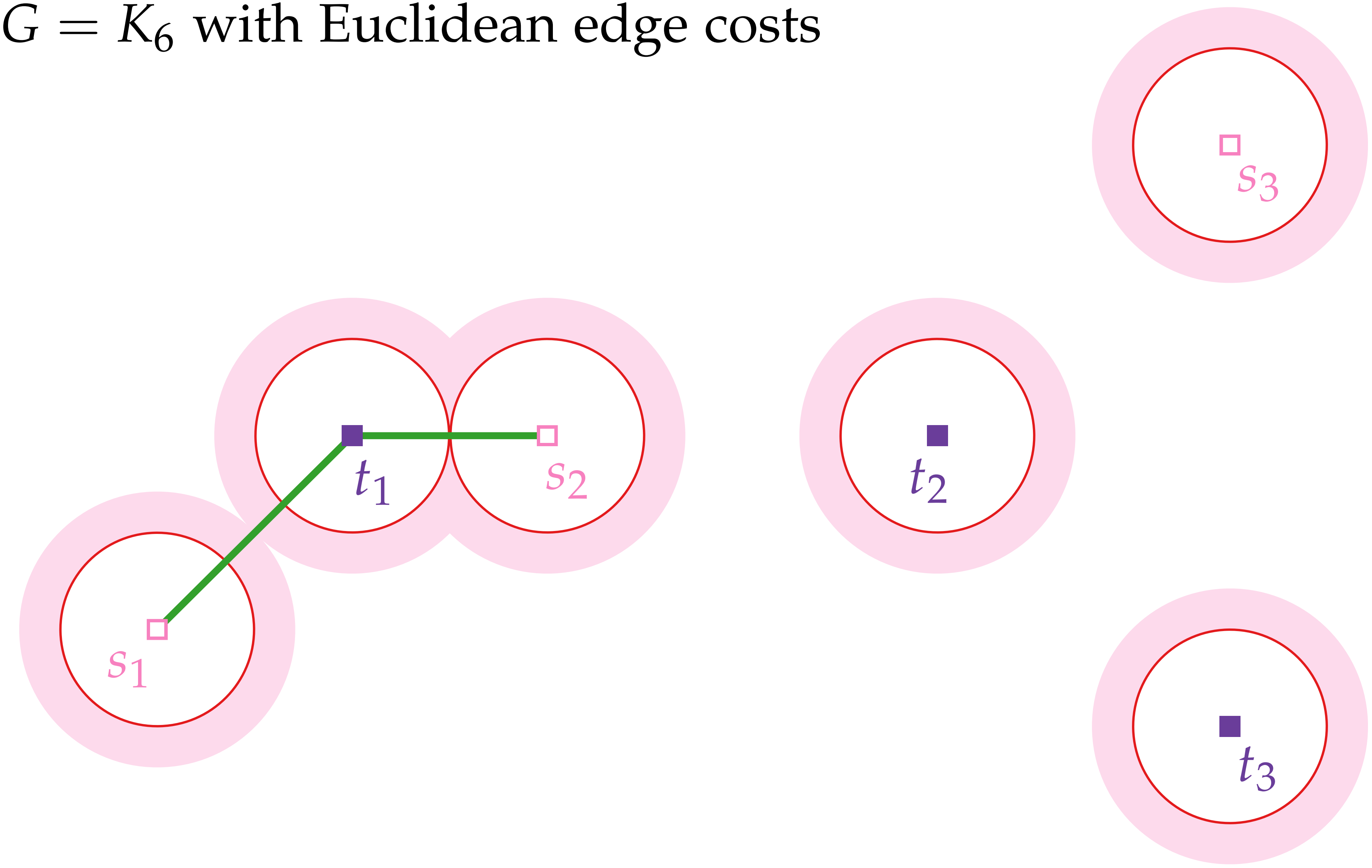
Illustration

$G = K_6$ with Euclidean edge costs



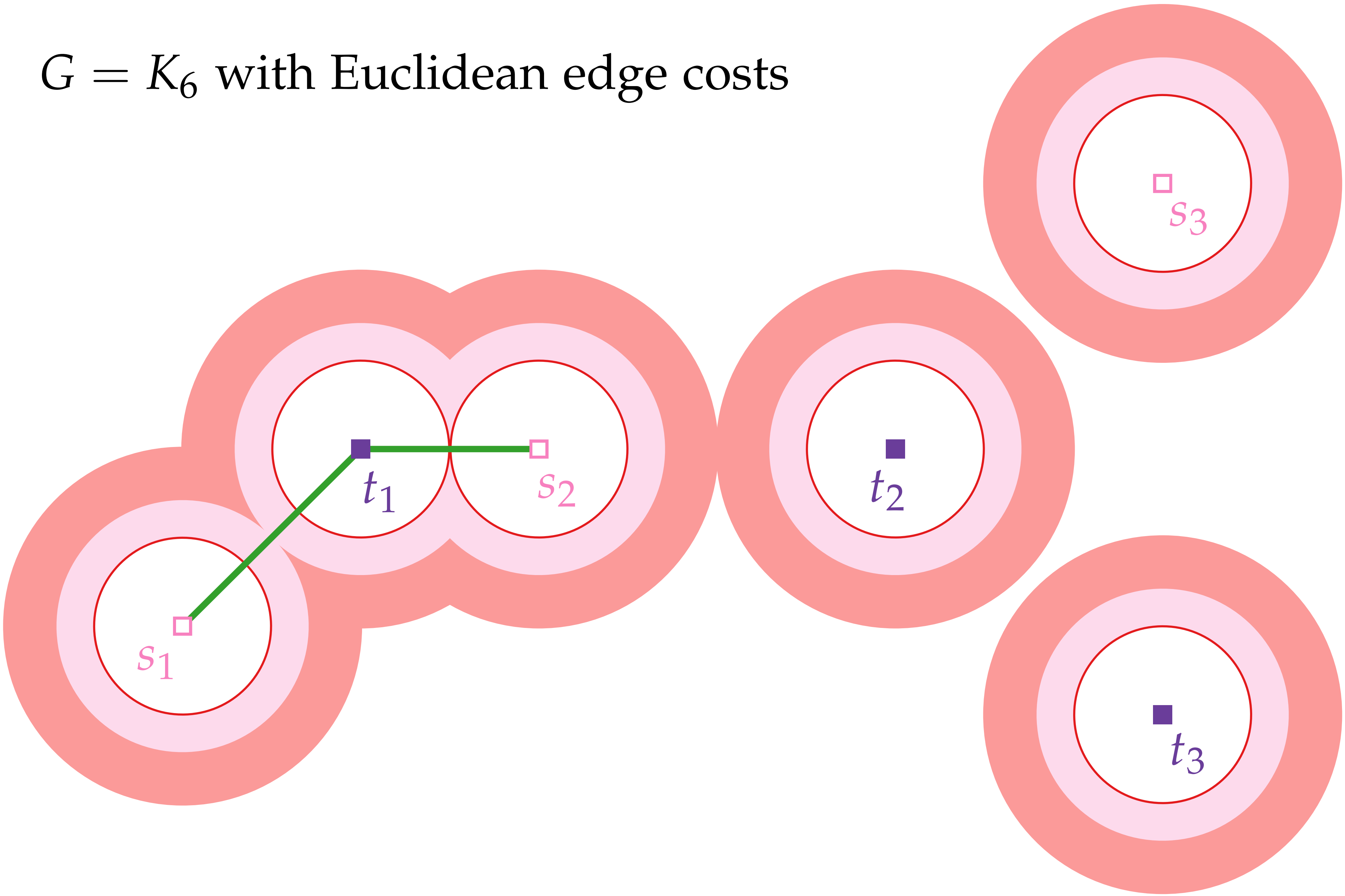
Illustration

$G = K_6$ with Euclidean edge costs



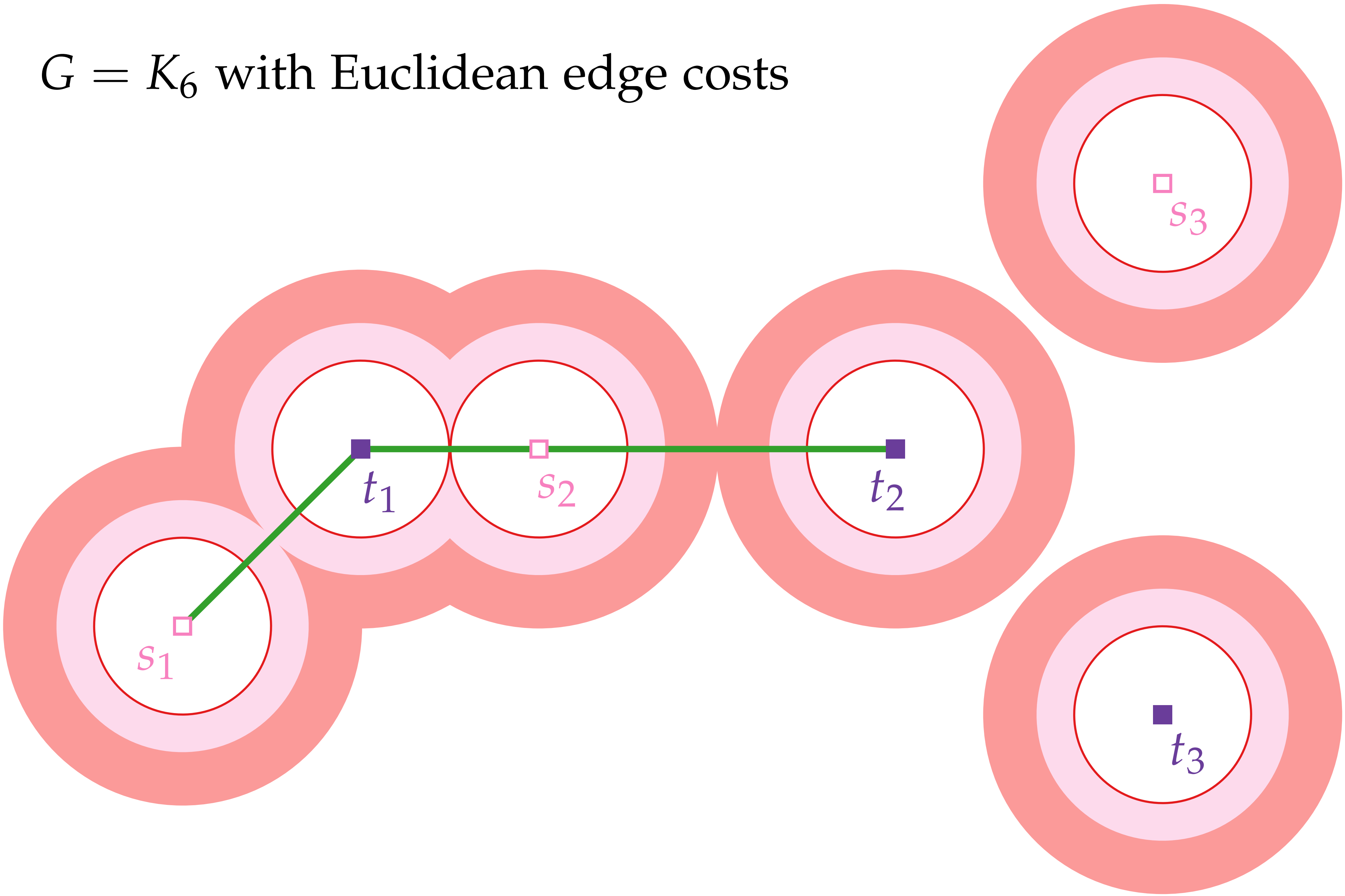
Illustration

$G = K_6$ with Euclidean edge costs



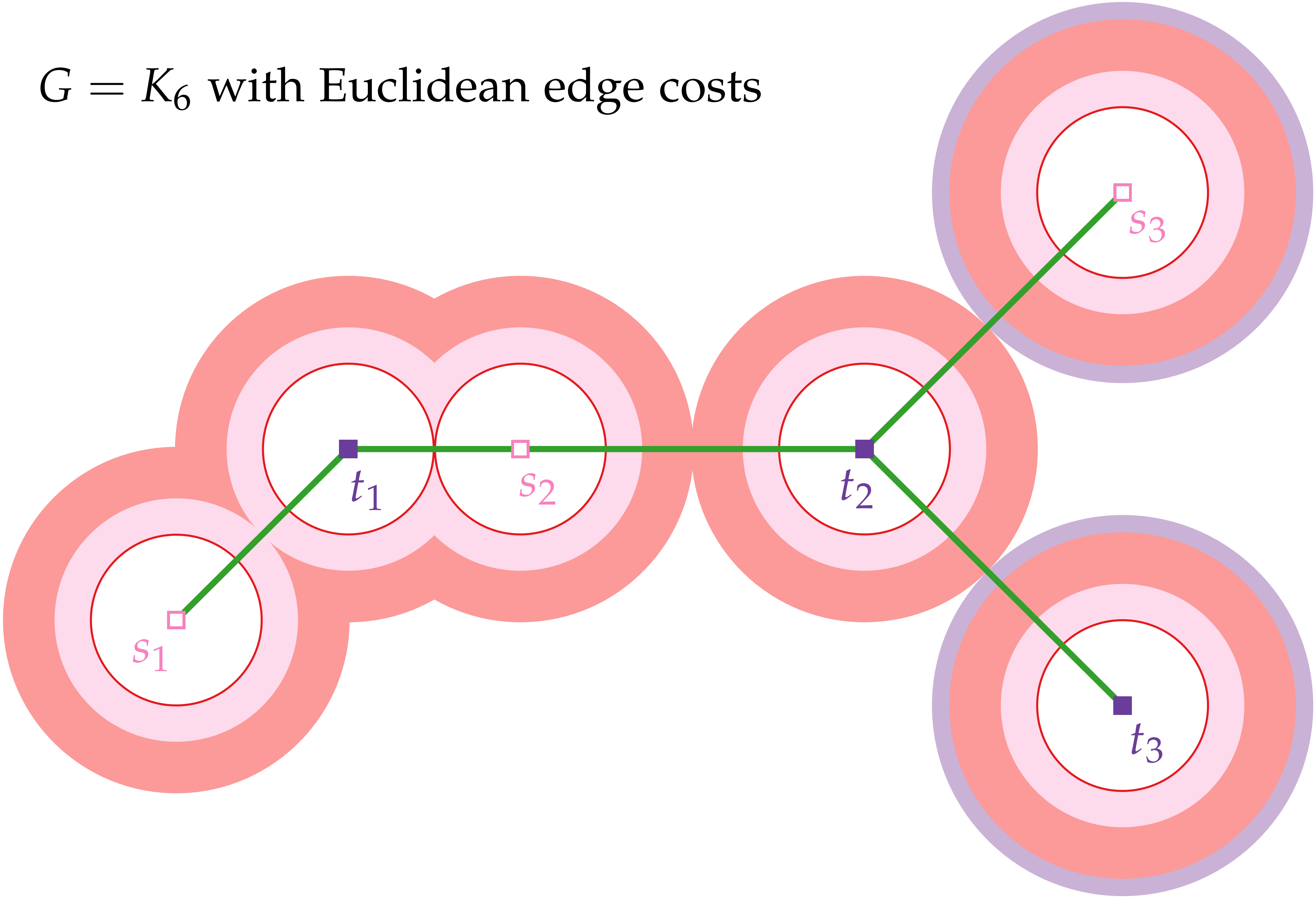
Illustration

$G = K_6$ with Euclidean edge costs



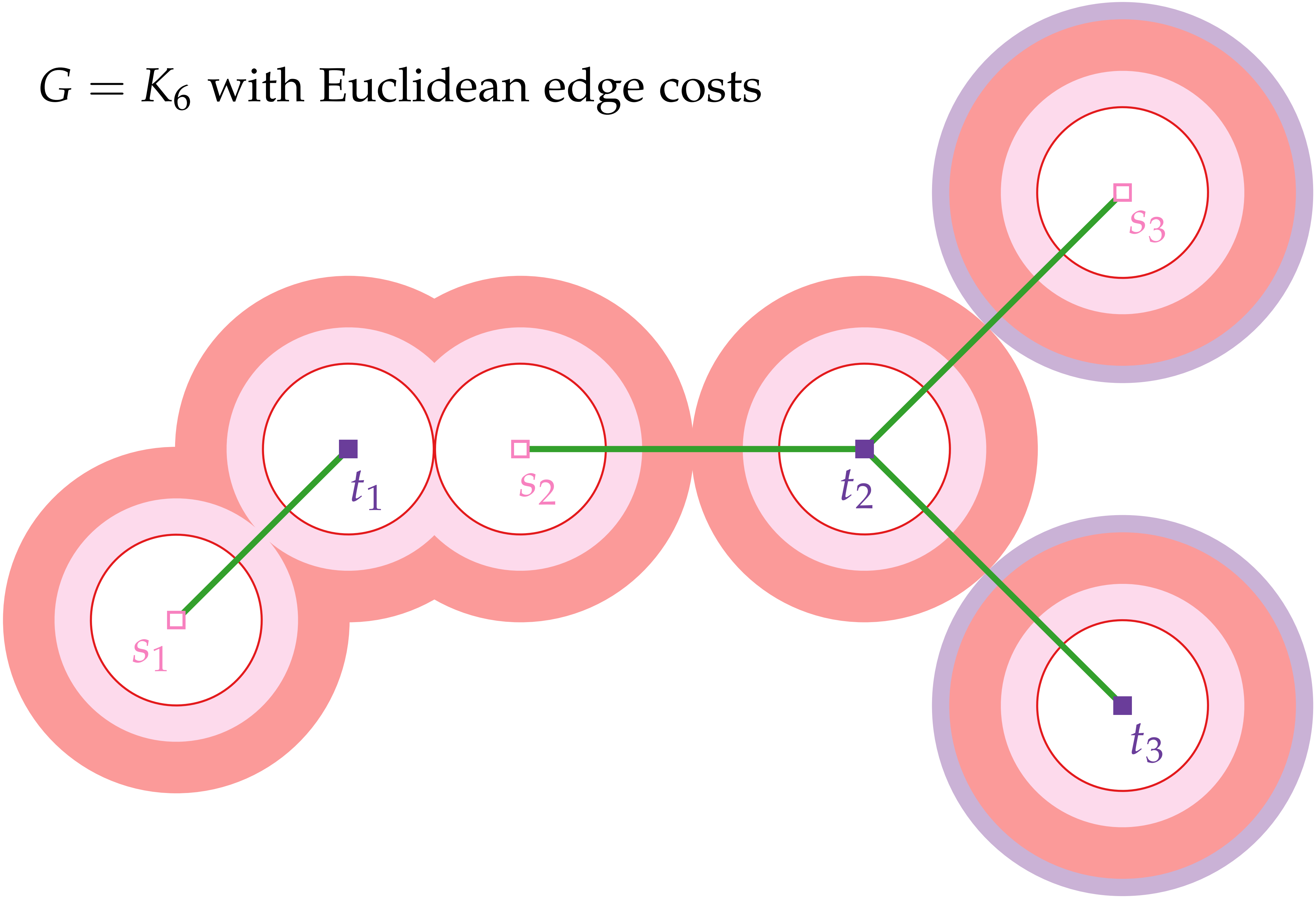
Illustration

$G = K_6$ with Euclidean edge costs



Illustration

$G = K_6$ with Euclidean edge costs



Approximation Algorithms

Lecture 12:
SteinerForest via Primal-Dual

Part V:
Structure Lemma

Philipp Kindermann

Summer Semester 2020

Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq$$

Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

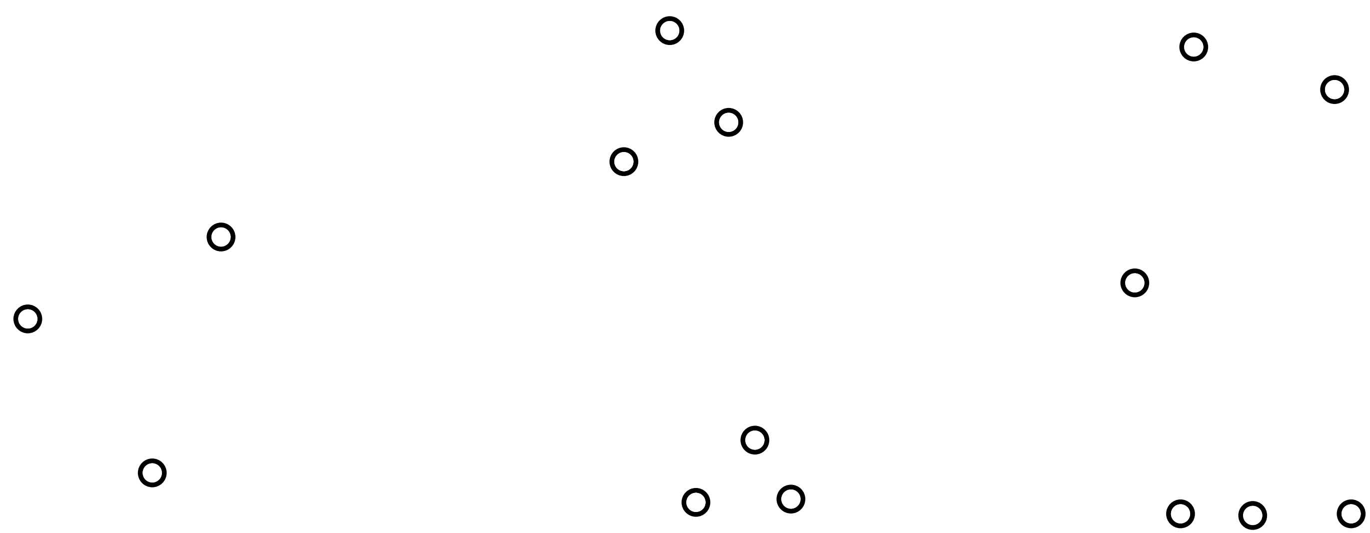
Proof. First the intuition…

Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition…

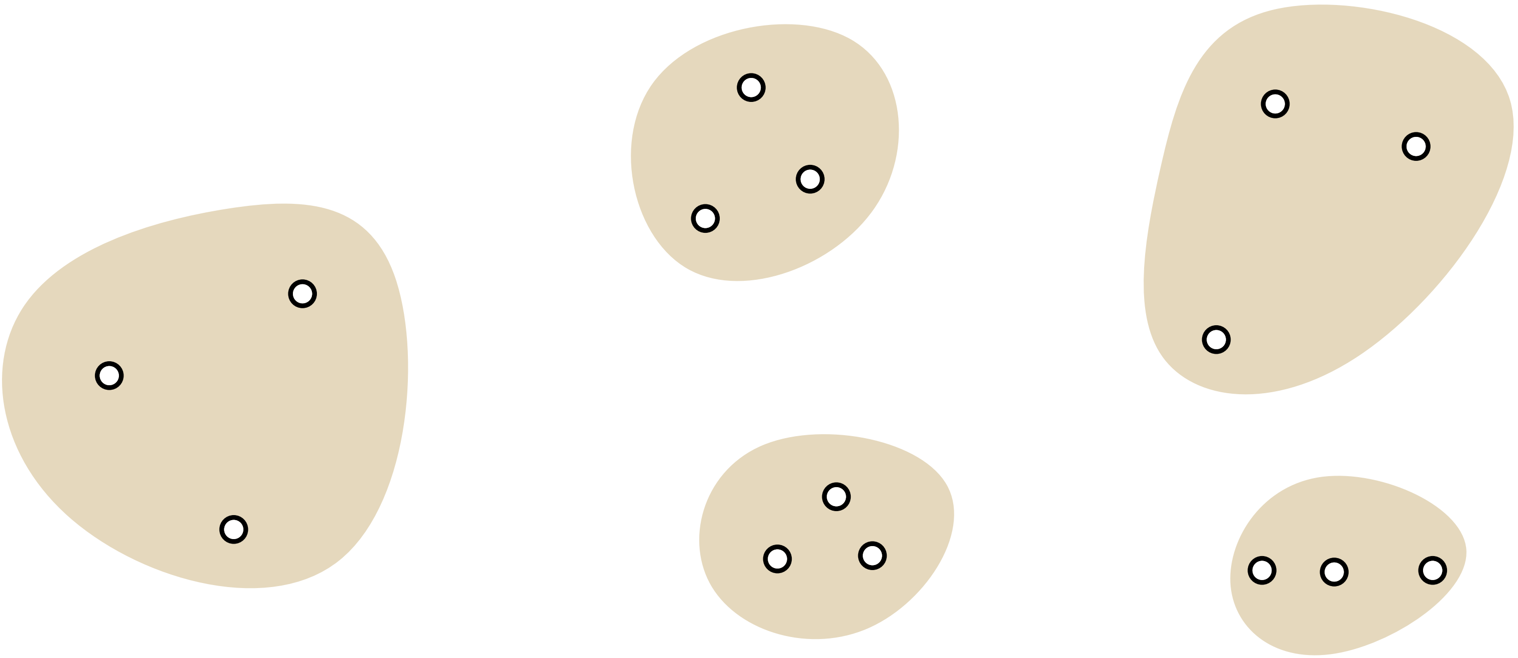


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition. . .

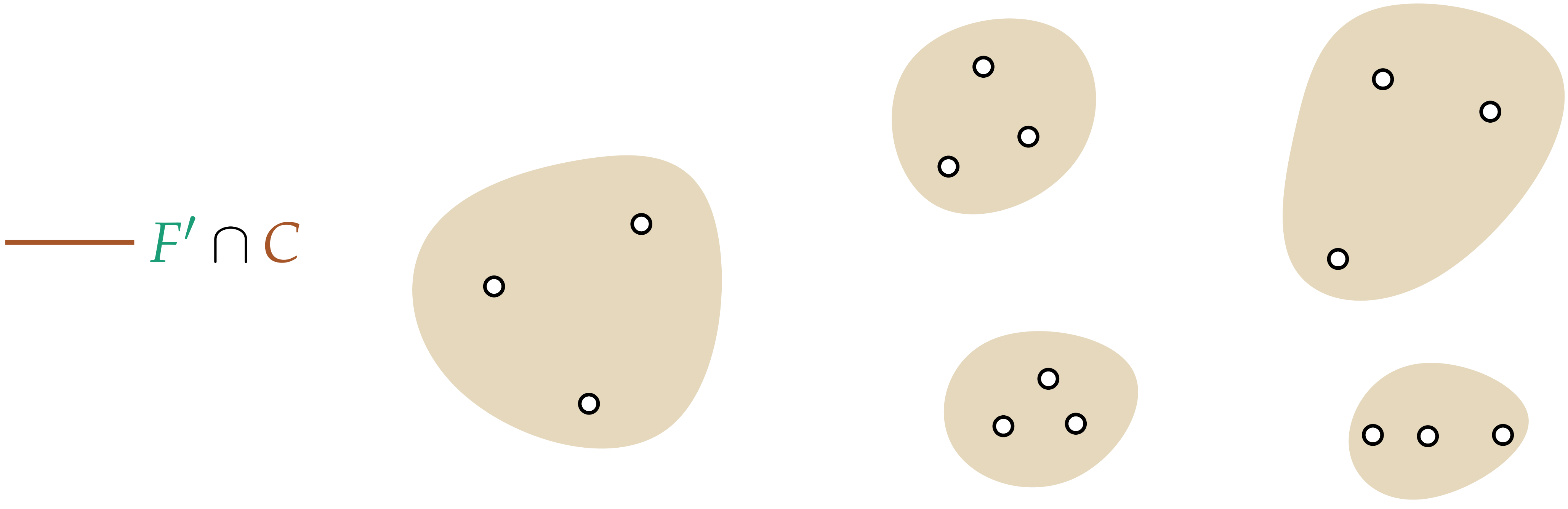


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition…

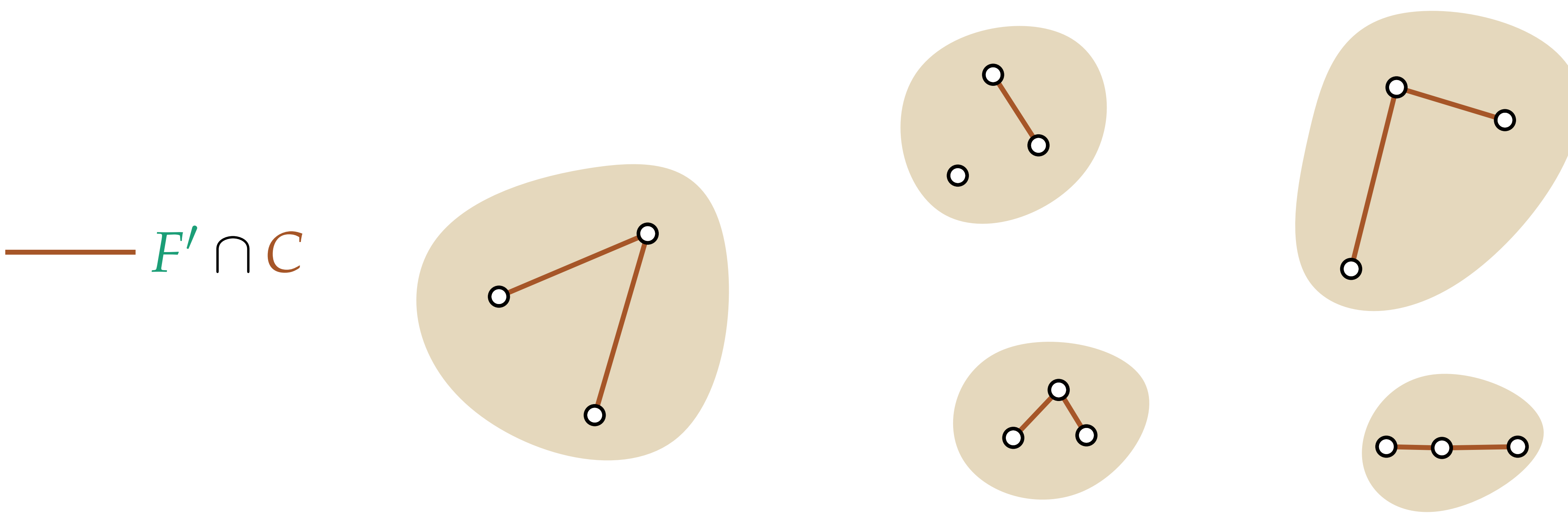


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition…

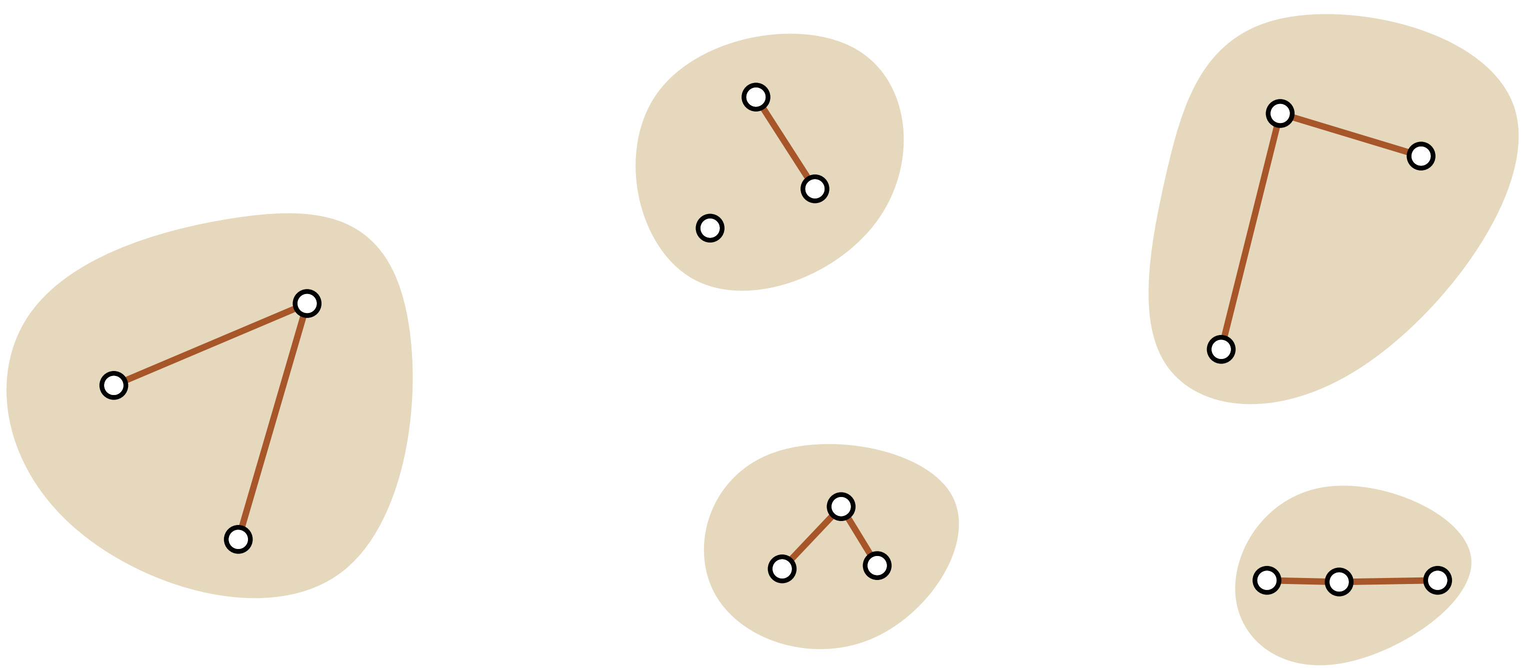


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition. . .



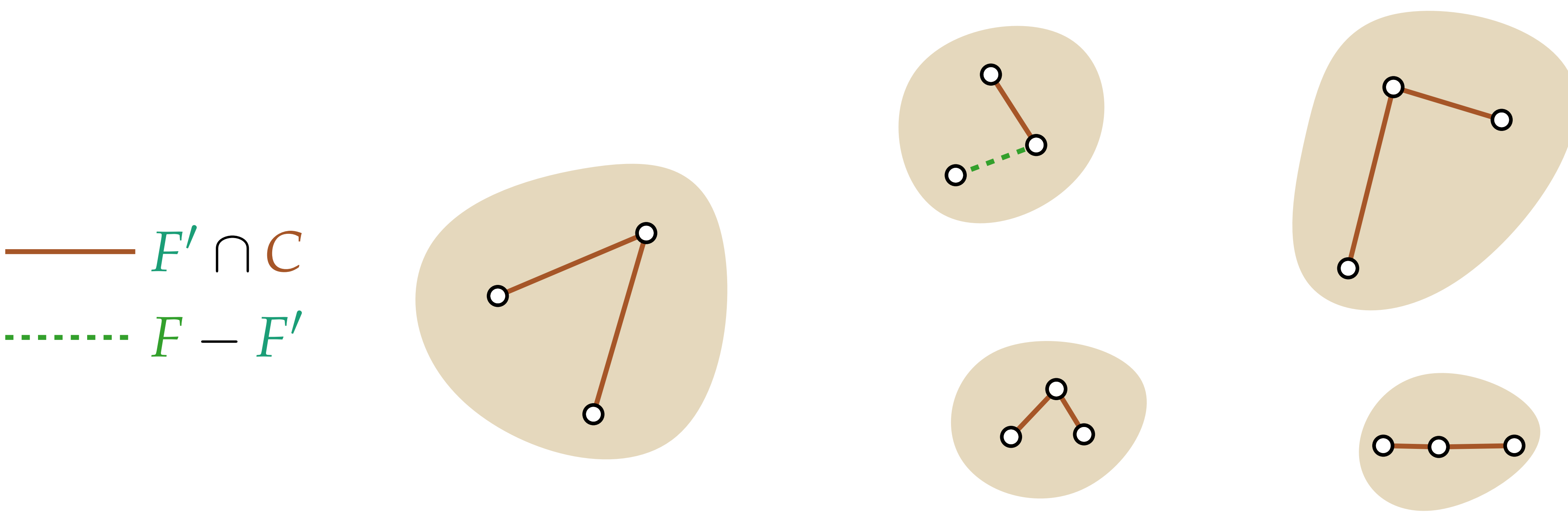
$F' \cap C$

$F - F'$

Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition...

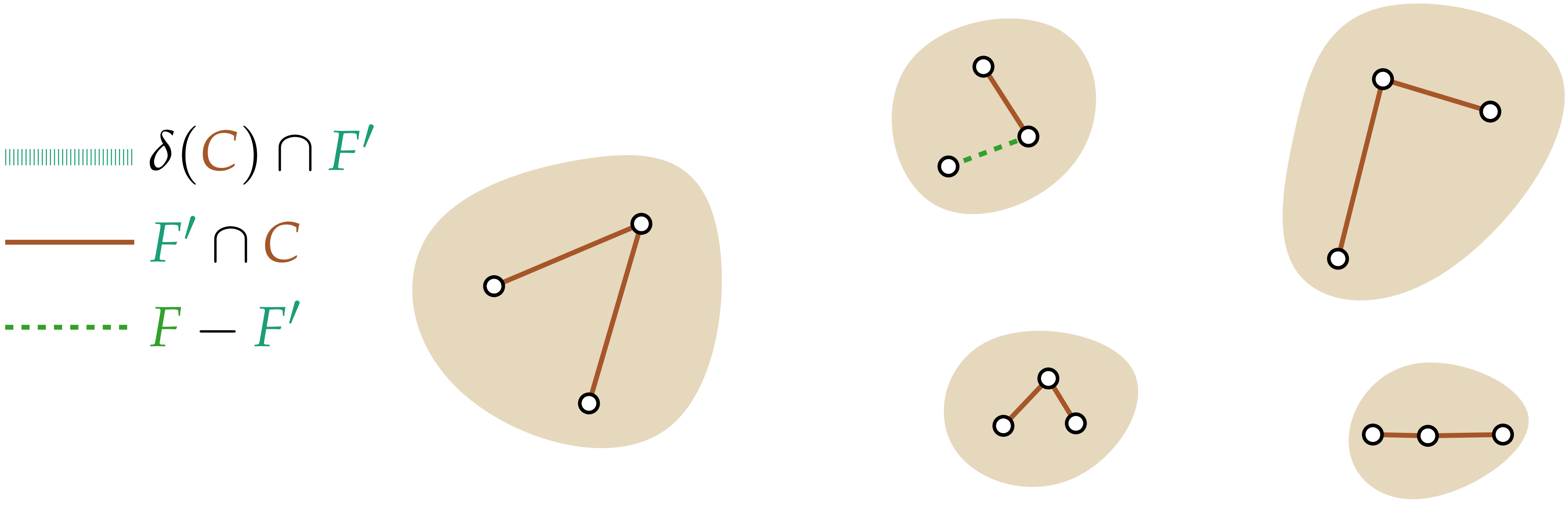


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition…

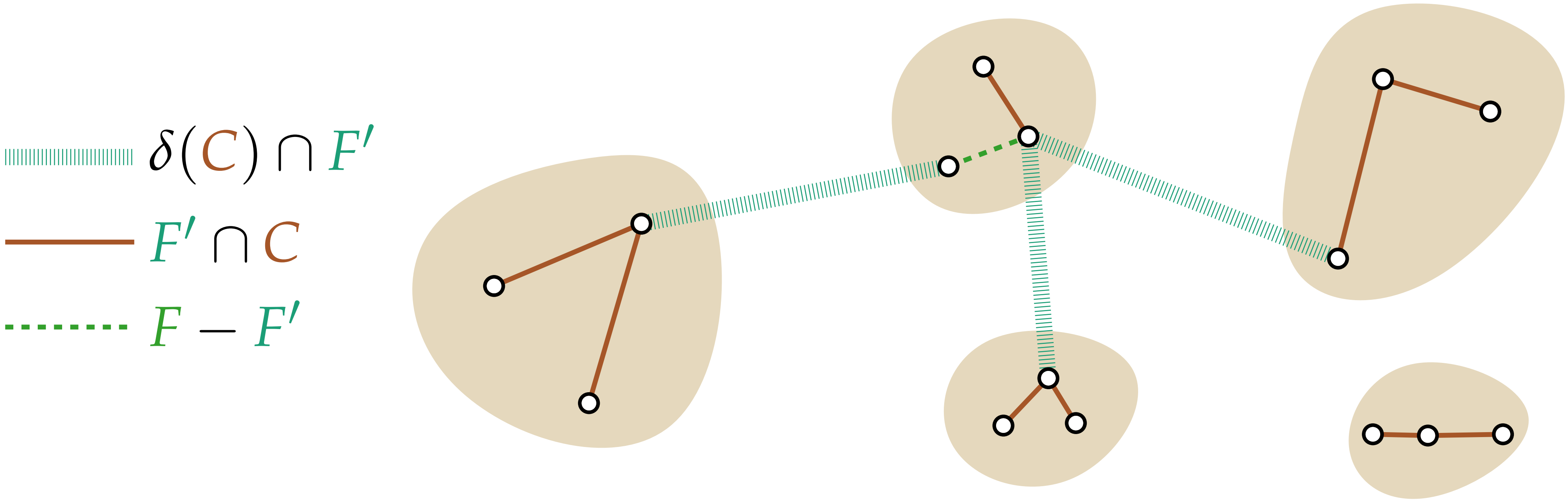


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition...

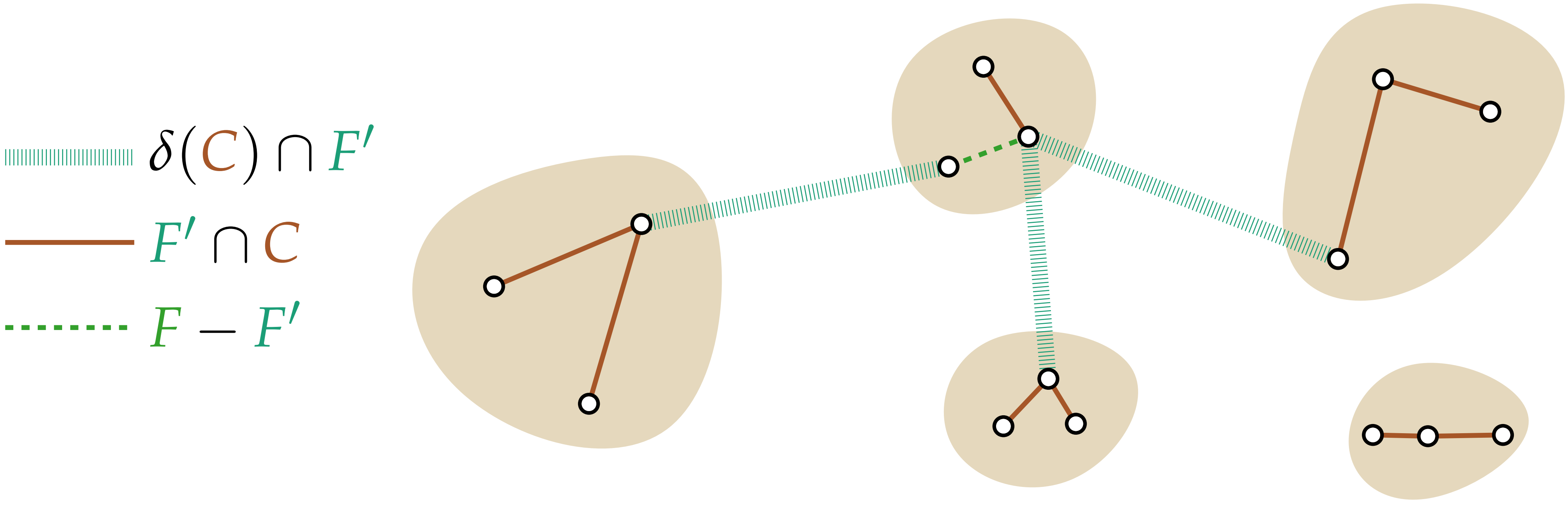


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition...
each component C of F is a forest in F'
$\rightsquigarrow$ avg. degree

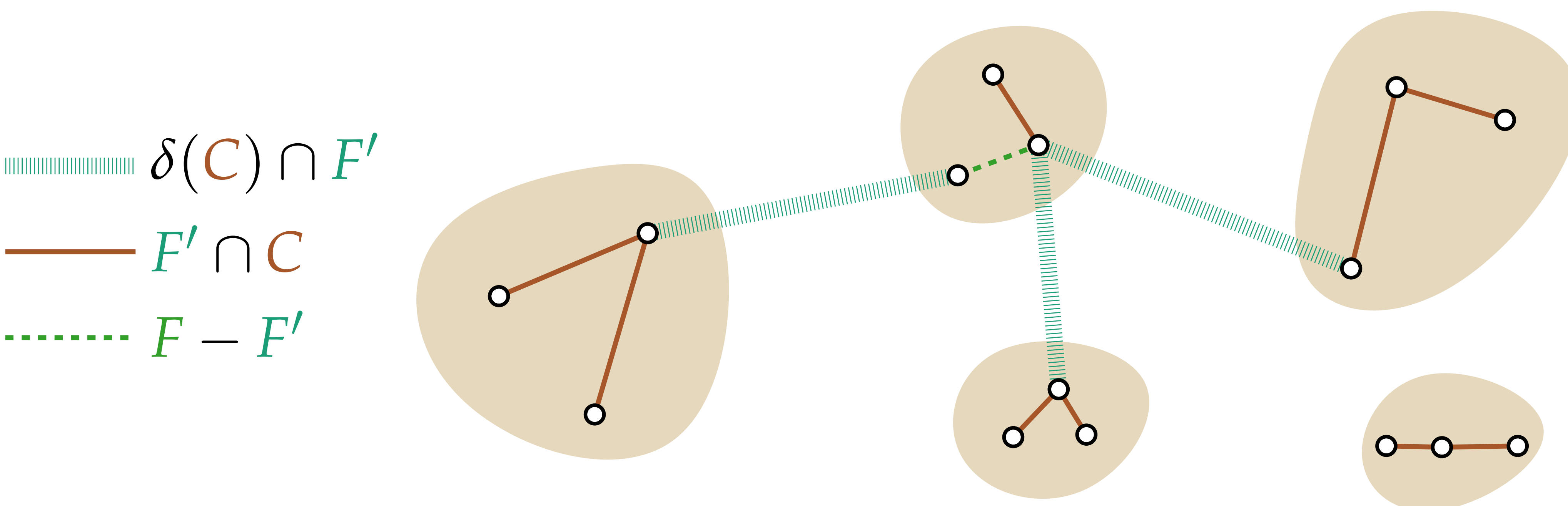


Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \le 2|\mathcal{C}|\,.$$

Proof. First the intuition…
each component C of F is a forest in F'
$\rightsquigarrow$ avg. degree ≤ 2



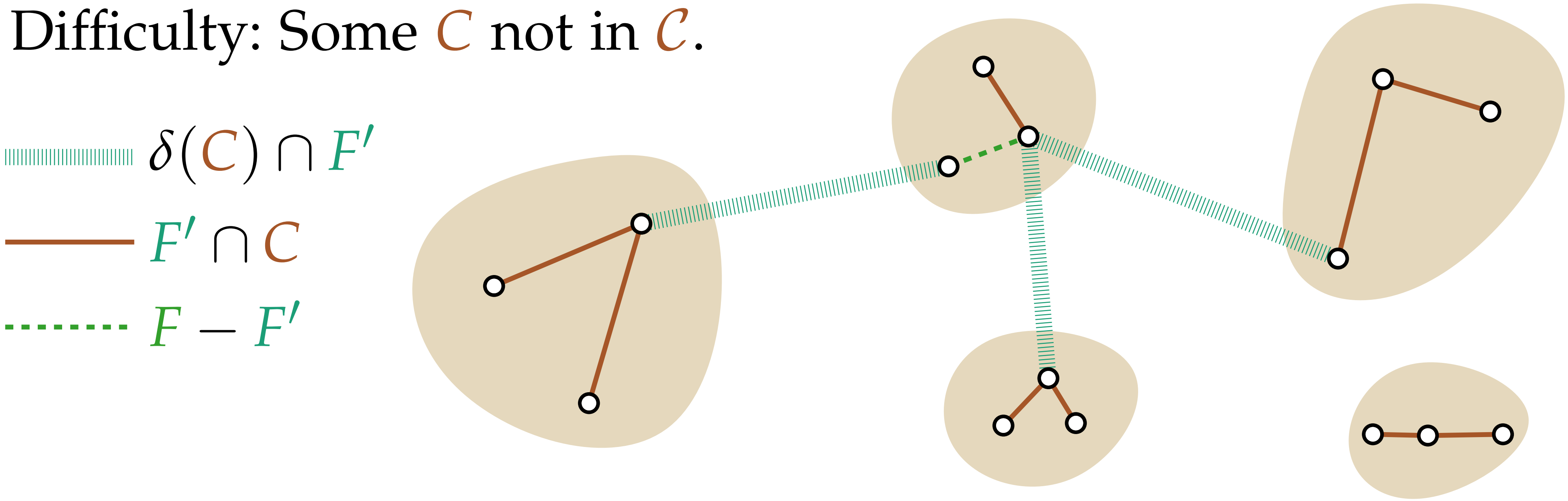
Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof. First the intuition...
each component C of F is a forest in F'
$\rightsquigarrow$ avg. degree ≤ 2

Difficulty: Some C not in $\mathcal{C}$.

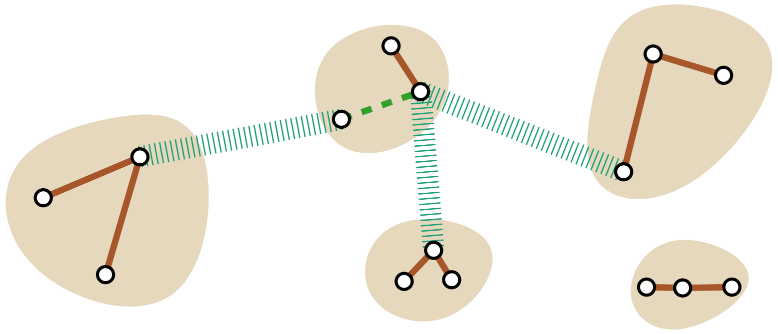


Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.



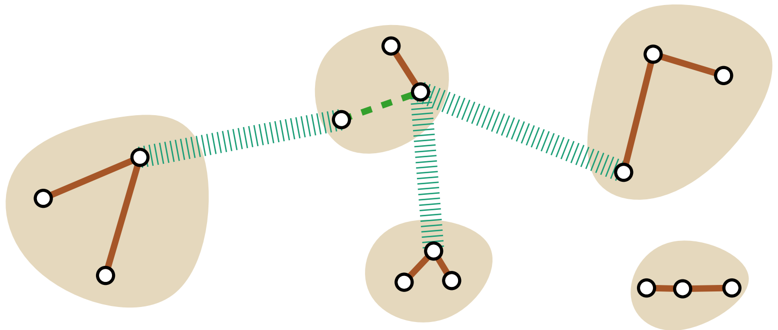
Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}| \,.$$

Proof.

Consider i-th iteration where e_i is added to F.



Proof of Structure Lemma

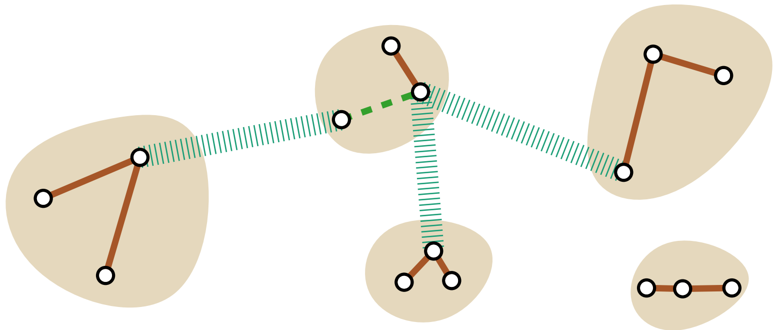
Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$



Proof of Structure Lemma

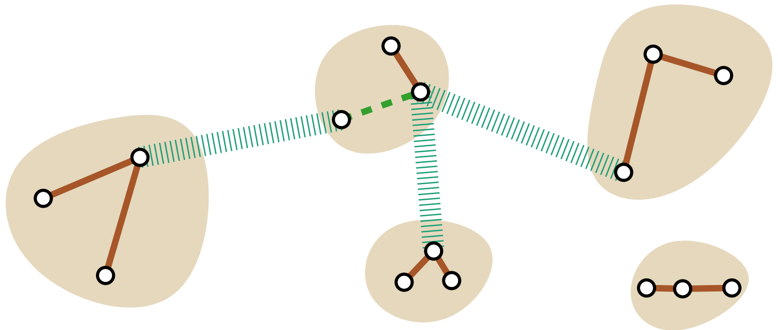
Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$



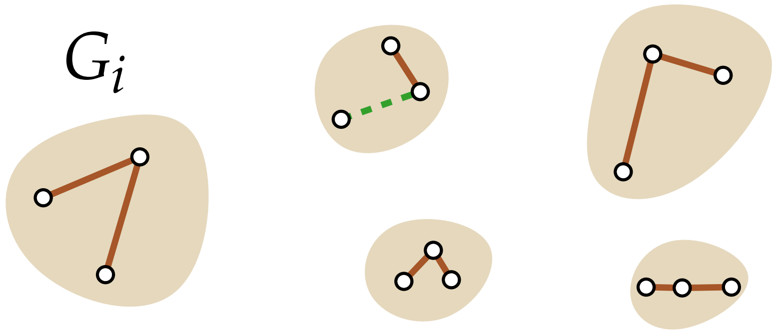
Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$



Proof of Structure Lemma

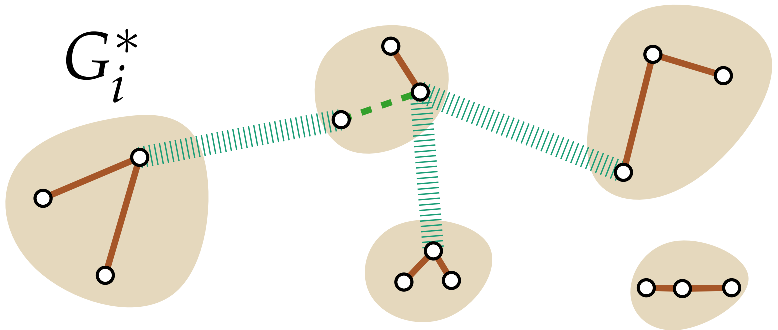
Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.



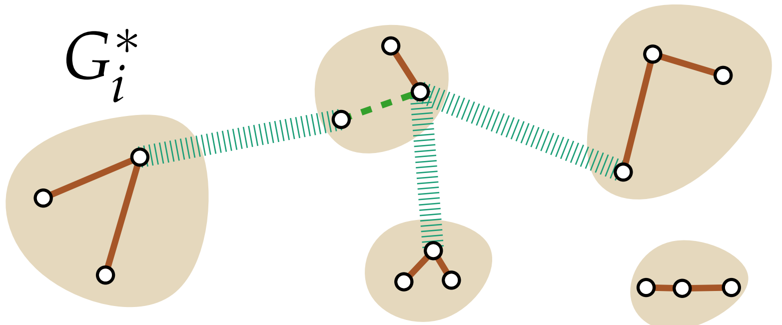
Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

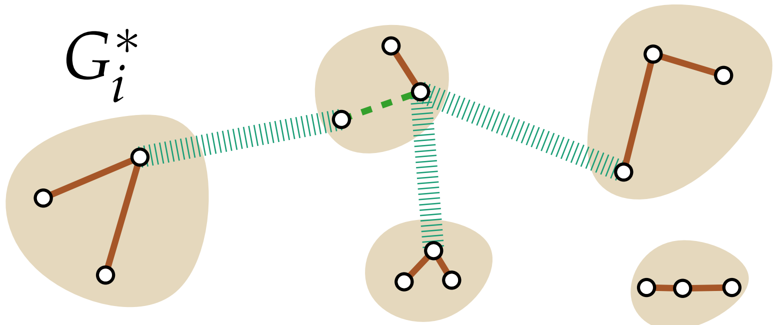
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

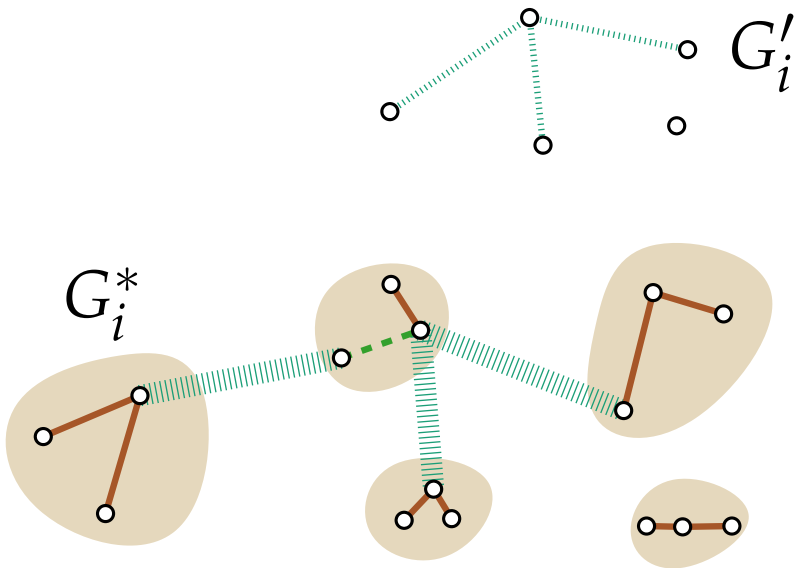
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

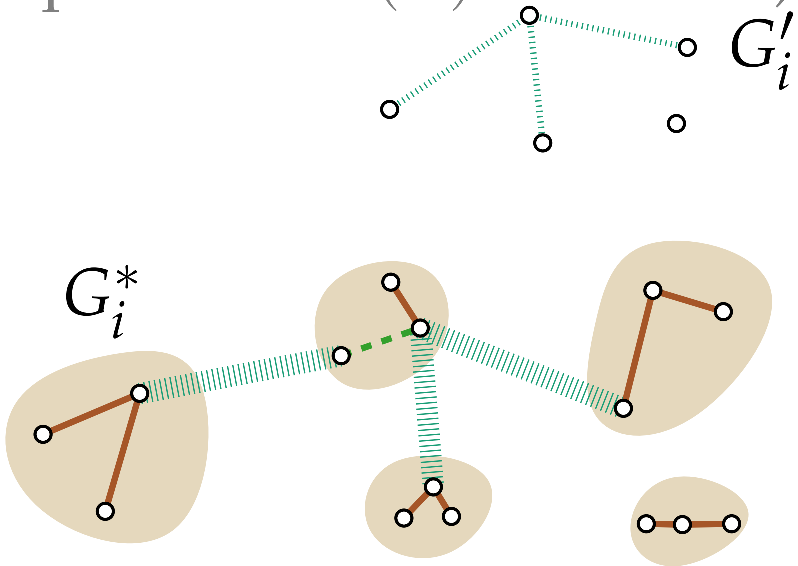
Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \dots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

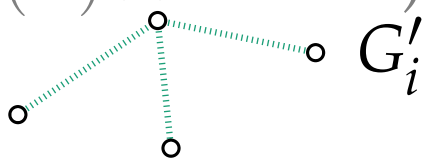
Proof.

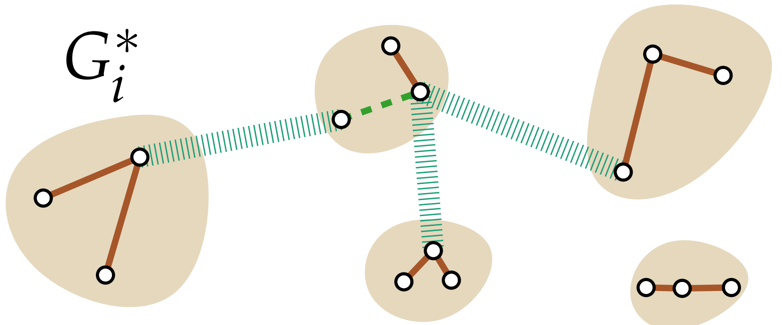
Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)





Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

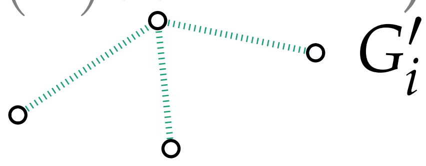
Consider i-th iteration where e_i is added to F.

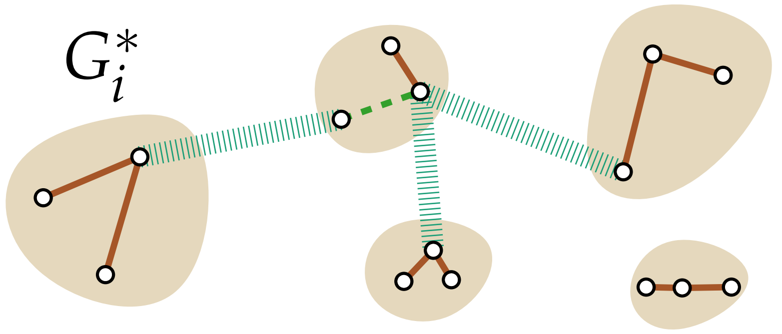
Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.





Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

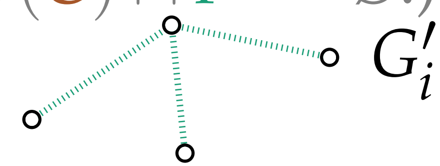
Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

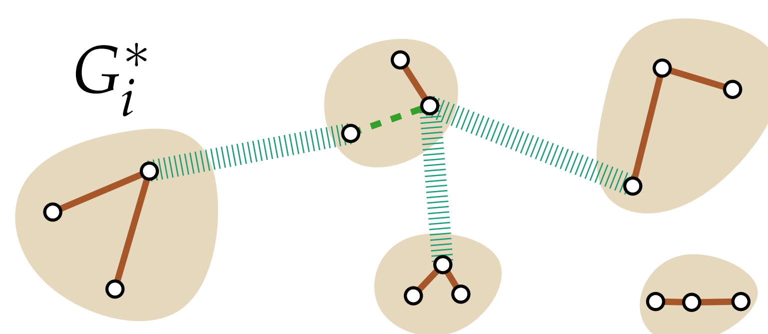
Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$





Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

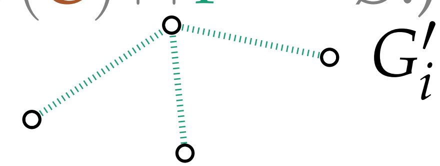
Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

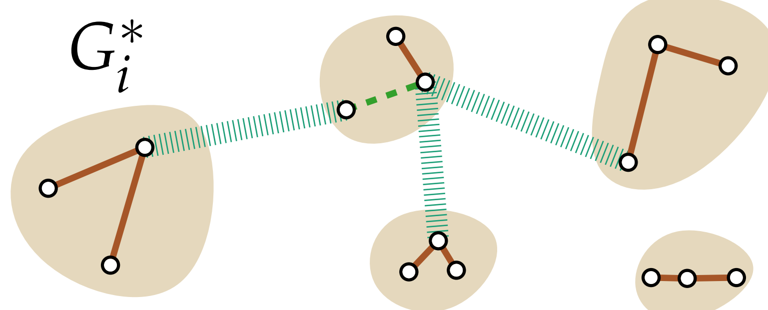
Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.



Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$= 2|E(G')|$



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \le 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

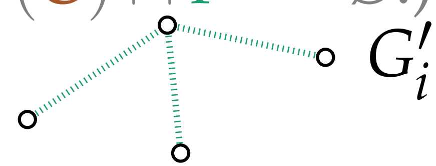
Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

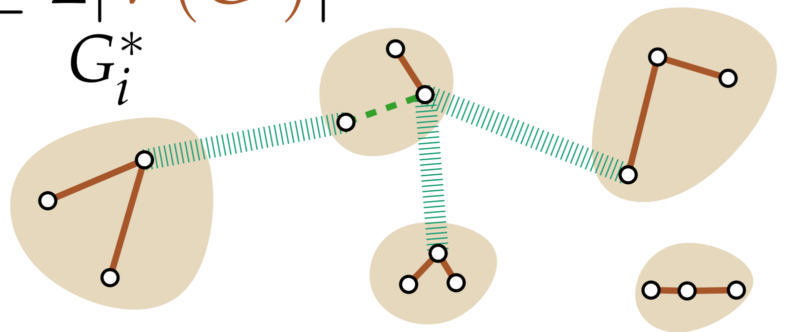
(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$= 2|E(G')| \le 2|V(G')|$





Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

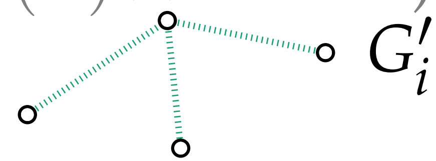
Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

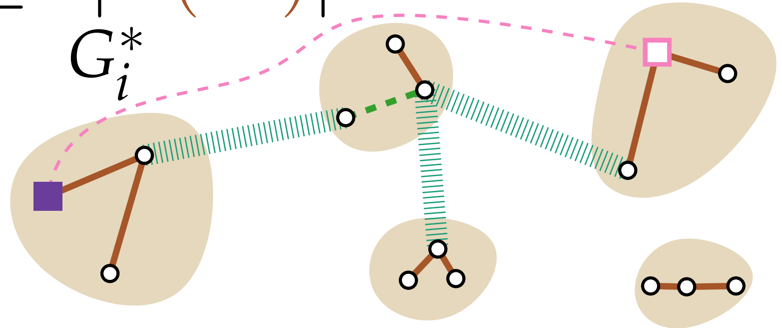
(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$= 2|E(G')| \leq 2|V(G')|$





Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C\in\mathcal{C}} |\delta(C)\cap F'| \le 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1,\ldots,e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

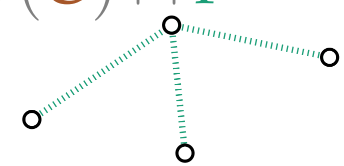
Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

Ignore all comp. C with $\delta(C)\cap F' = \emptyset$.)

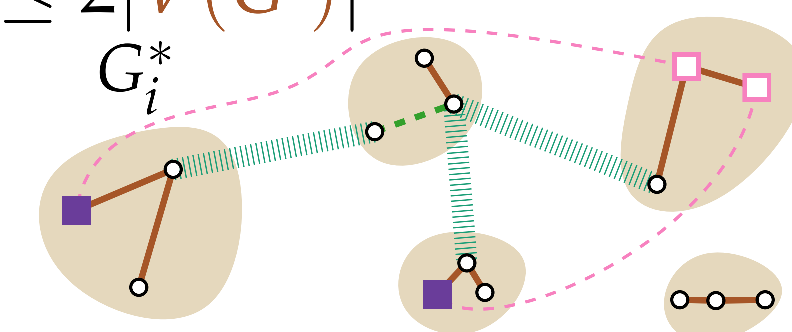
Claim. G_i' is a forest.

Note: $\sum_{C \text{ comp.}} |\delta(C)\cap F'| = \sum_{v\in V(G_i')} |\deg_{G'}(v)|$

$$= 2|E(G')| \le 2|V(G')|$$



G_i'



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

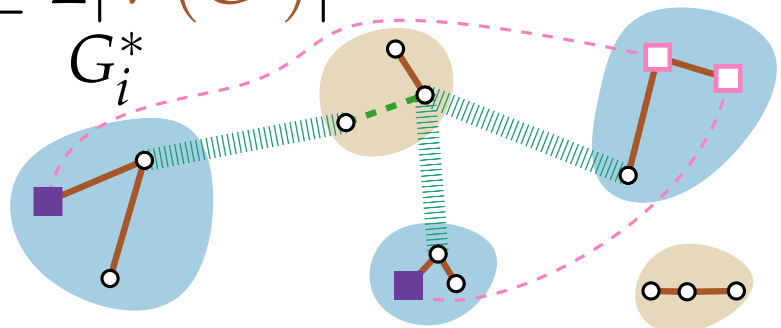
Let $F_i = \{e_1, \dots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$
$$= 2|E(G')| \leq 2|V(G')|$$



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \dots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

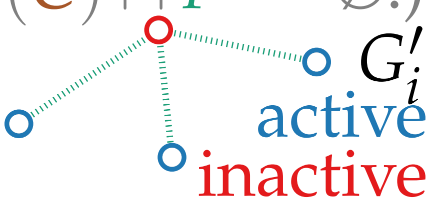
Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

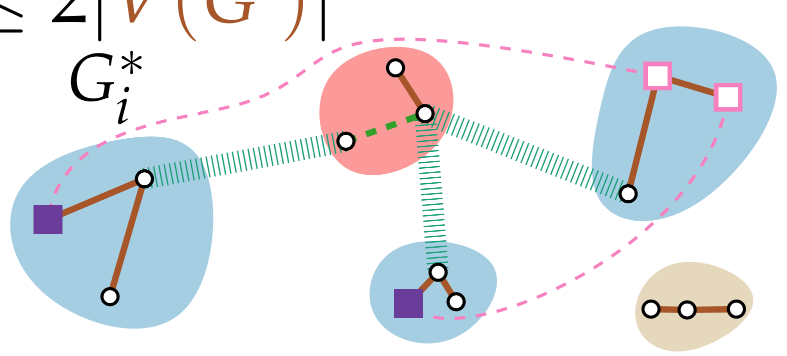
(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$= 2|E(G')| \leq 2|V(G')|$





Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

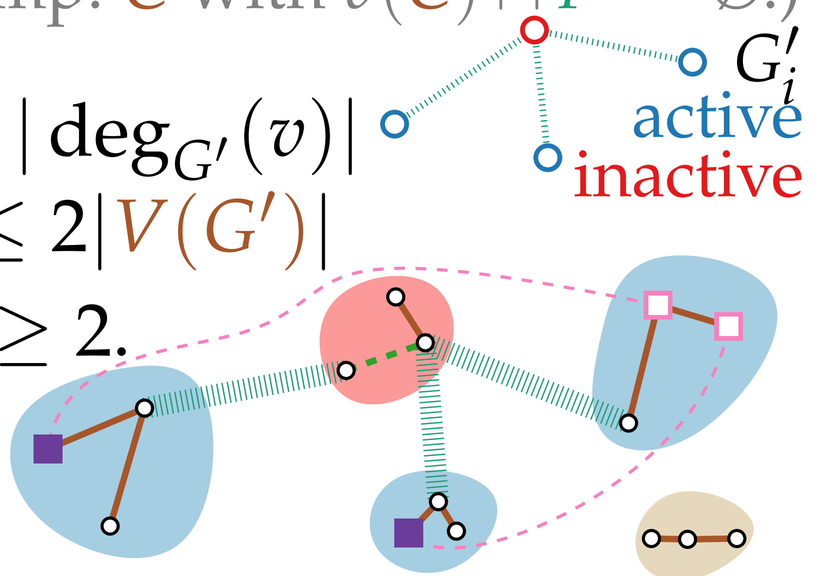
Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$$= 2|E(G')| \leq 2|V(G')|$$

Claim. Inactive vertices have degree ≥ 2.



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

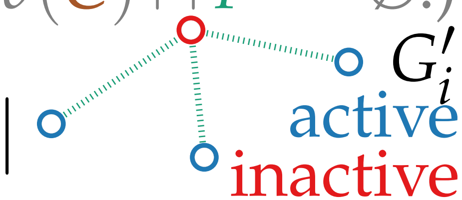
Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

(Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

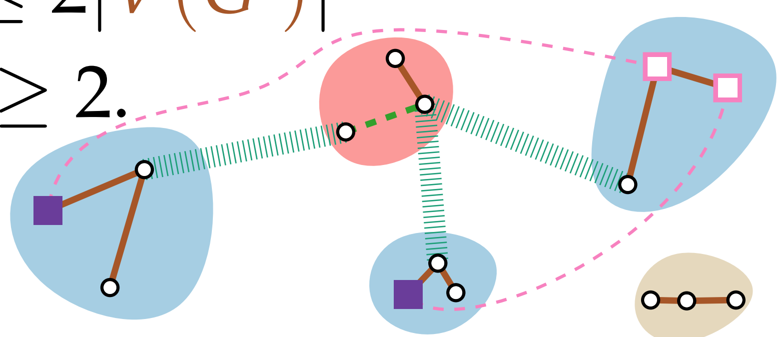


Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$$= 2|E(G')| \leq 2|V(G')|$$

Claim. Inactive vertices have degree ≥ 2.

Then $\sum_{v \text{ active}} |\deg_{G'}(v)| \leq$



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

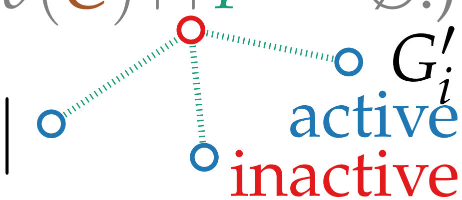
Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.



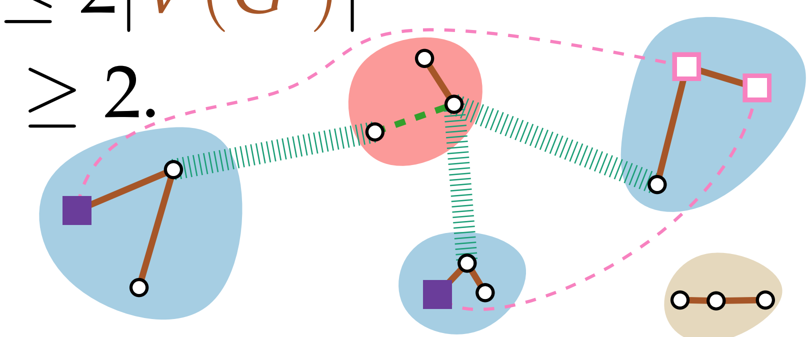
Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$= 2|E(G')| \leq 2|V(G')|$

Claim. Inactive vertices have degree ≥ 2.

Then $\sum_{v \text{ active}} |\deg_{G'}(v)| \leq$

$2 \cdot |V(G')| - 2 \cdot \#(\text{inactive})$



Proof of Structure Lemma

Lemma. For each $\mathcal{C}$ of an iteration of the algorithm:

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Proof.

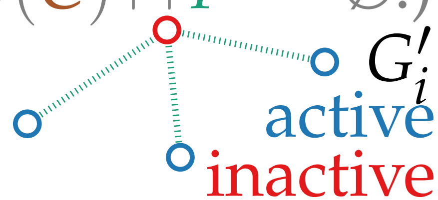
Consider i-th iteration where e_i is added to F.

Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

Contract each comp. C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

Ignore all comp. C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.



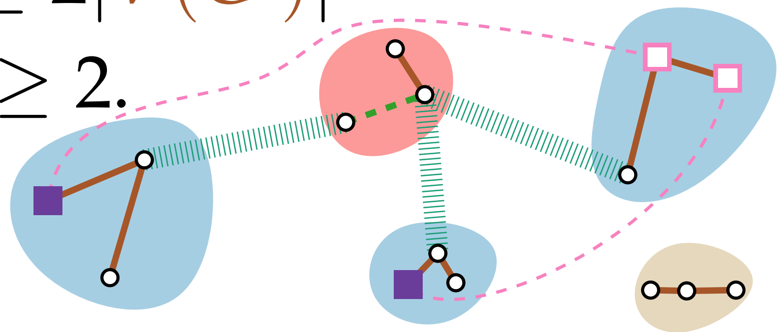
Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} |\deg_{G'}(v)|$

$= 2|E(G')| \leq 2|V(G')|$

Claim. Inactive vertices have degree ≥ 2.

Then $\sum_{v \text{ active}} |\deg_{G'}(v)| \leq$

$2 \cdot |V(G')| - 2 \cdot \#(\text{inactive}) = 2|\mathcal{C}|$. $\square$



Approximation Algorithms

Lecture 12:
SteinerForest via Primal-Dual

Part VI:
Analysis

Philipp Kindermann

Summer Semester 2020

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

As before

$$\sum_{e \in F'} c_e \stackrel{\text{CS}}{=} \sum_{e \in F'} \sum_{S \colon e \in \delta(S)} y_S = \sum_{S} |\delta(S) \cap F'| \cdot y_S.$$

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

As before

$$\sum_{e \in F'} c_e \overset{\text{CS}}{=} \sum_{e \in F'} \sum_{S \colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F'| \cdot y_S.$$

We prove by induction over the number of iterations of the algorithm that

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

As before

$$\sum_{e \in F'} c_e \overset{\text{CS}}{=} \sum_{e \in F'} \sum_{S\colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F'| \cdot y_S.$$

We prove by induction over the number of iterations of the algorithm that

$$\sum_S |\delta(S) \cap F'| \cdot y_S \leq$$

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

As before

$$\sum_{e \in F'} c_e \stackrel{\text{CS}}{=} \sum_{e \in F'} \sum_{S \colon e \in \delta(S)} y_S = \sum_{S} |\delta(S) \cap F'| \cdot y_S.$$

We prove by induction over the number of iterations of the algorithm that

$$\sum_{S} |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_{S} y_S. \qquad (*)$$

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

As before

$$\sum_{e \in F'} c_e \overset{\text{CS}}{=} \sum_{e \in F'} \sum_{S \colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F'| \cdot y_S.$$

We prove by induction over the number of iterations of the algorithm that

$$\sum_S |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_S y_S. \qquad (*)$$

From that, the claim of the theorem follows.

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_{S} |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_{S} y_S. \qquad (*)$$

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_{S} |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_{S} y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_S |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_S y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Assume that $(*)$ holds at the start of each iteration.

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_S |\delta(S) \cap F'| \cdot y_S \le 2 \sum_S y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Assume that $(*)$ holds at the start of each iteration.

In the active iteration, we increase y_C for all $C \in \mathcal{C}$ by the same amount, say $\varepsilon \ge 0$.

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_S |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_S y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Assume that $(*)$ holds at the start of each iteration.

In the active iteration, we increase y_C for all $C \in \mathcal{C}$ by the same amount, say $\varepsilon \geq 0$.

This increases the left side of $(*)$ by

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_{S} |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_{S} y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Assume that $(*)$ holds at the start of each iteration.

In the active iteration, we increase y_C for all $C \in \mathcal{C}$ by the same amount, say $\varepsilon \geq 0$.

This increases the left side of $(*)$ by $\varepsilon \sum_{C \in \mathcal{C}} |\delta(C) \cap F'|$

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_S |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_S y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Assume that $(*)$ holds at the start of each iteration.

In the active iteration, we increase y_C for all $C \in \mathcal{C}$ by the same amount, say $\varepsilon \geq 0$.

This increases the left side of $(*)$ by $\varepsilon \sum_{C \in \mathcal{C}} |\delta(C) \cap F'|$ and the right side by

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_{S} |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_{S} y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Assume that $(*)$ holds at the start of each iteration.

In the active iteration, we increase y_C for all $C \in \mathcal{C}$ by the same amount, say $\varepsilon \geq 0$.

This increases the left side of $(*)$ by $\varepsilon \sum_{C \in \mathcal{C}} |\delta(C) \cap F'|$ and the right side by $2\varepsilon|\mathcal{C}|$.

Analysis

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Proof.

$$\sum_{S} |\delta(S) \cap F'| \cdot y_S \leq 2 \sum_{S} y_S. \qquad (*)$$

Base case trivial since we start with $y_S = 0$ for each S.

Assume that $(*)$ holds at the start of each iteration.

In the active iteration, we increase y_C for all $C \in \mathcal{C}$ by the same amount, say $\varepsilon \geq 0$.

This increases the left side of $(*)$ by $\varepsilon \sum_{C \in \mathcal{C}} |\delta(C) \cap F'|$ and the right side by $2\varepsilon|\mathcal{C}|$.

Thus, by the Structure Lemma, $(*)$ also holds after the active iteration. $\square$

Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Summary

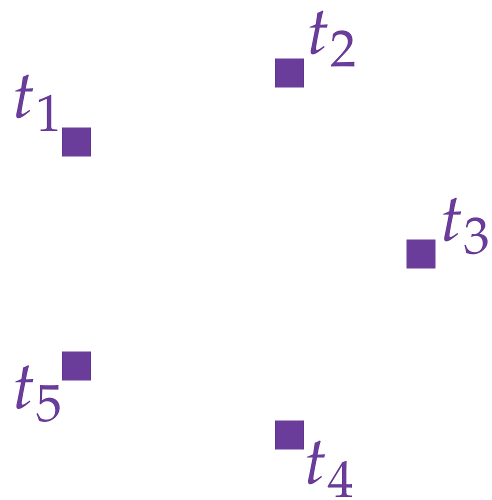
Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Analysis tight?

Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

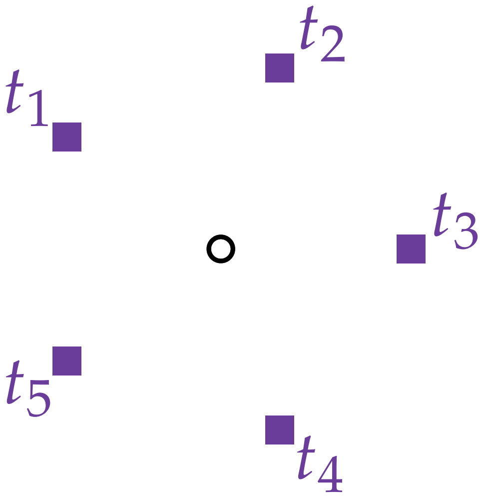
Analysis tight?



Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

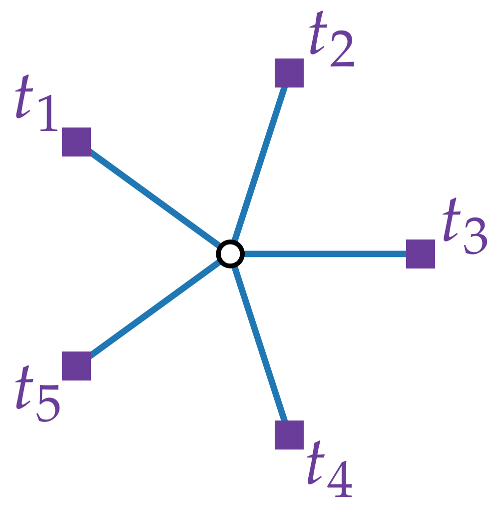
Analysis tight?



Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

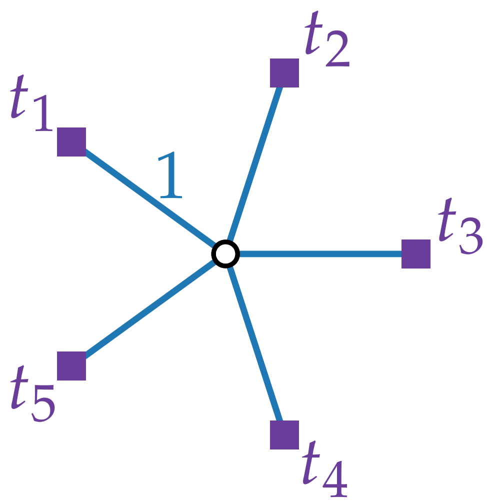
Analysis tight?



Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

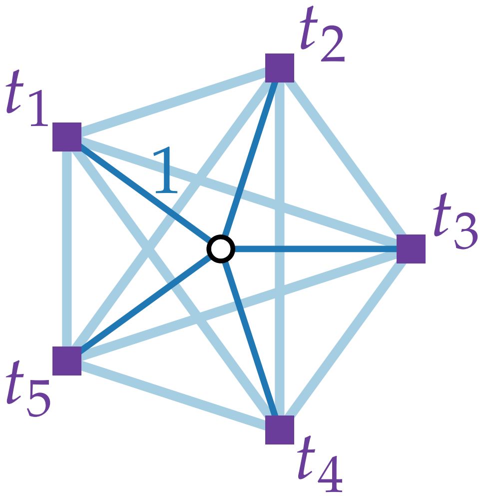
Analysis tight?



Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

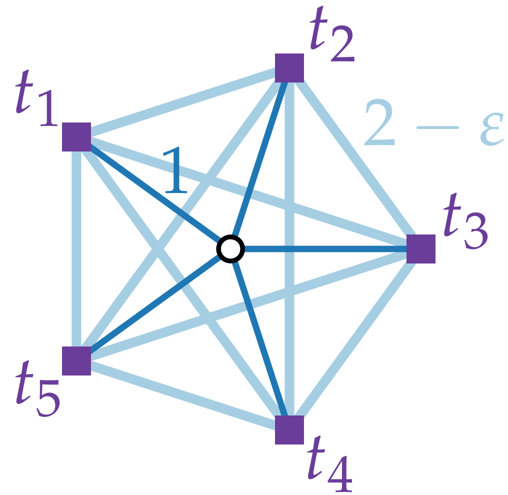
Analysis tight?



Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

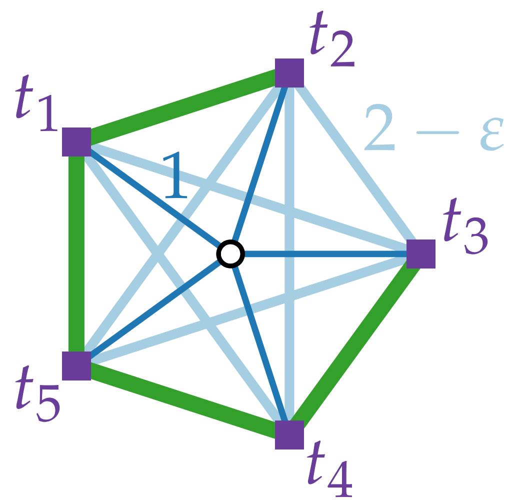
Analysis tight?



Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Analysis tight?

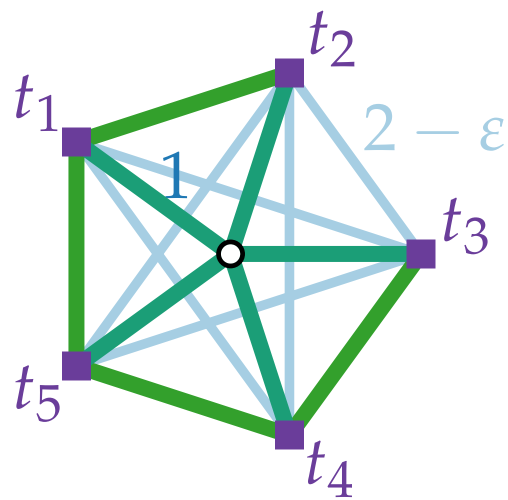


ALG $= (2-\varepsilon)(n-1)$

Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Analysis tight?

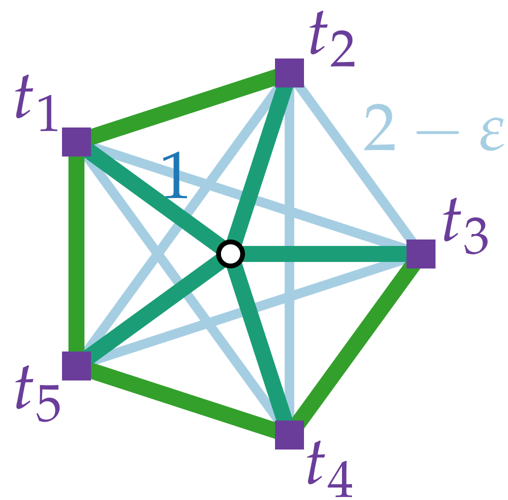


ALG $= (2-\varepsilon)(n-1)$
OPT $= n$

Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Analysis tight?



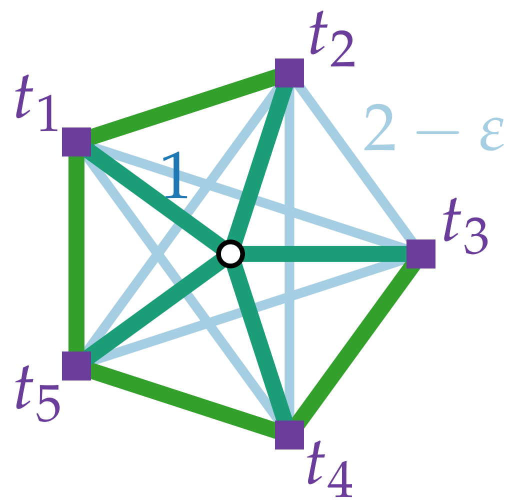
ALG $= (2-\varepsilon)(n-1)$
OPT $= n$

better?

Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Analysis tight?



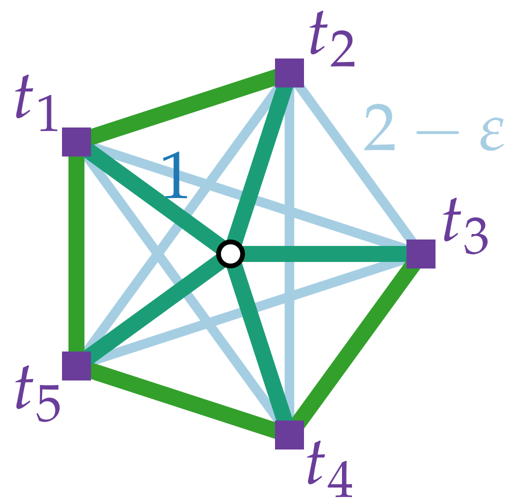
ALG $= (2-\varepsilon)(n-1)$
OPT $= n$

better?
No better approximation factor is known.

Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Analysis tight?



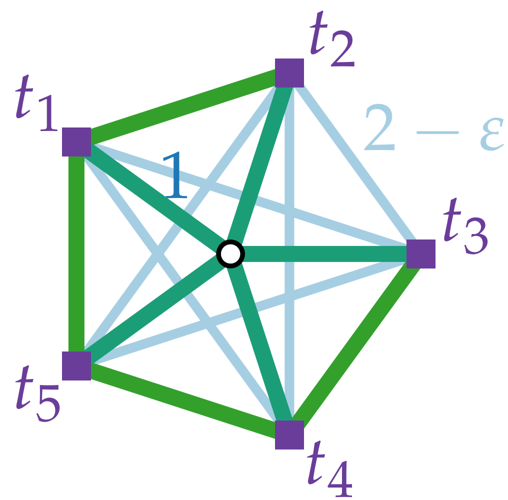
ALG $= (2-\varepsilon)(n-1)$
OPT $= n$

better?
No better approximation factor is known.
The integrality gap is $2 - 1/n$.

Summary

Theorem. The Primal-Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Analysis tight?



ALG $= (2-\varepsilon)(n-1)$
OPT $= n$

better?
No better approximation factor is known.
The integrality gap is $2 - 1/n$.
SteinerForest cannot be approximated within factor 1.0074 (unless P=NP) [Thimm '03]