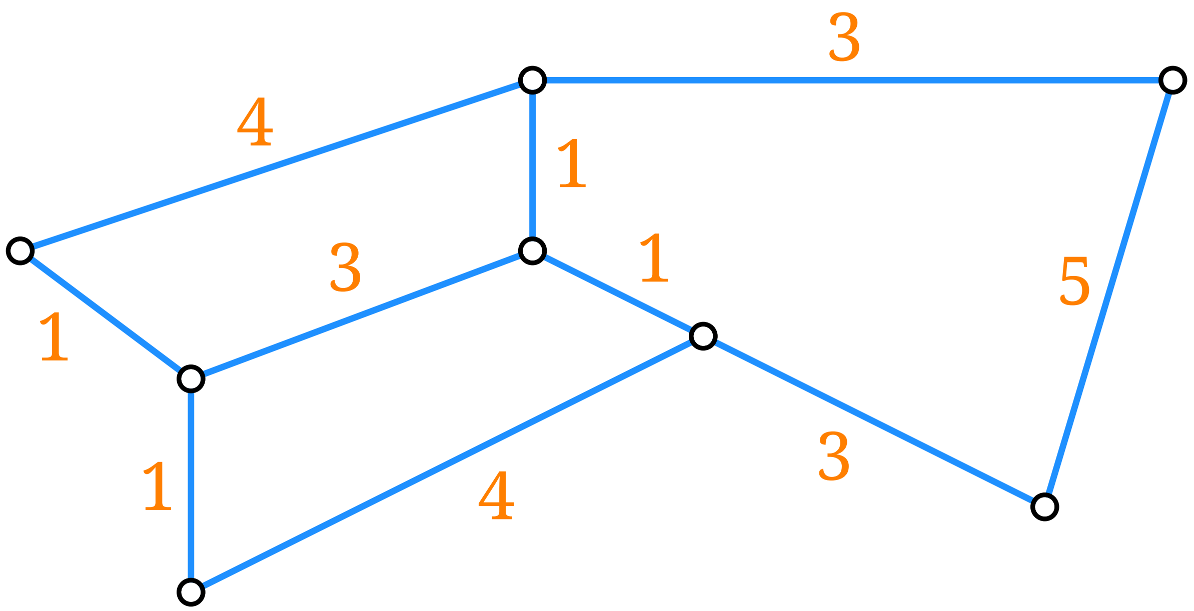
SteinerForest via Primal–Dual

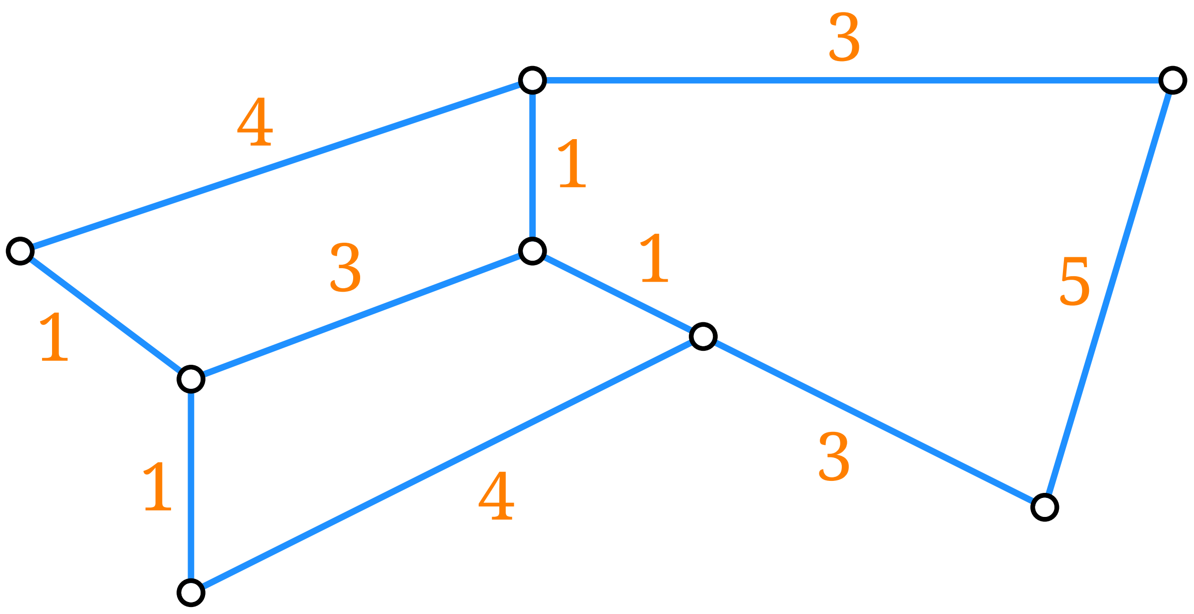
SteinerForest

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



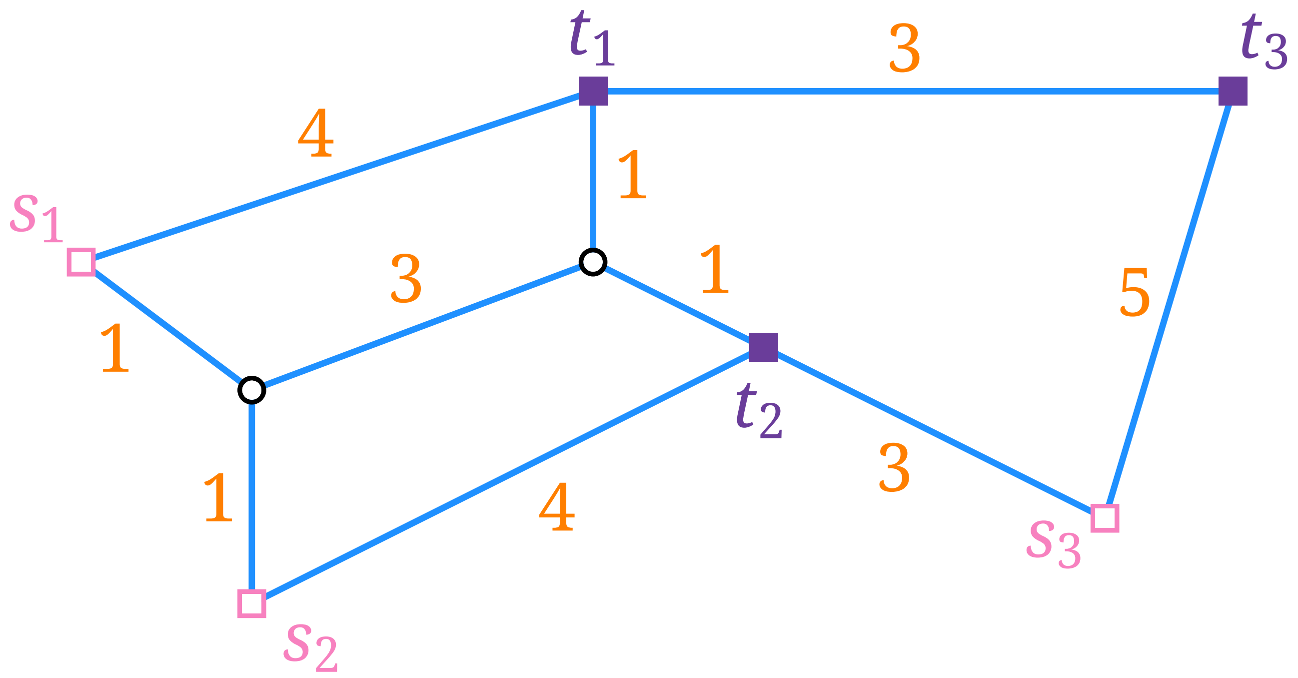
SteinerForest

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 vertex pairs.



SteinerForest

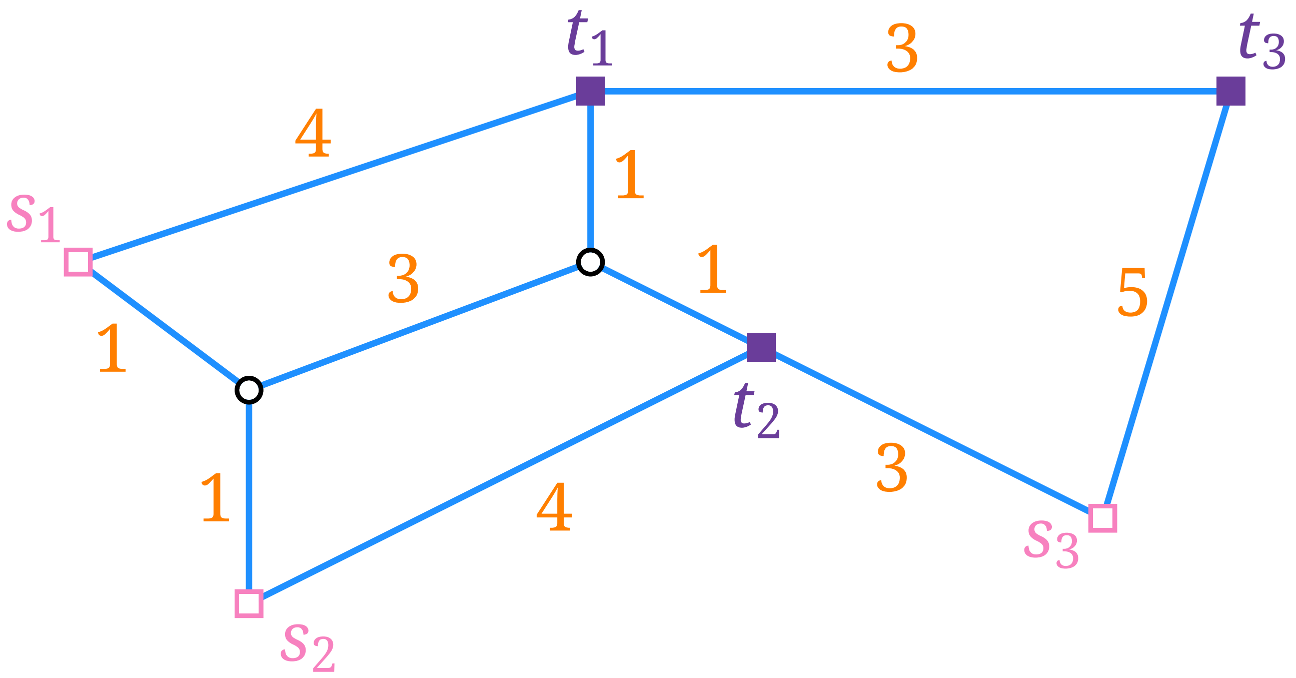
Given: A graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{N}$ and a set $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of k vertex pairs.



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 vertex pairs.

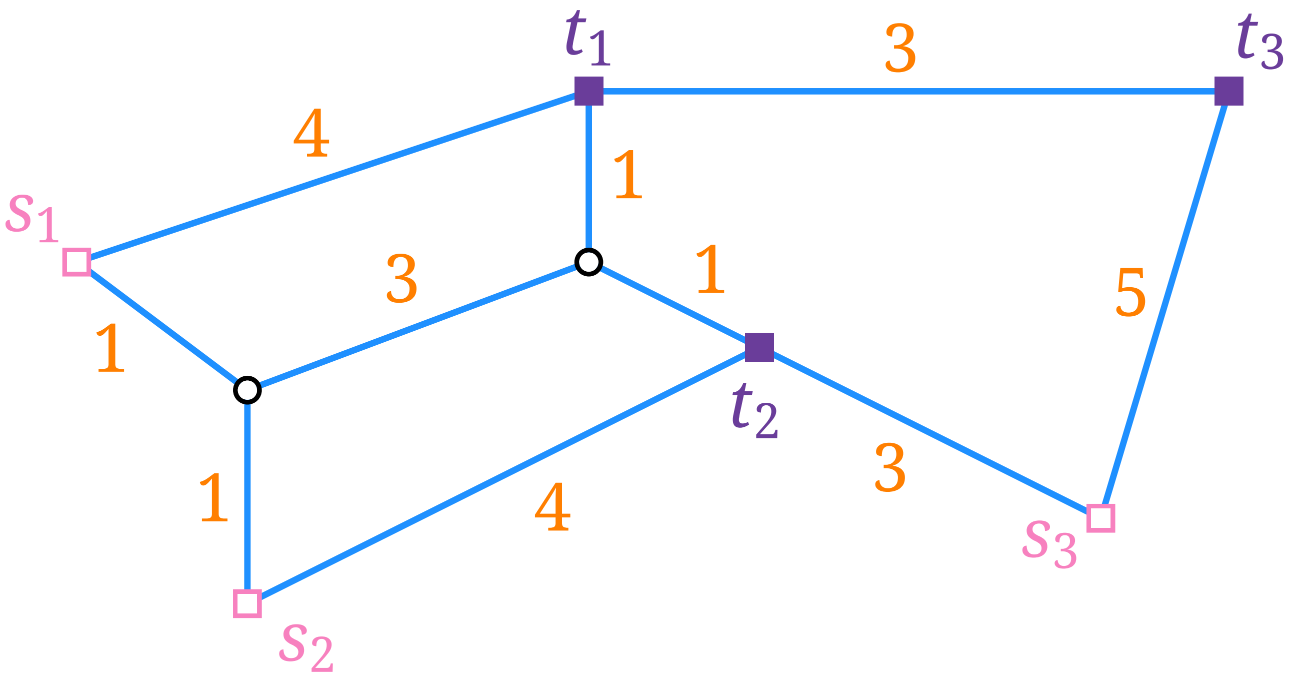
Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$



SteinerForest

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

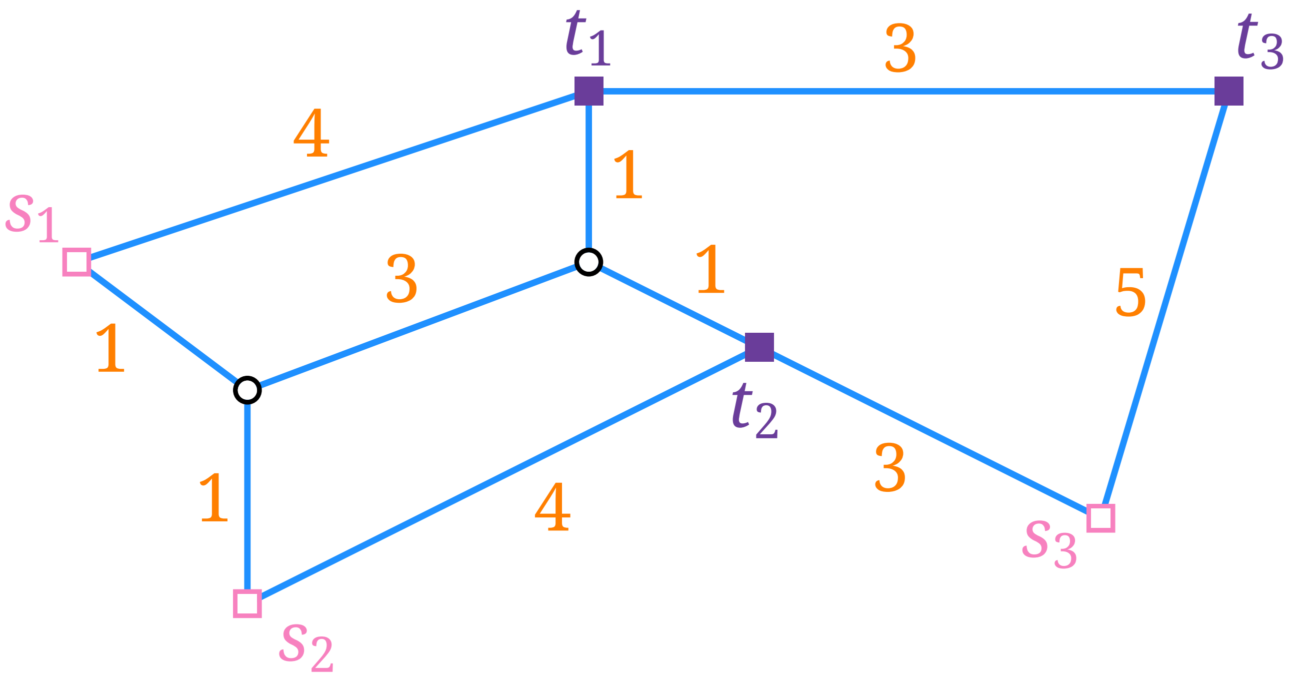
Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$ such that the subgraph (V, F) connects every pair (s_i, t_i), $i = 1, \ldots, k$.



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 vertex pairs.

Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$ such that the subgraph (V, F) connects every pair (s_i, t_i), $i = 1, \dots, k$.

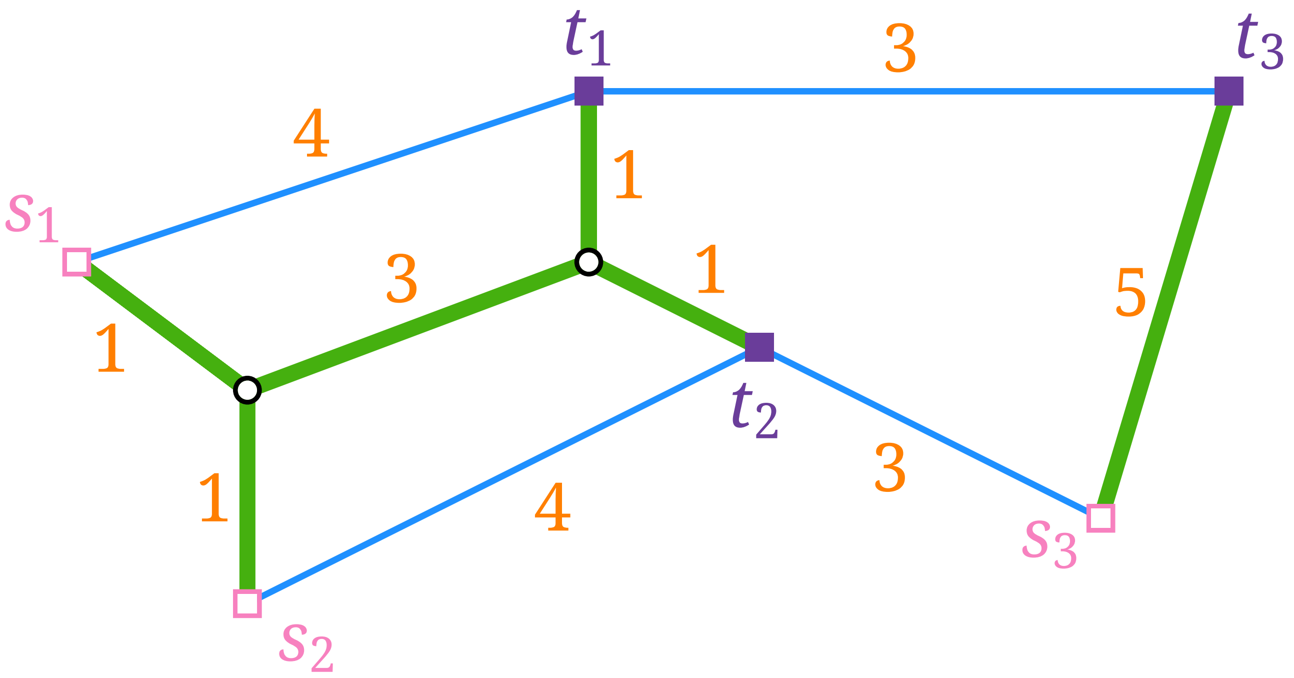


What is a minimal SteinerForest here?

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 vertex pairs.

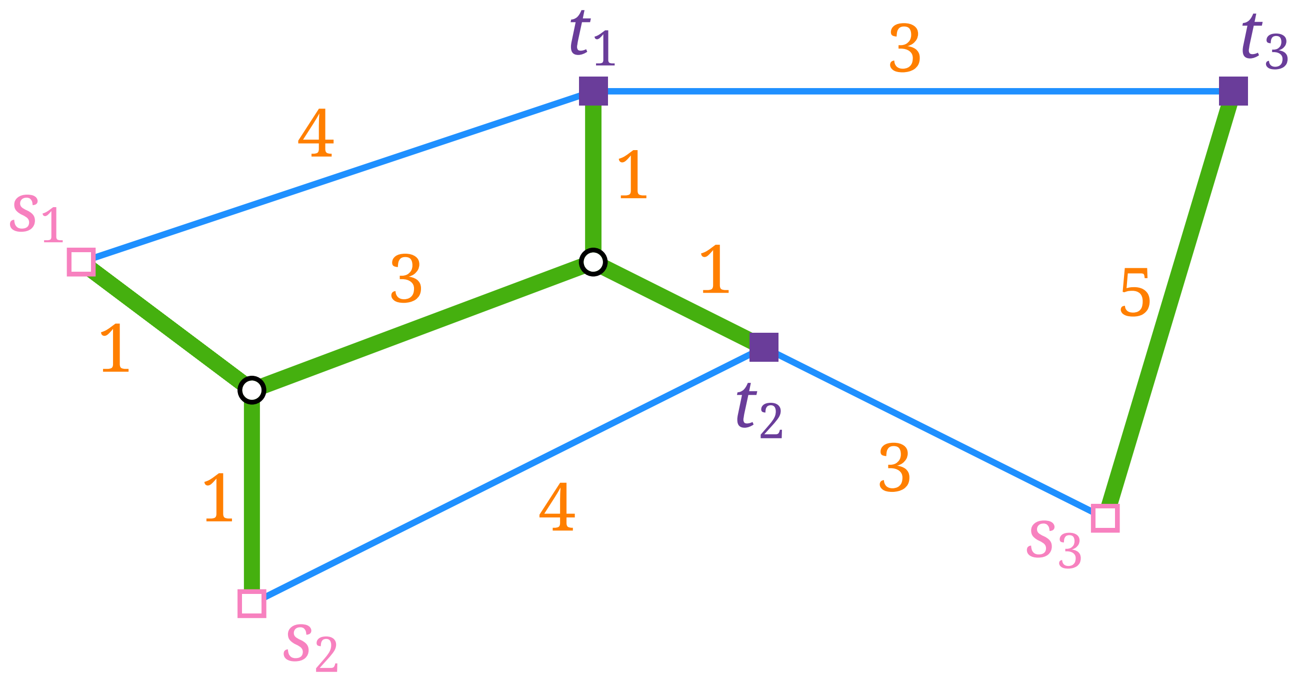
Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$ such that the subgraph (V, F) connects every pair (s_i, t_i), $i = 1, \dots, k$.



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 vertex pairs.

Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$ such that the subgraph (V, F) connects every pair (s_i, t_i), $i = 1, \ldots, k$.

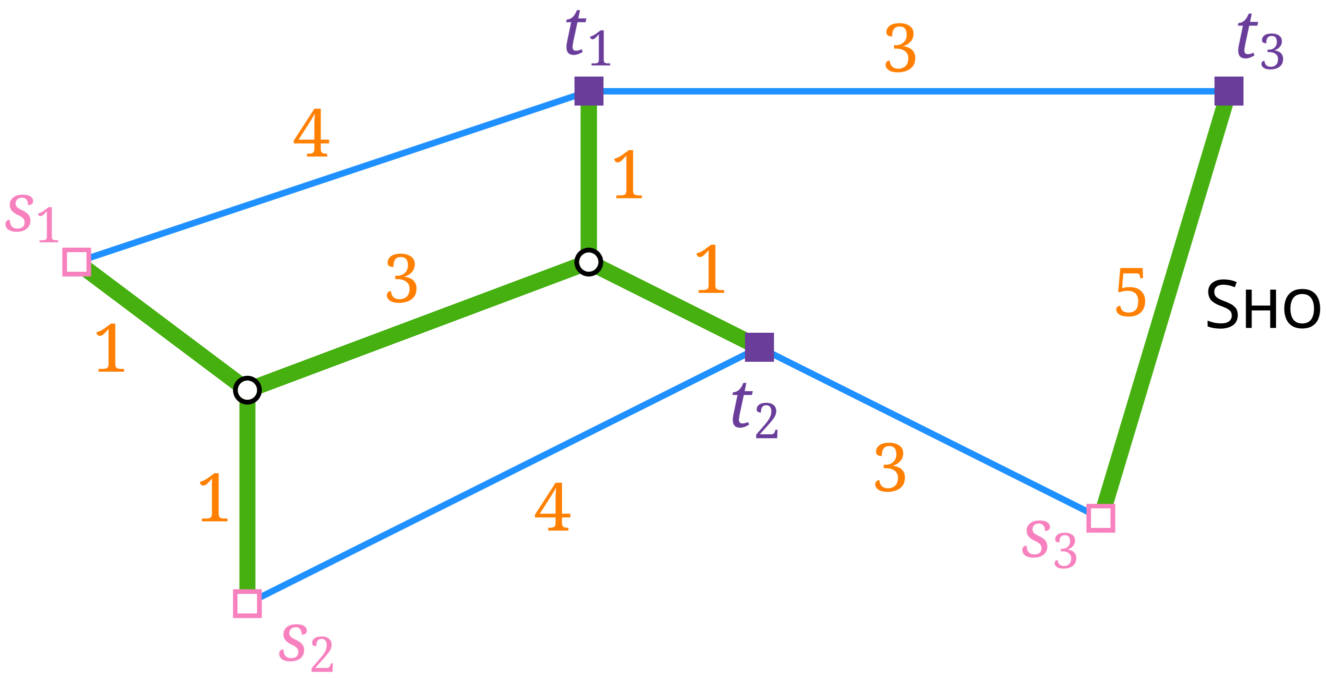


Special cases?

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 vertex pairs.

Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$ such that the subgraph (V, F) connects every pair (s_i, t_i), $i = 1, \dots, k$.



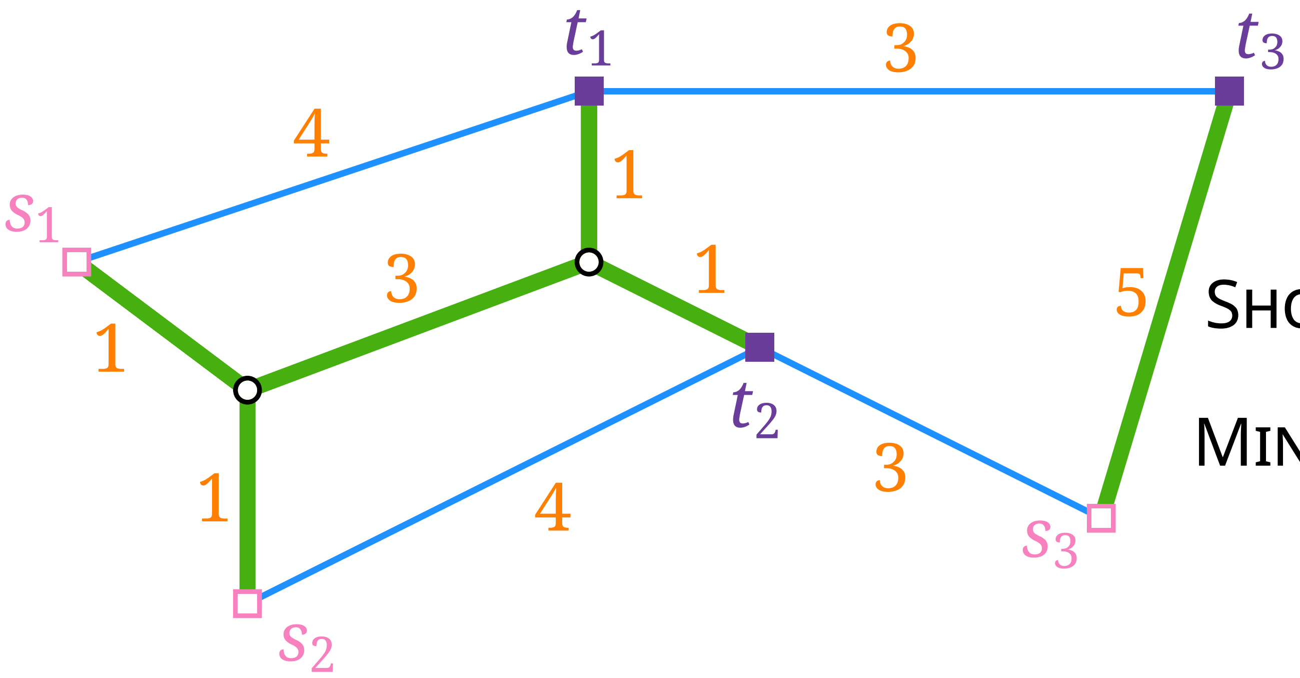
Special cases?

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

SteinerForest

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

Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$ such that the subgraph (V, F) connects every pair (s_i, t_i), $i = 1, \ldots, k$.



Special cases?

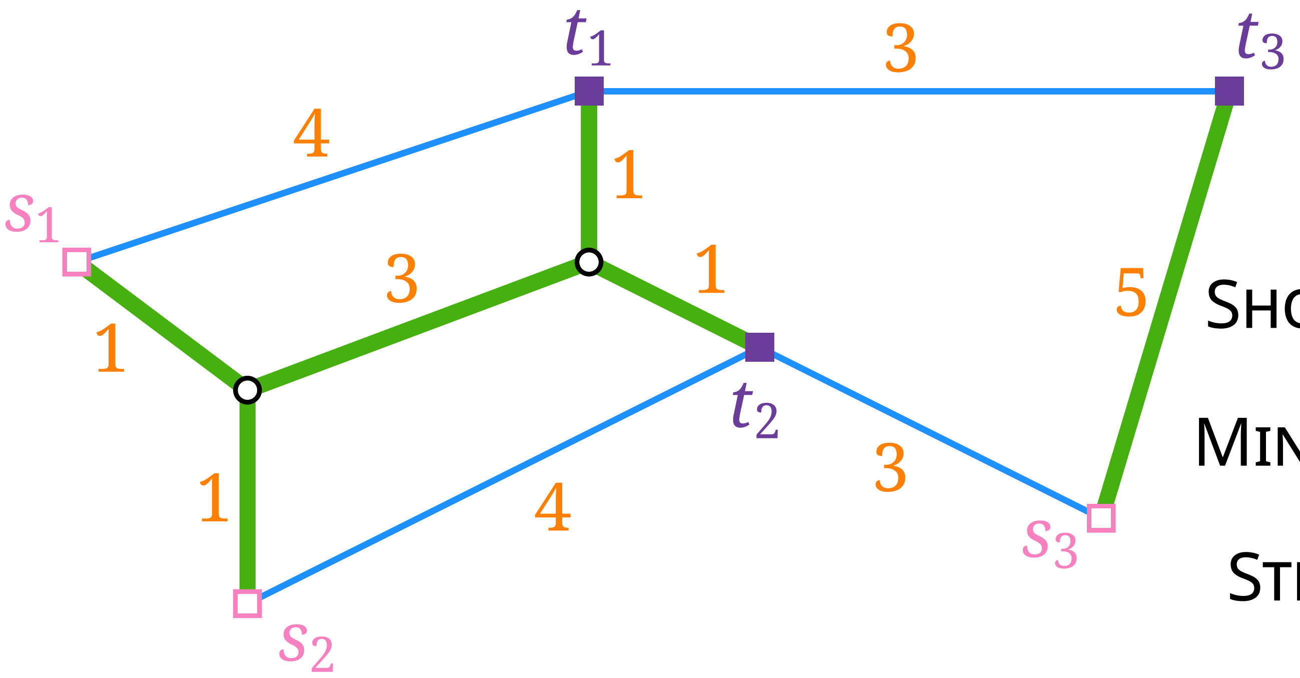
ShortestPath ($R = \{s, t\}$)

MinSpanningTree ($R = V \times V$)

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 vertex pairs.

Task: Find an edge set $F \subseteq E$ of minimum total cost $c(F)$ such that the subgraph (V, F) connects every pair (s_i, t_i), $i = 1, \ldots, k$.



Special cases?

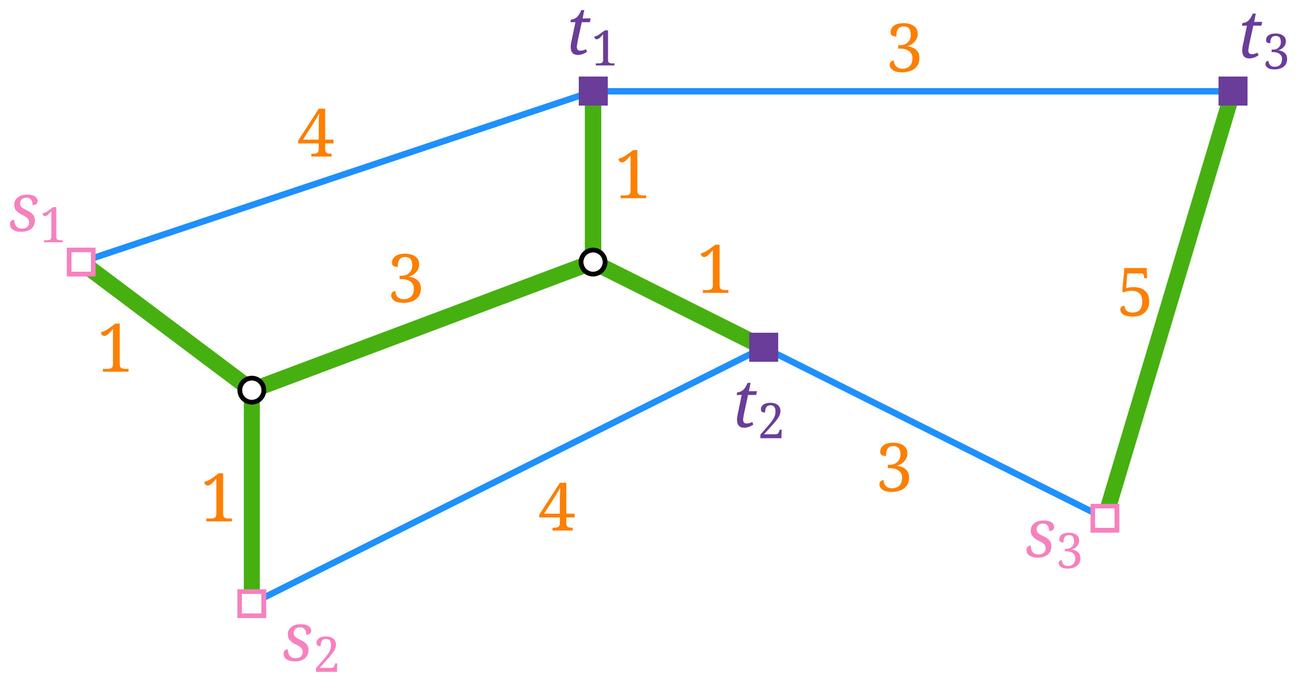
ShortestPath ($R = \{s, t\}$)

MinSpanningTree ($R = V \times V$)

SteinerTree ($R = T \times T$)

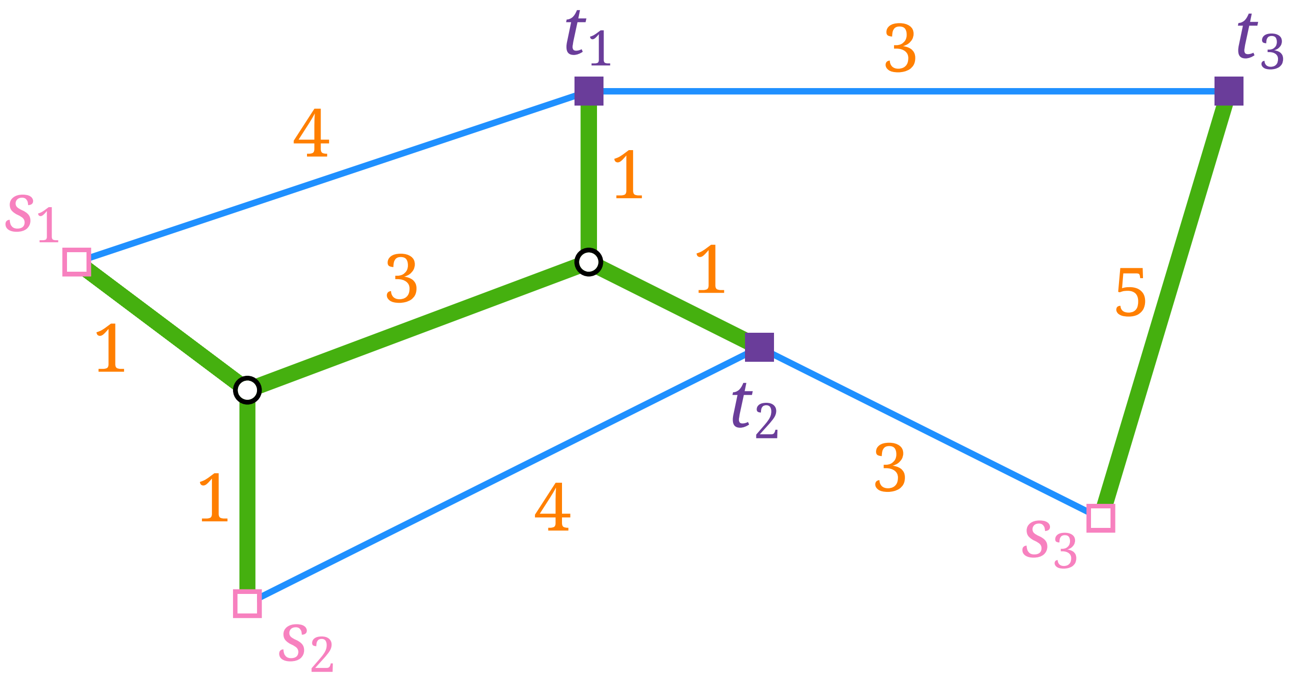
Approaches?

- Merge k shortest s_i–t_i paths



Approaches?

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

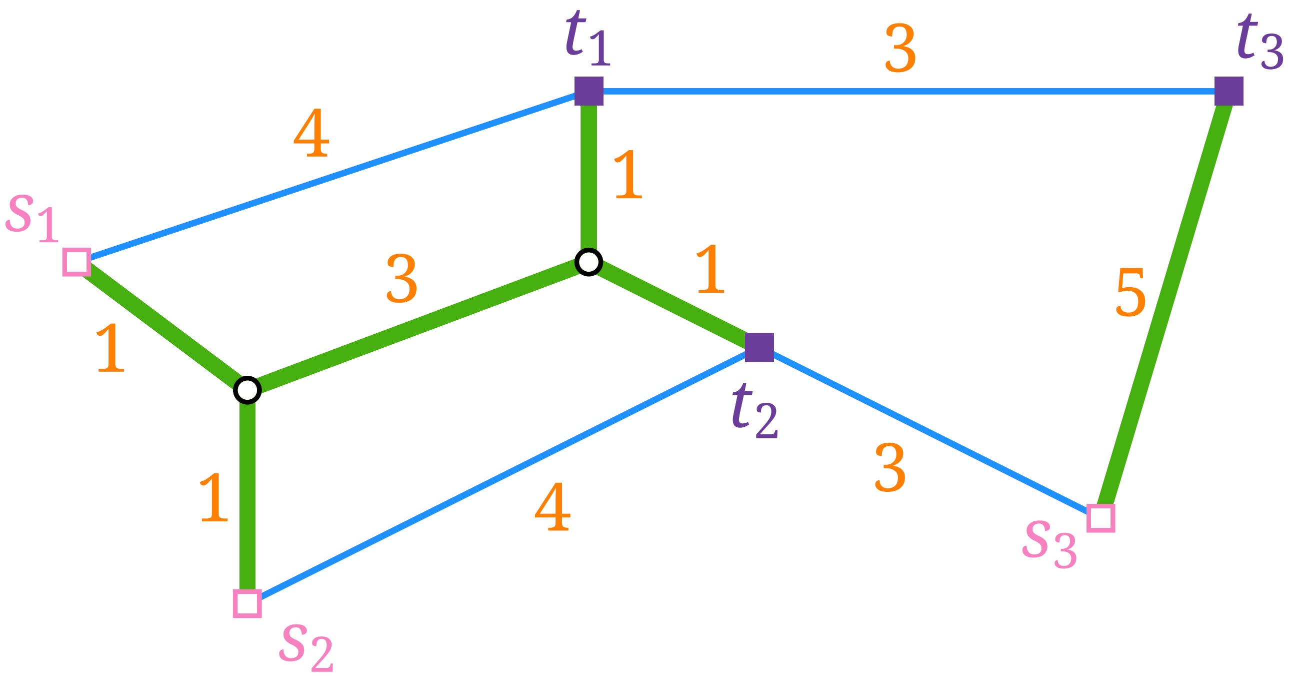


Approaches?

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

Above approaches perform poorly :-(

Why?

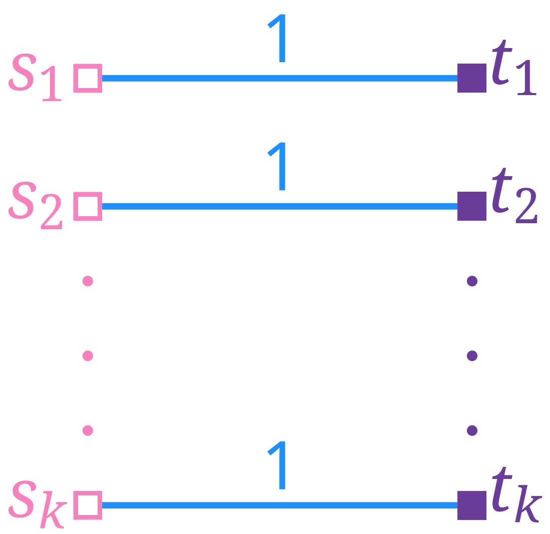


Approaches?

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

Above approaches perform poorly :-(

Why?

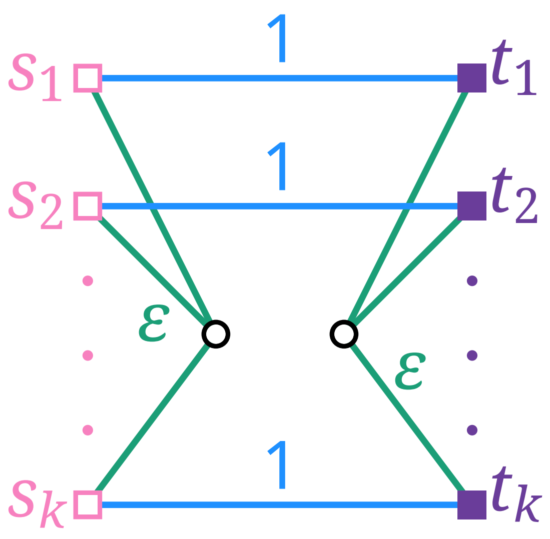


Approaches?

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

Above approaches perform poorly :-(

Why?

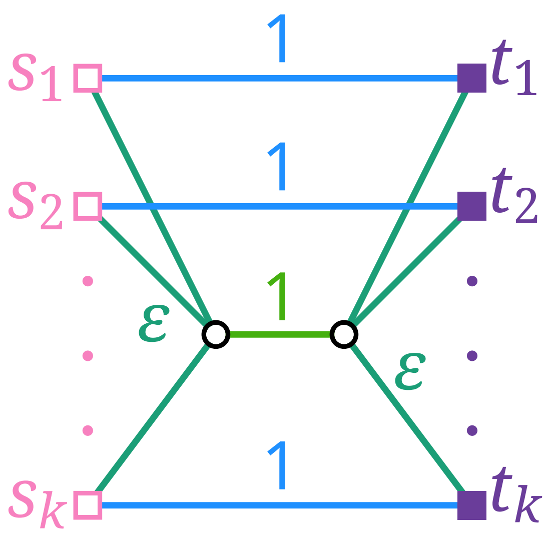


Approaches?

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

Above approaches perform poorly :-(

Why?

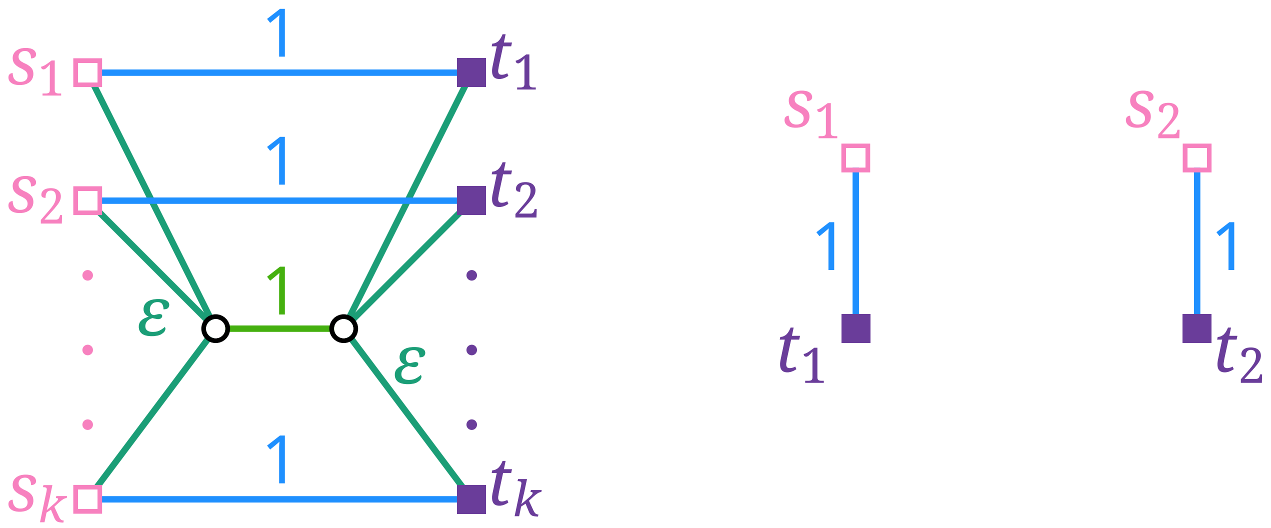


Approaches?

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

Above approaches perform poorly :-(

Why?

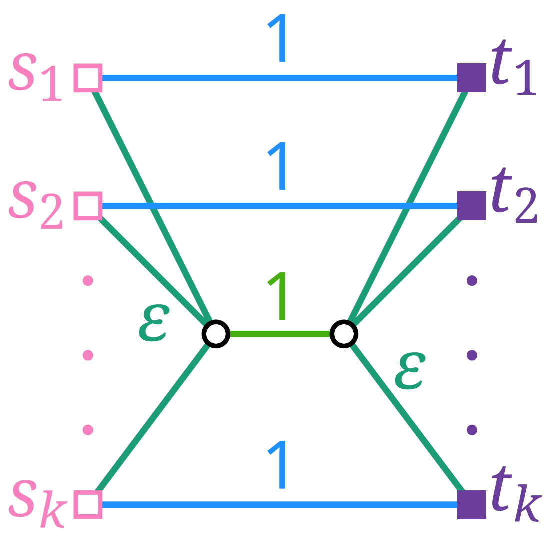


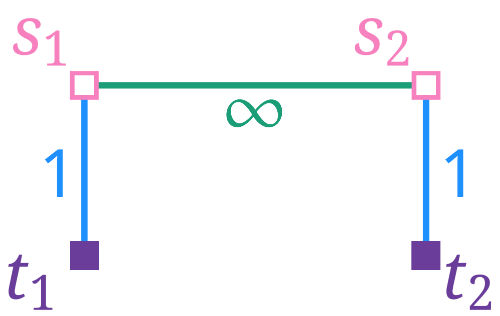
Approaches?

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

Above approaches perform poorly :-(

Why?



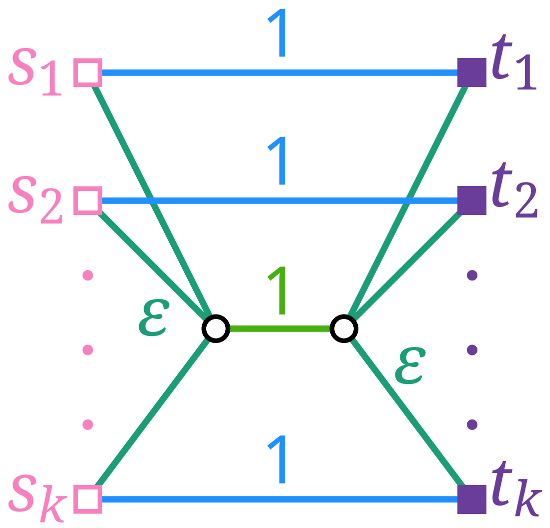


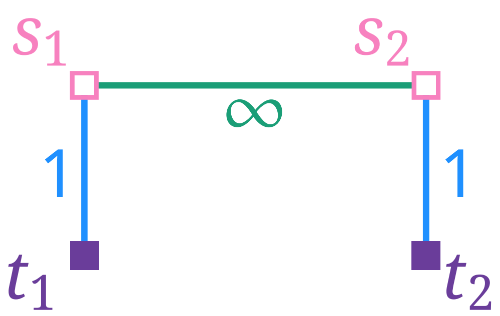
Approaches?

- 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?





Primal and Dual LP

ILP for SteinerForest

minimize

subject to

ILP for SteinerForest

minimize

subject to

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

ILP for SteinerForest

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

subject to

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

ILP for SteinerForest

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

subject to

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

How to ensure connectedness of all pairs in R?

ILP for SteinerForest

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

How to ensure connectedness of all pairs in R?

Using cuts?

ILP for SteinerForest

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

subject to

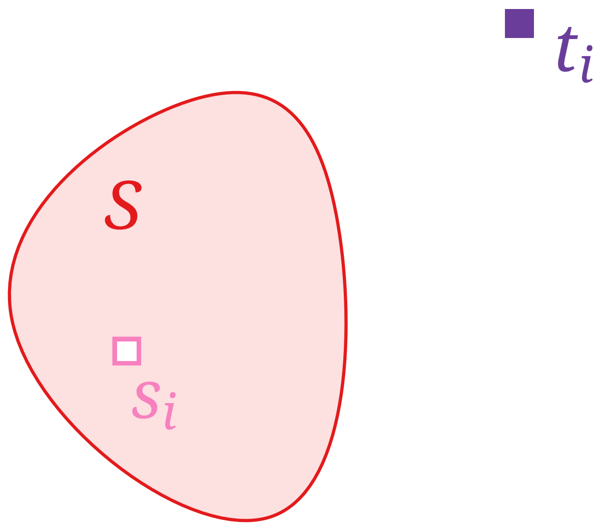
$x_e \in \{0, 1\} \qquad e \in E$

ILP for SteinerForest

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

subject to

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

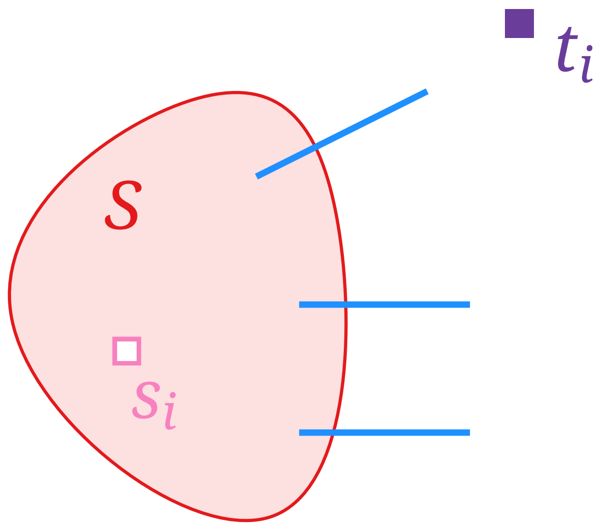


ILP for SteinerForest

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

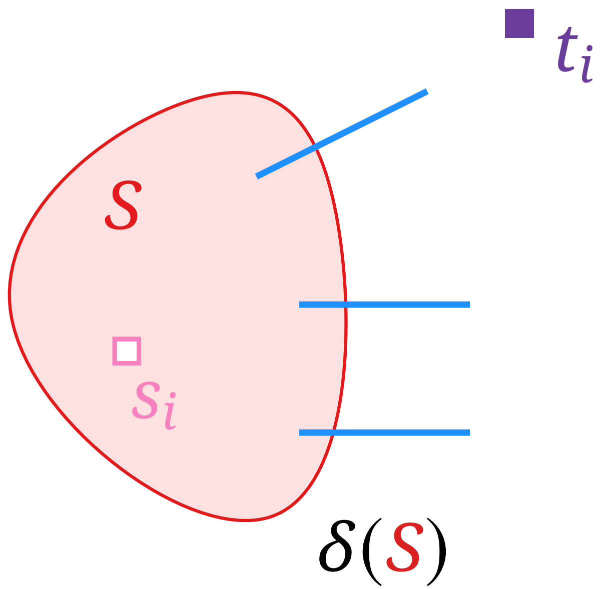
subject to

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



ILP for SteinerForest

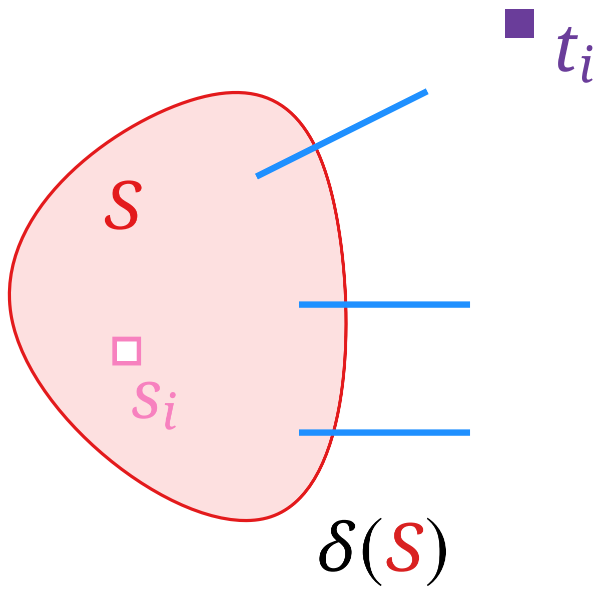
minimize	$\sum_{e \in E} c_e x_e$
subject to	
	$x_e \in \{0, 1\} \qquad e \in E$



ILP for SteinerForest

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

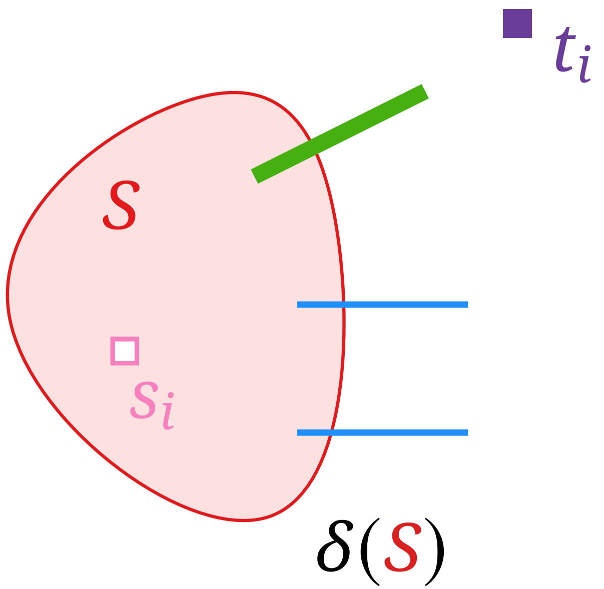
$\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$



ILP for SteinerForest

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

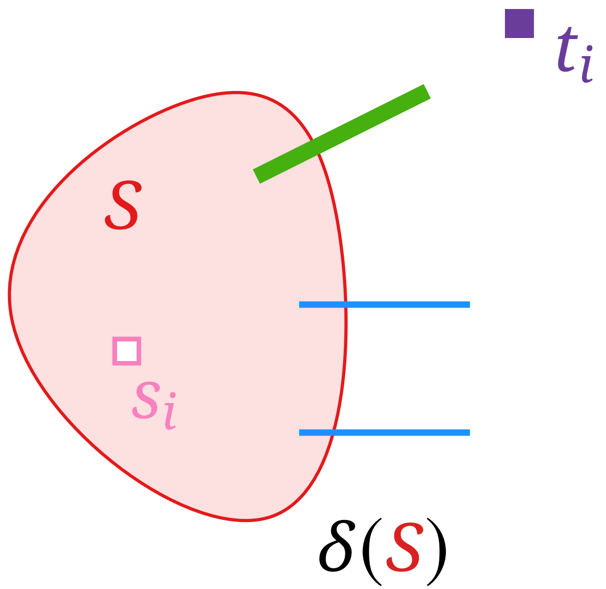
$\delta(S) := \{(u, v) \in E : u \in S \text{ and } v \notin S\}$



ILP for SteinerForest

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

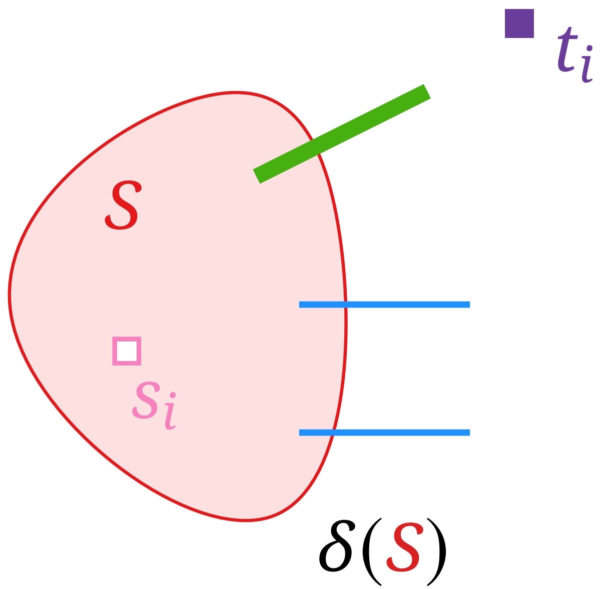
$$\delta(S) := \{(u,v) \in E : u \in S \text{ and } v \notin S\}$$



ILP for SteinerForest

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

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



ILP for SteinerForest

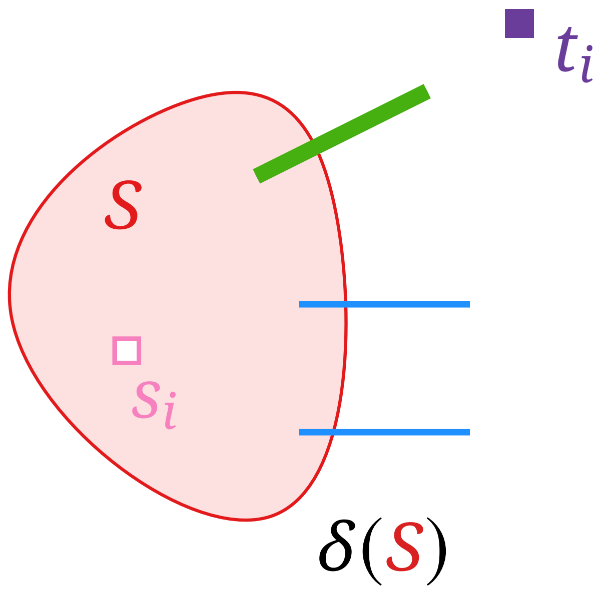
minimize	$\sum_{e \in E} c_e x_e$
subject to	$\sum_{e \in \delta(S)} x_e \geq 1 \qquad S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}$
	$x_e \in \{0, 1\} \qquad e \in E$

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

cuts separating s_i and t_i

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

cut edges



ILP for SteinerForest

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

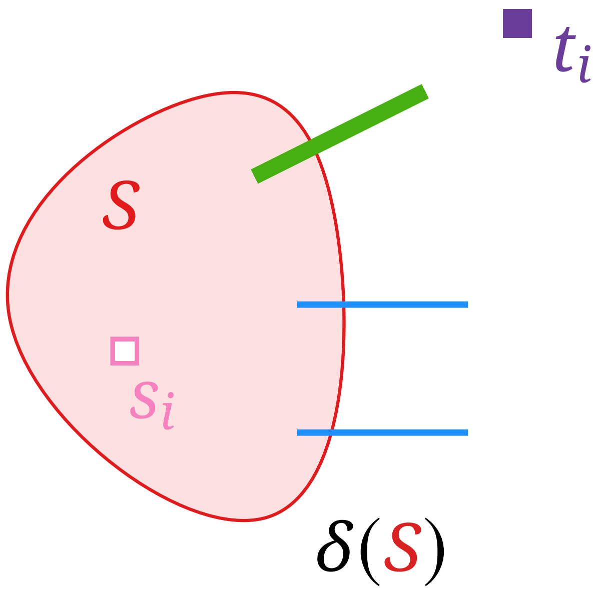
Why does this enforce connectedness?

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

cuts separating s_i and t_i

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

cut edges



ILP for SteinerForest

minimize	$\sum_{e \in E} c_e x_e$
subject to	$\sum_{e \in \delta(S)} x_e \geq 1 \qquad S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}$
	$x_e \in \{0, 1\} \qquad e \in E$

Why does this enforce connectedness?

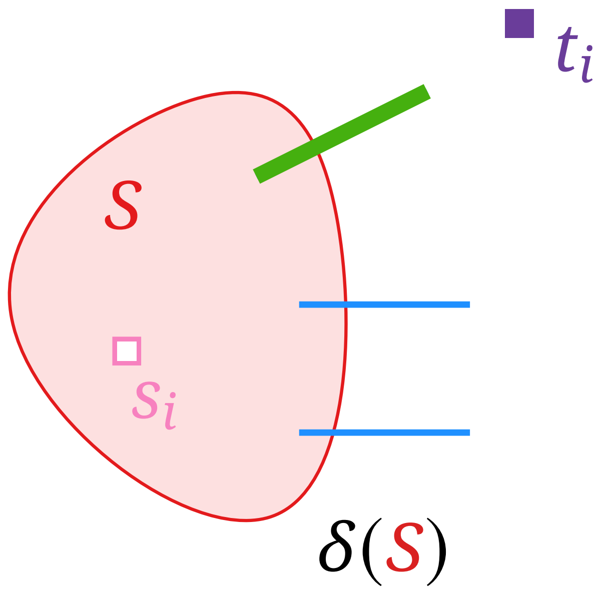
incrementally grow a path from s_i to t_i

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

cuts separating s_i and t_i

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

cut edges



ILP for SteinerForest

minimize	$\sum_{e \in E} c_e x_e$	
subject to	$\sum_{e \in \delta(S)} x_e \geq 1$	$S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}$
	$x_e \in \{0, 1\}$	$e \in E$

Why does this enforce connectedness?

incrementally grow a path from s_i to t_i

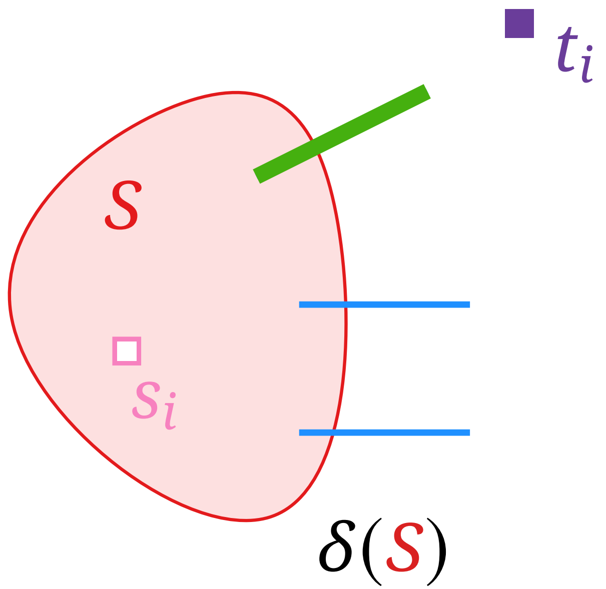
where $\mathcal{S}_i := \{S \subseteq V : |S \cap \{s_i, t_i\}| = 1\}$

cuts separating s_i and t_i

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

cut edges

$\rightsquigarrow$ exponentially many constraints!



LP-Relaxation and Dual LP

minimize	$\sum_{e \in E} c_e x_e$	
subject to	$\sum_{e \in \delta(S)} x_e \geq 1$	$S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}$
	$x_e \geq 0$	$e \in E$

LP-Relaxation and Dual LP

minimize	$\sum_{e \in E} c_e x_e$	
subject to	$\sum_{e \in \delta(S)} x_e \geq 1$	$S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}$ (y_S)
	$x_e \geq 0$	$e \in E$

LP-Relaxation and Dual LP

minimize	$\sum_{e \in E} c_e x_e$	
subject to	$\sum_{e \in \delta(S)} x_e \geq 1$	$S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}\ (y_S)$
	$x_e \geq 0$	$e \in E$

maximize		
subject to		
	$y_S \geq 0$	$S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}$

LP-Relaxation and Dual LP

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

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

LP-Relaxation and Dual LP

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

subject to $\displaystyle\sum_{e \in \delta(S)} x_e \geq 1 \qquad S \in \mathcal{S}_i,\ i \in \{1, \ldots, k\}\ (y_S)$

$x_e \geq 0 \qquad e \in E$

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

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

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

LP-Relaxation and Dual LP

covering LP

minimize	$\sum_{e\in E} c_e x_e$	
subject to	$\sum_{e\in\delta(S)} x_e \geq 1$	$S \in \mathcal{S}_i,\ i \in \{1,\ldots,k\}\ (y_S)$
	$x_e \geq 0$	$e \in E$

packing LP

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

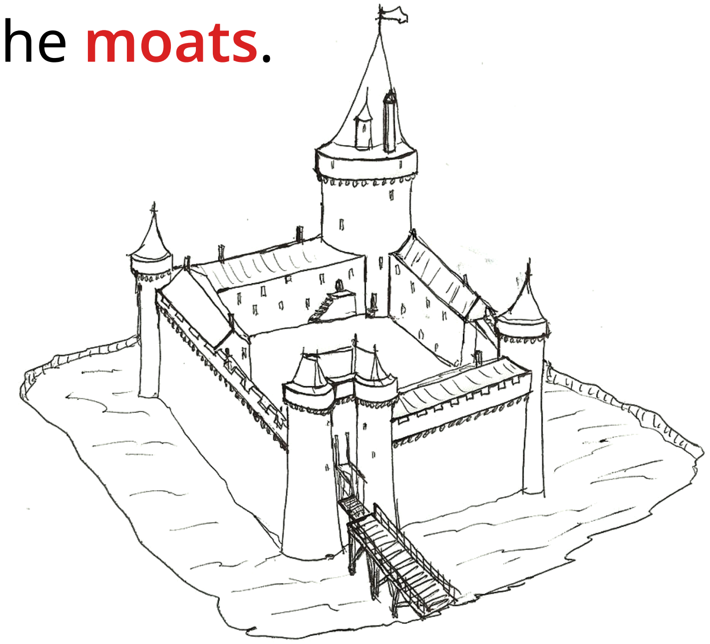
Intuition for the Dual

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

Intuition for the Dual

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

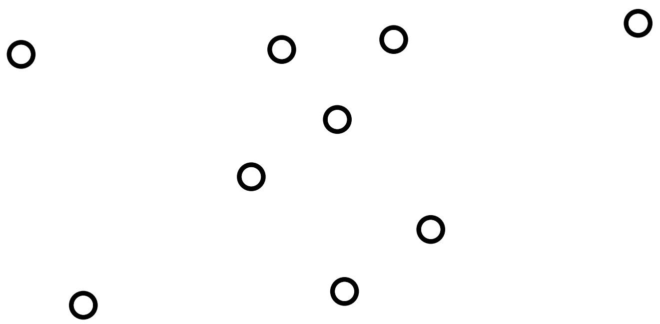
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

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

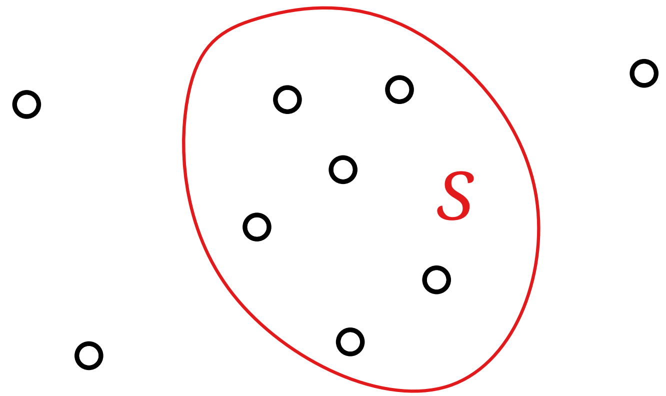
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

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

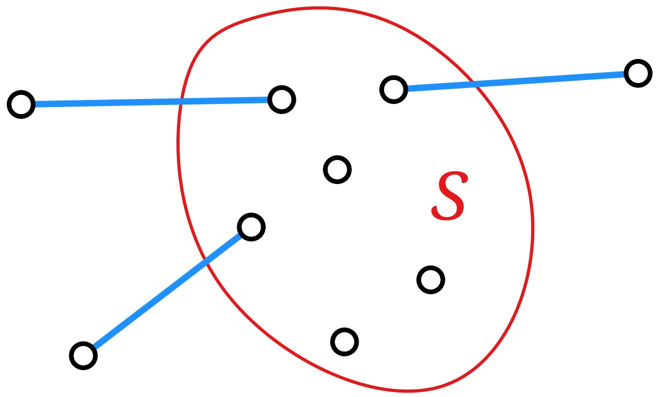
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

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

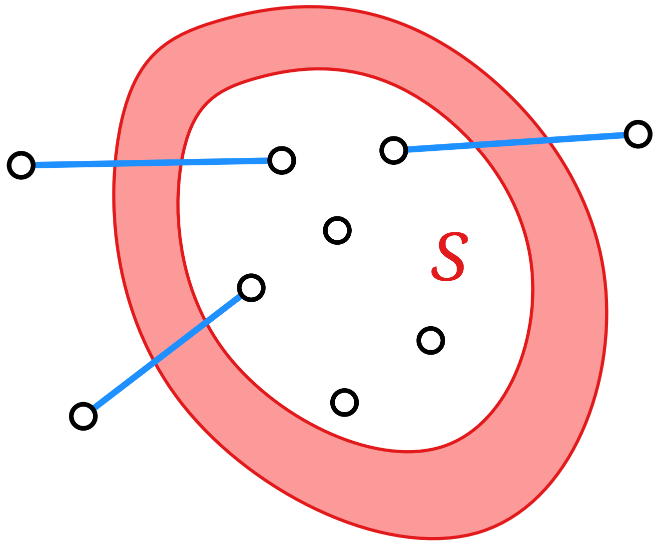
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

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

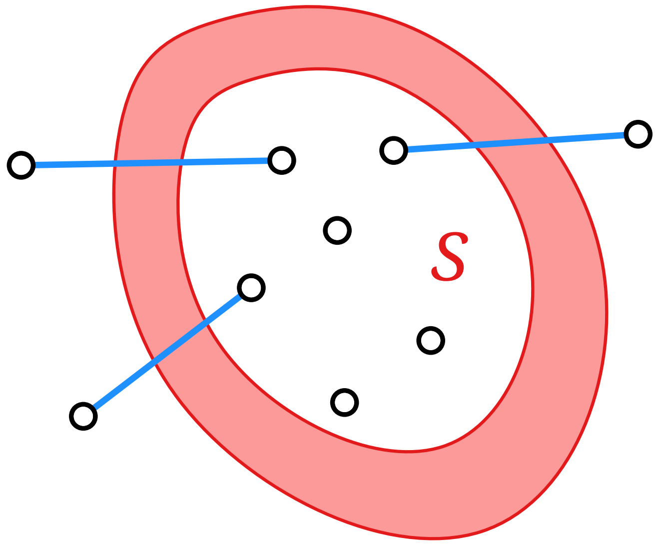
The graph is a network of **bridges**, spanning the **moats**.



Intuition for the Dual

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

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

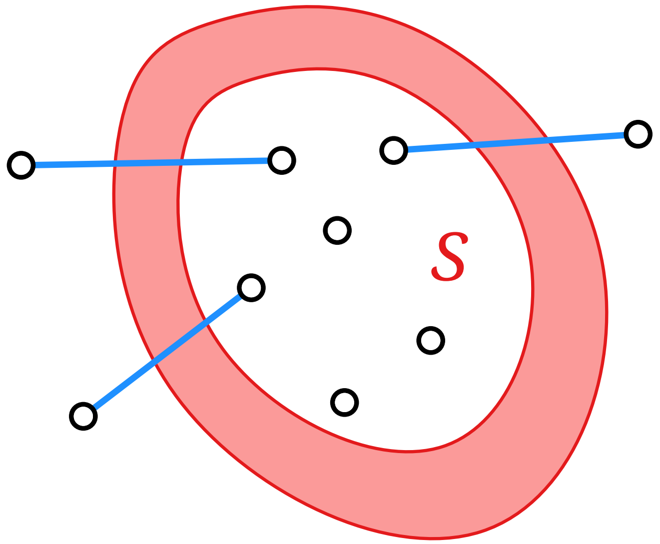


$\delta(S)$ = set of edges / bridges over the moat around S

Intuition for the Dual

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

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



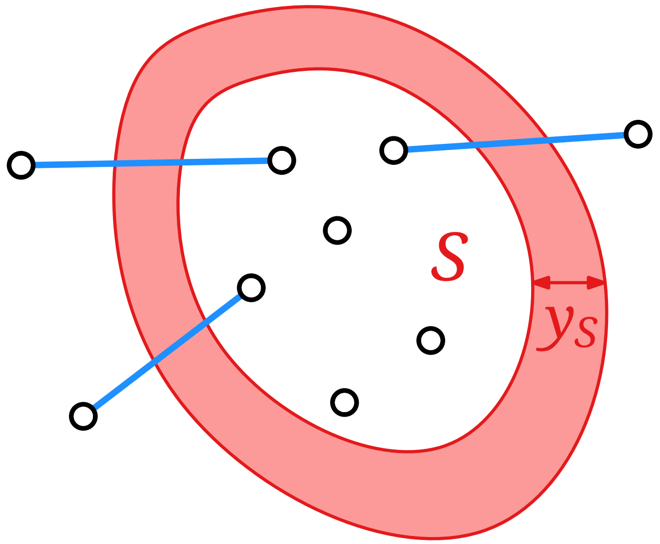
$\delta(S)$ = set of edges / bridges over the moat around S

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

Intuition for the Dual

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

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



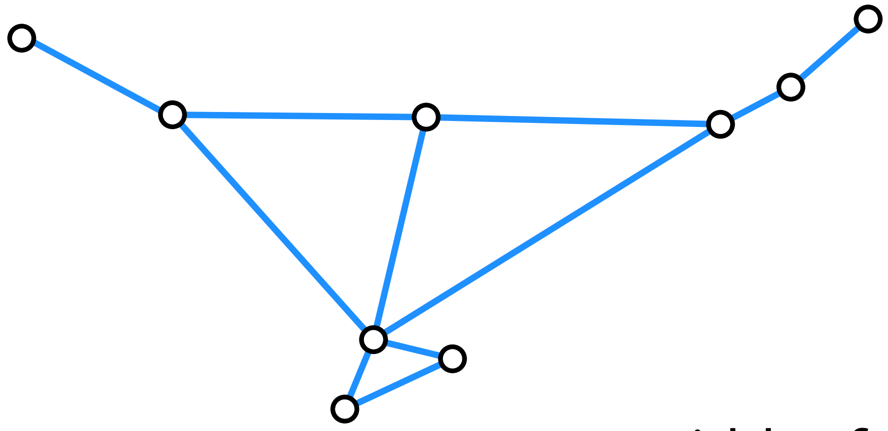
$\delta(S)$ = set of edges / bridges over the moat around S

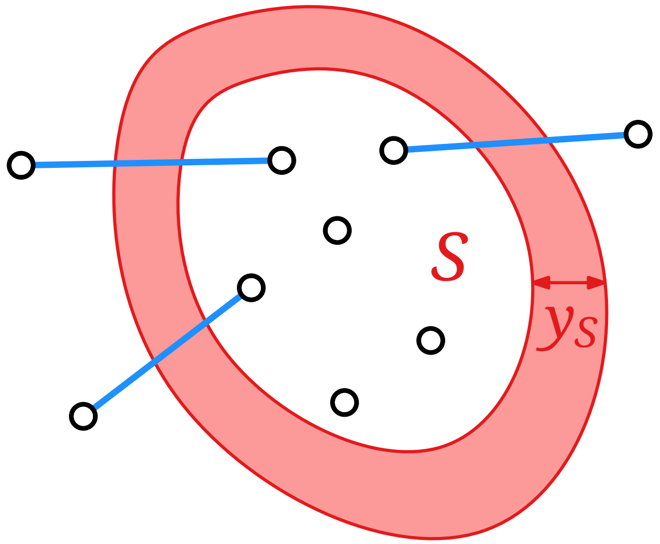
y_S = width of the **moat** around S

Intuition for the Dual

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

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





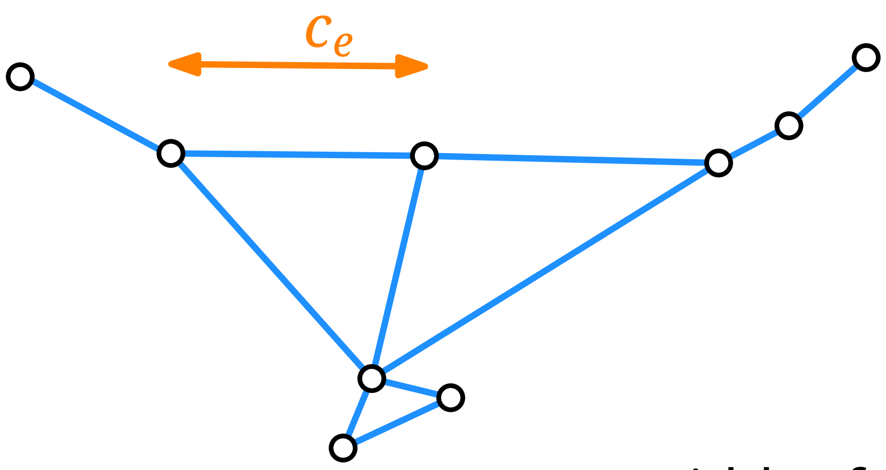
$\delta(S)$ = set of edges / bridges over the moat around S

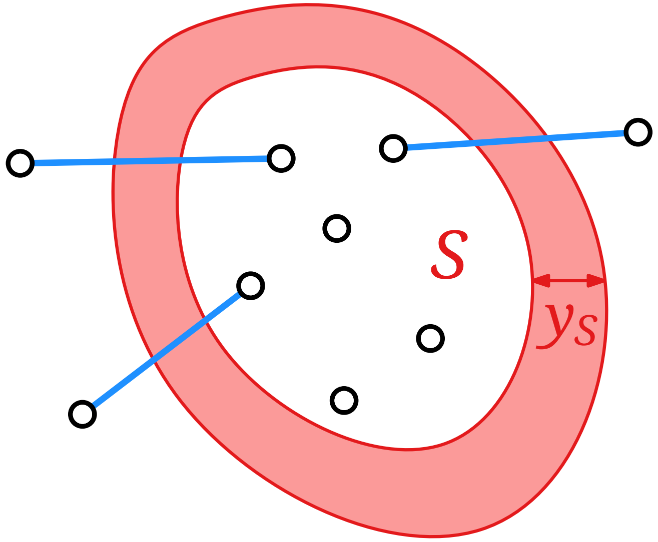
y_S = width of the **moat** around S

Intuition for the Dual

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

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





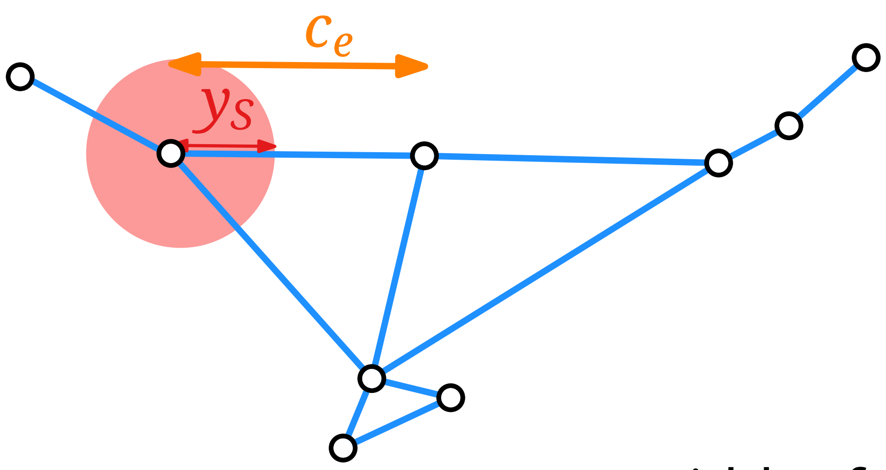
$\delta(S)$ = set of edges / bridges over the moat around S

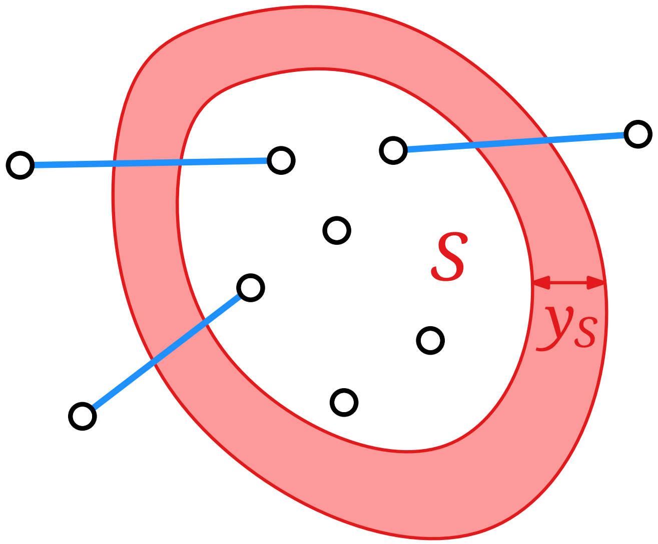
y_S = width of the **moat** around S

Intuition for the Dual

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

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





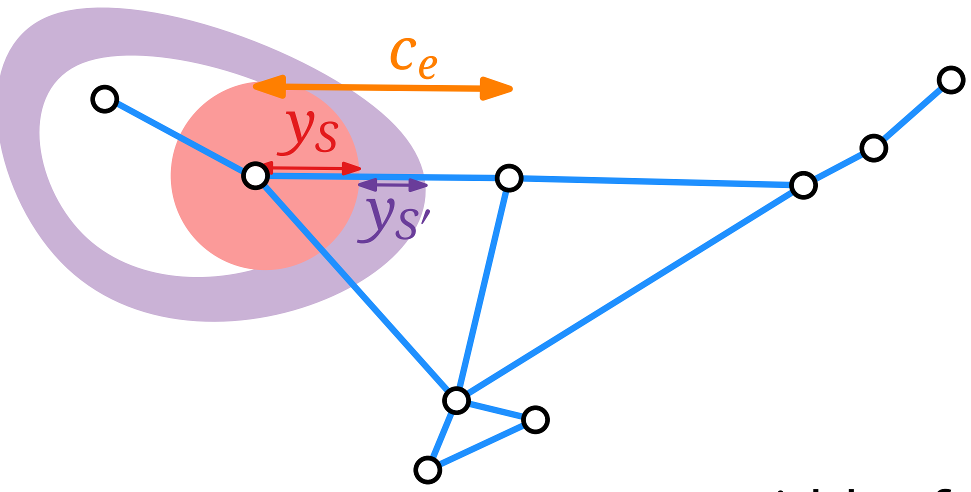
$\delta(S)$ = set of edges / bridges over the moat around S

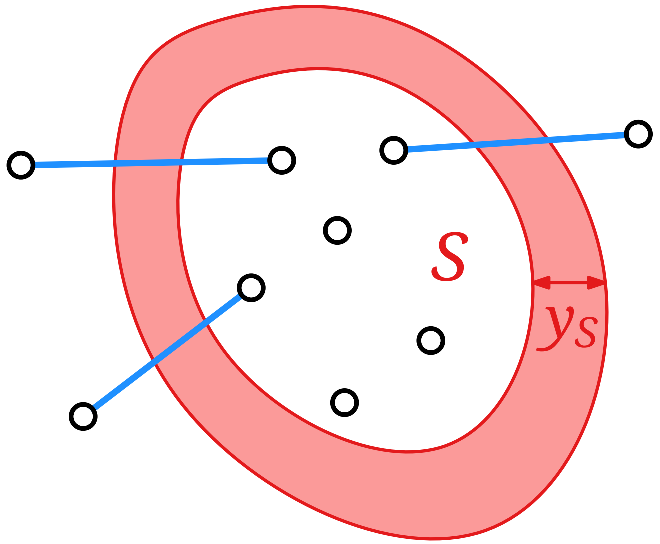
y_S = width of the **moat** around S

Intuition for the Dual

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

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





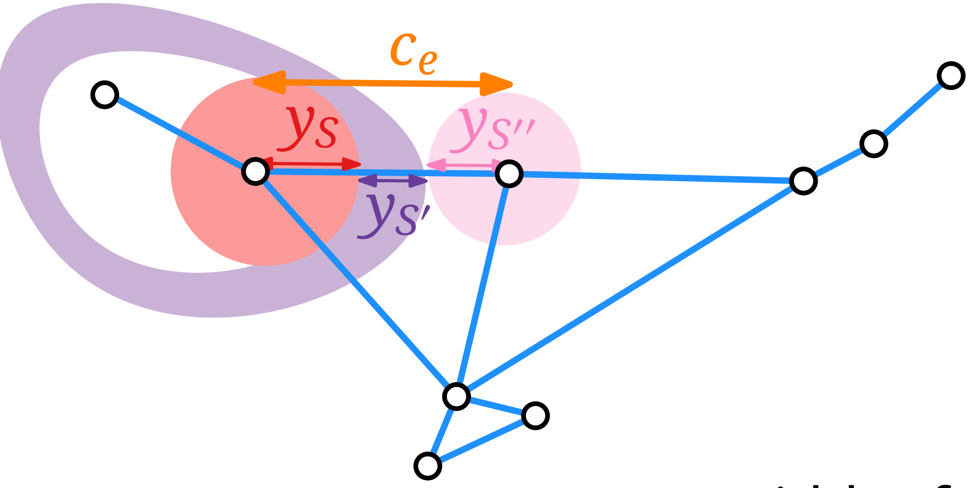
$\delta(S)$ = set of edges / bridges over the moat around S

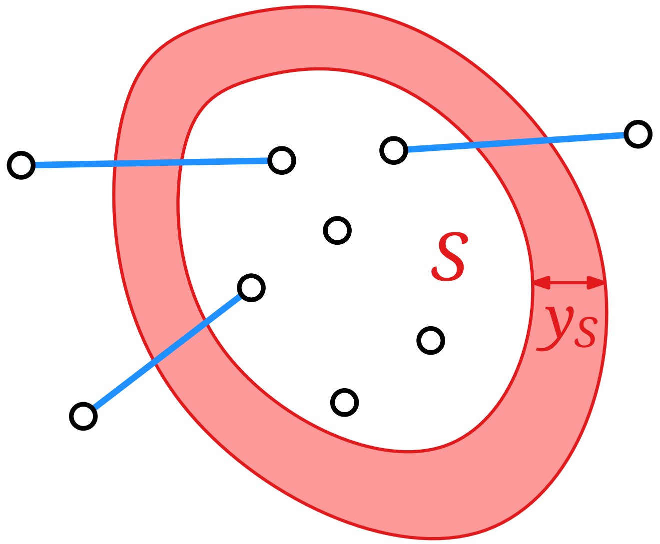
y_S = width of the **moat** around S

Intuition for the Dual

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

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





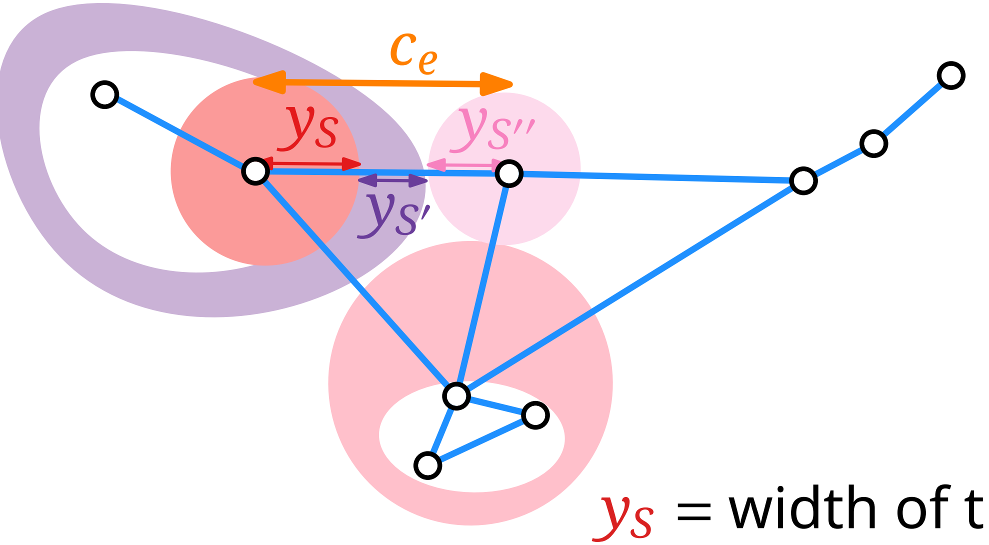
$\delta(S)$ = set of edges / bridges over the moat around S

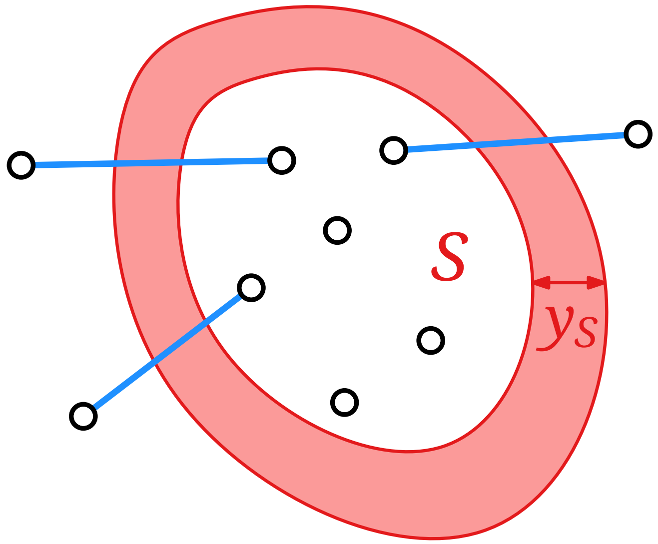
y_S = width of the **moat** around S

Intuition for the Dual

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

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

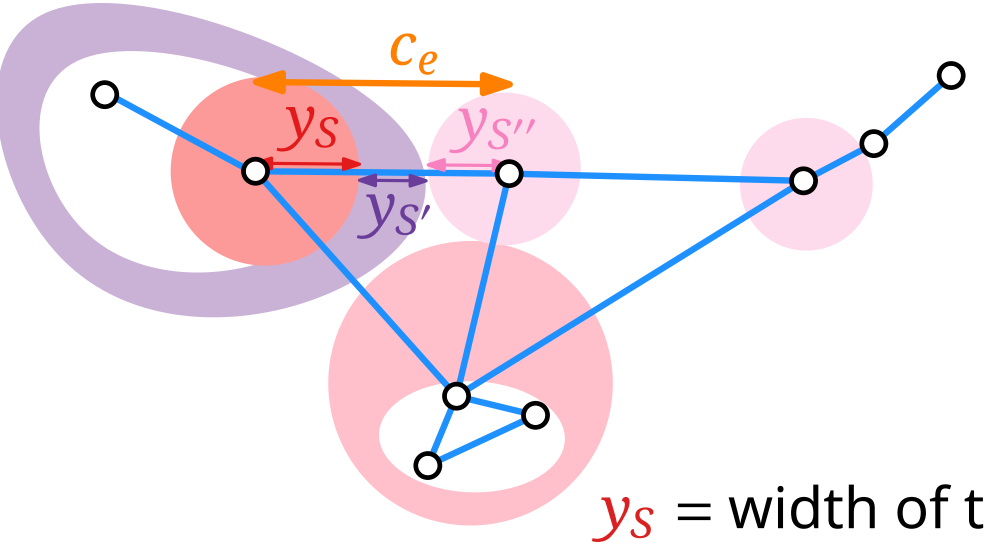
Intuition for the Dual

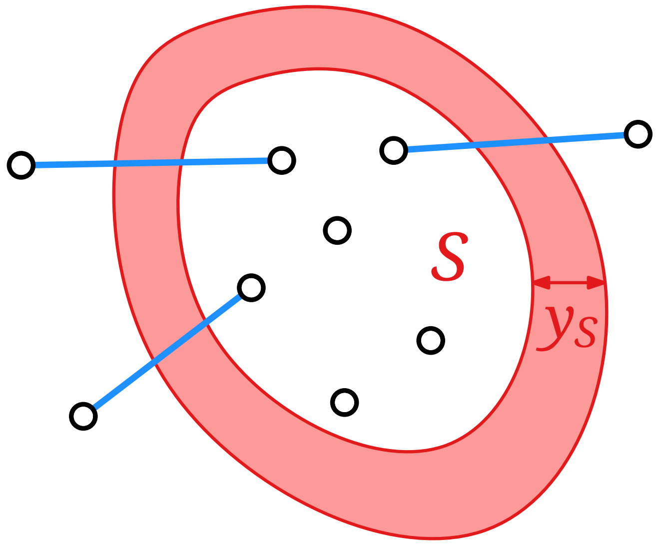
$$\text{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i \in \{1,\dots,k\}}} y_S$$

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

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

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

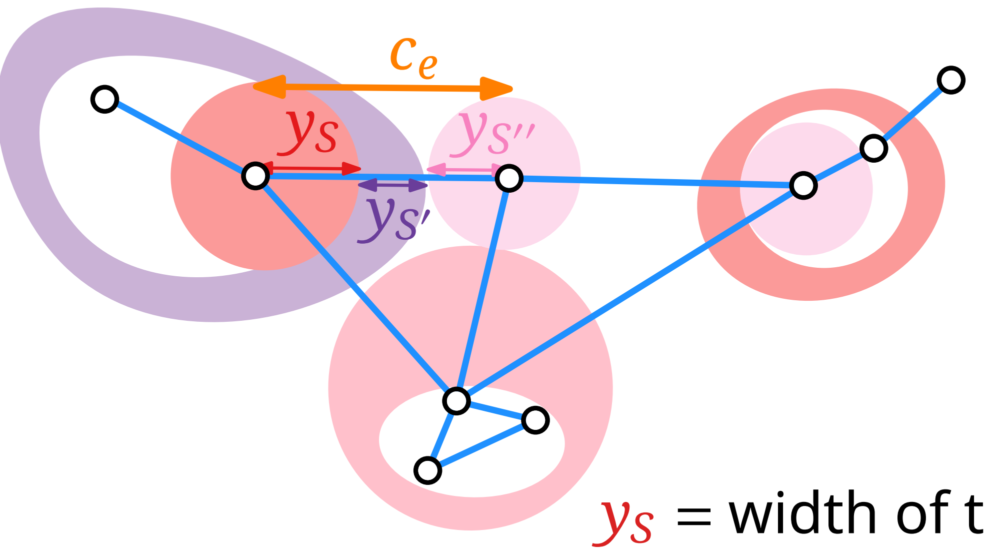
Intuition for the Dual

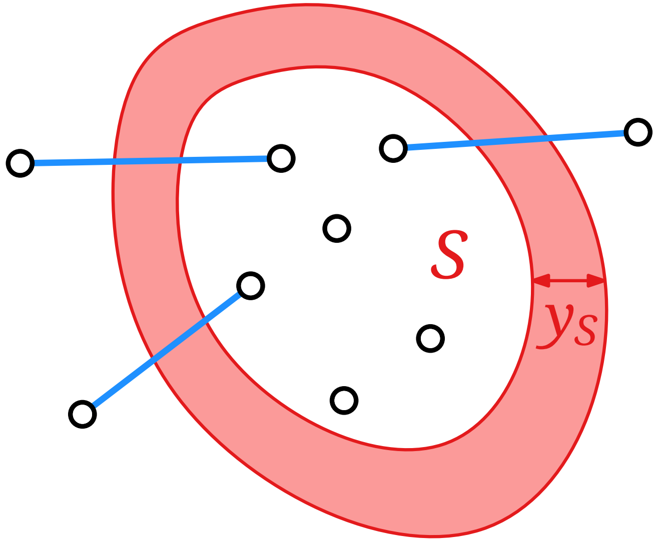
$$\text{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i \in \{1,\dots,k\}}} y_S$$

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

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

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





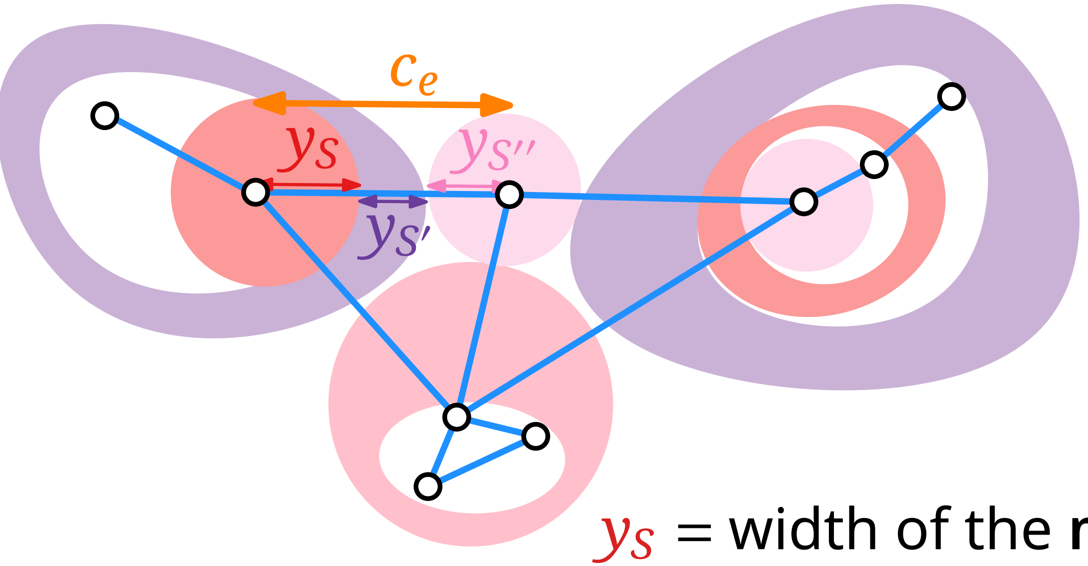
$\delta(S)$ = set of edges / bridges over the moat around S

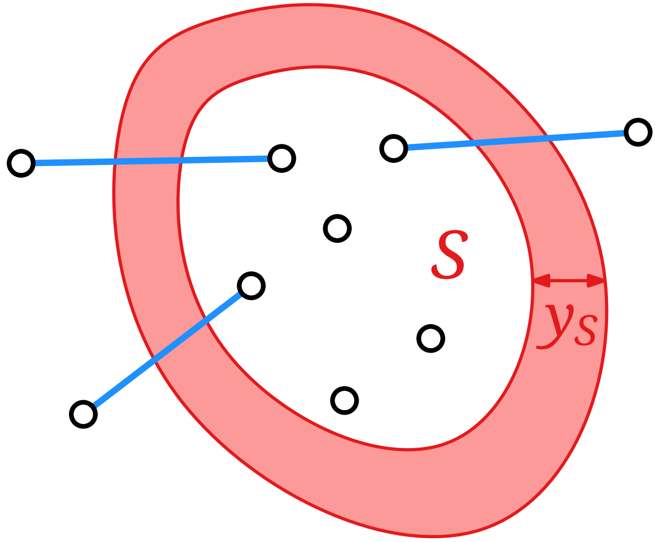
y_S = width of the **moat** around S

Intuition for the Dual

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

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

A First Primal–Dual Approach

Reminder: Complementary Slackness

minimize	$c^\top x$		
subject to	Ax	$\geq$	b
	x	$\geq$	0

maximize	$b^\top y$		
subject to	$A^\top y$	$\leq$	c
	y	$\geq$	0

Reminder: Complementary Slackness

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

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

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

A First Primal–Dual Approach

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

$\Rightarrow$ pick “critical” edges (and only those)

A First Primal–Dual Approach

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

$\Rightarrow$ pick “critical” edges (and only those)

Idea: iteratively build a feasible integral primal solution.

A First Primal–Dual Approach

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

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

$\Rightarrow$ 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 a corresponding connected component C!

A First Primal–Dual Approach

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

$\Rightarrow$ 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 a corresponding connected component C!

How do we iteratively improve the dual solution?

A First Primal–Dual Approach

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

$\Rightarrow$ 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 a corresponding connected 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 $(s_j, t_j) \in R$ 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 $(s_j, t_j) \in R$ not connected in (V, F) **do**

  $C \leftarrow$ component 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 $(s_j, t_j) \in R$ not connected in (V, F) **do**

- $C \leftarrow$ component 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 $(s_j, t_j) \in R$ not connected in (V, F) **do**

$C \leftarrow$ component 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 $(s_j, t_j) \in R$ not connected in (V, F) **do**

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

  Increase y_C

    until $\displaystyle\sum_{S\colon e' \in \delta(S)} y_S = c_{e'}$ 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 $(s_j, t_j) \in R$ not connected in (V, F) **do**

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

return F

Exponential Running Time?

A First Primal–Dual Approach

PrimalDualSteinerForestNaive(G, c, R)

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

while some $(s_j, t_j) \in R$ not connected in (V, F) **do**

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

Increase y_C

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

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

return F

Exponential 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

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

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 $y_S > 0$:

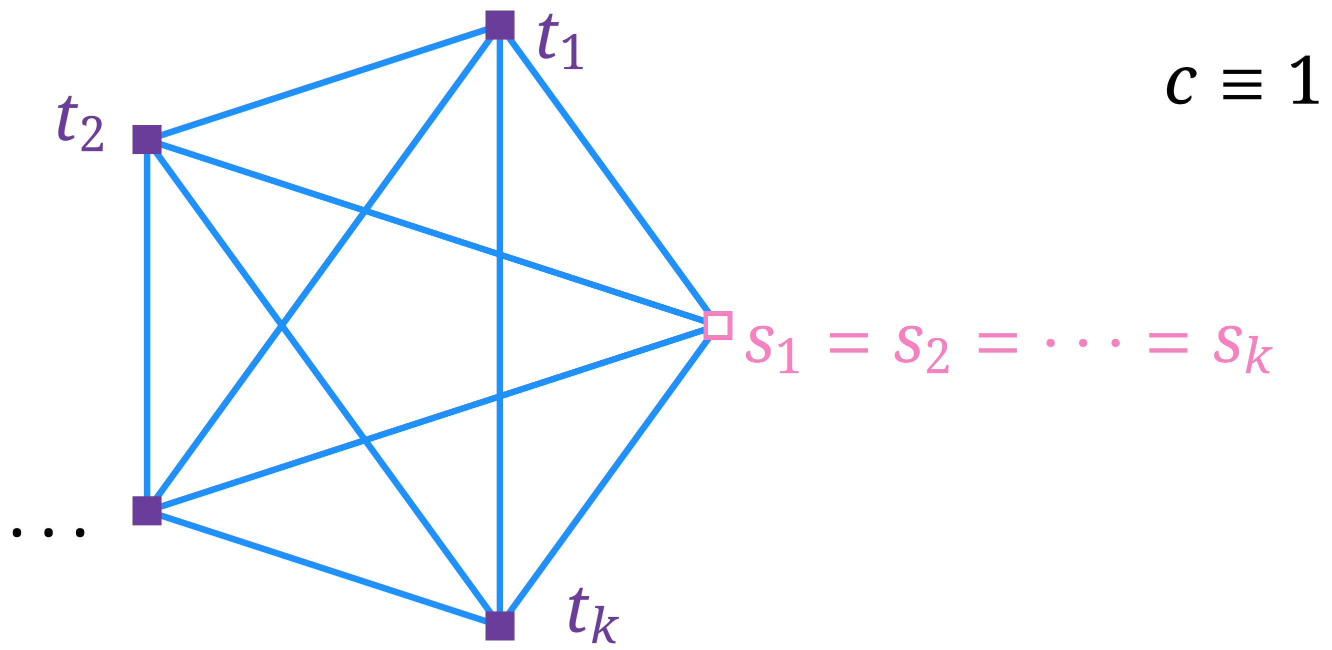
Analysis

The cost of the solution F can be written as

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

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

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



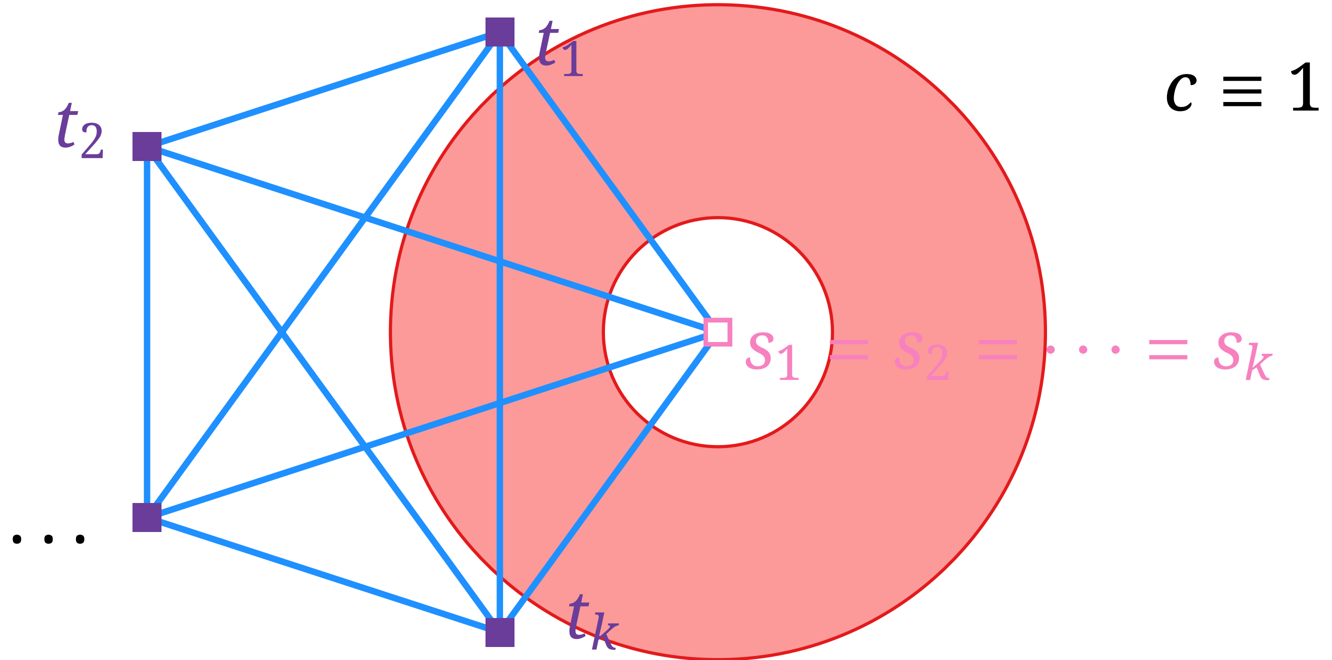
Analysis

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 $y_S > 0$:



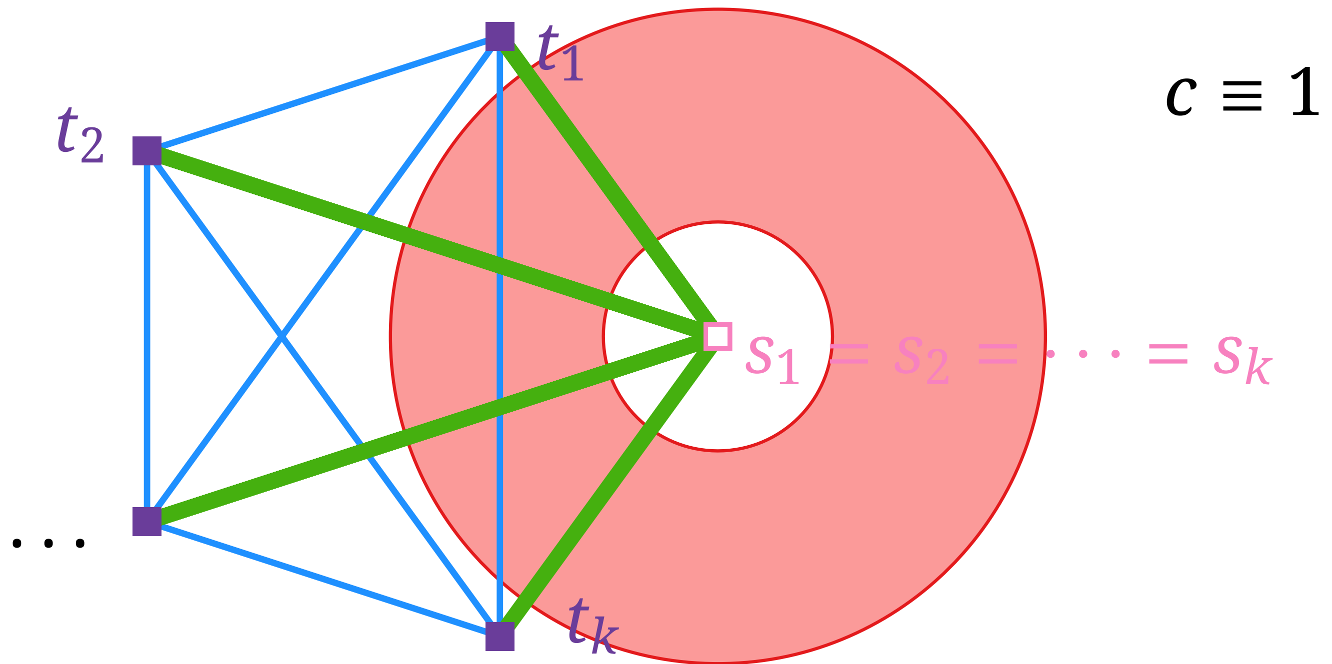
Analysis

The cost of the solution F can be written as

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

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

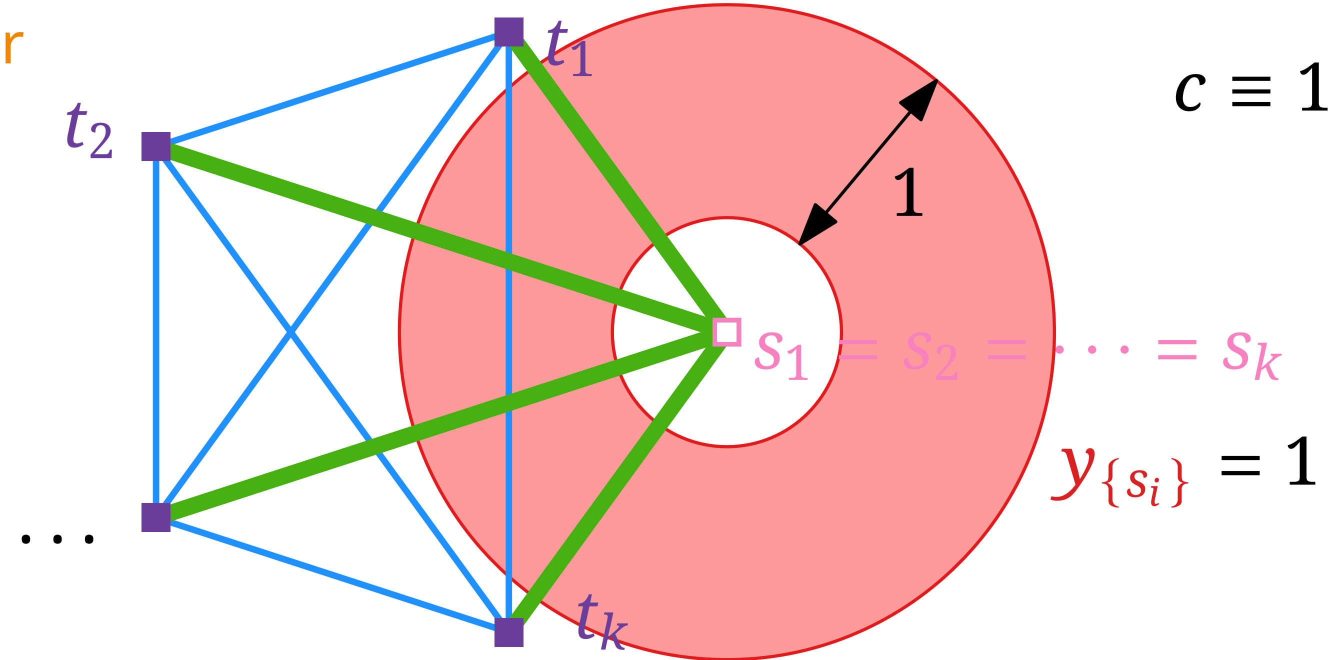
The cost of the solution F can be written as

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

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

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

distribution of cost over more cuts?



Analysis

The cost of the solution F can be written as

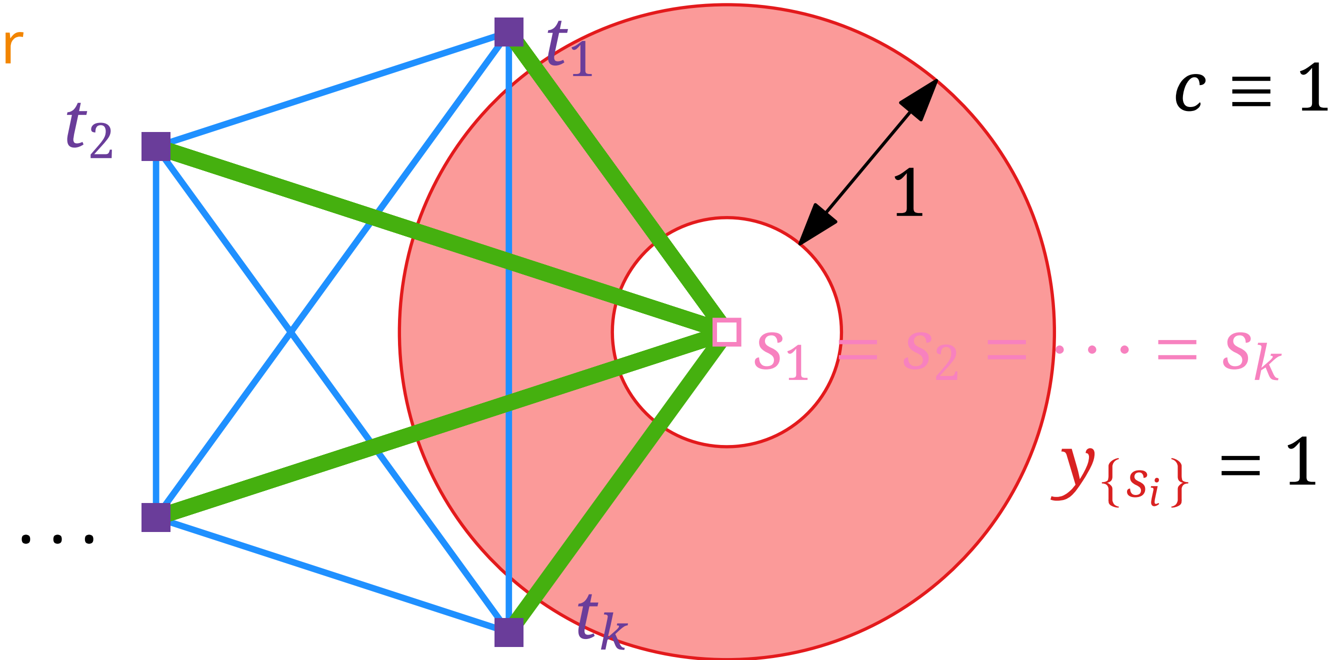
$$\sum_{e \in F} c_e \overset{\text{CS}}{=} \sum_{e \in F} \sum_{S:\, e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

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

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

distribution of cost over more cuts?

Initially 1 component per terminal



Analysis

The cost of the solution F can be written as

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

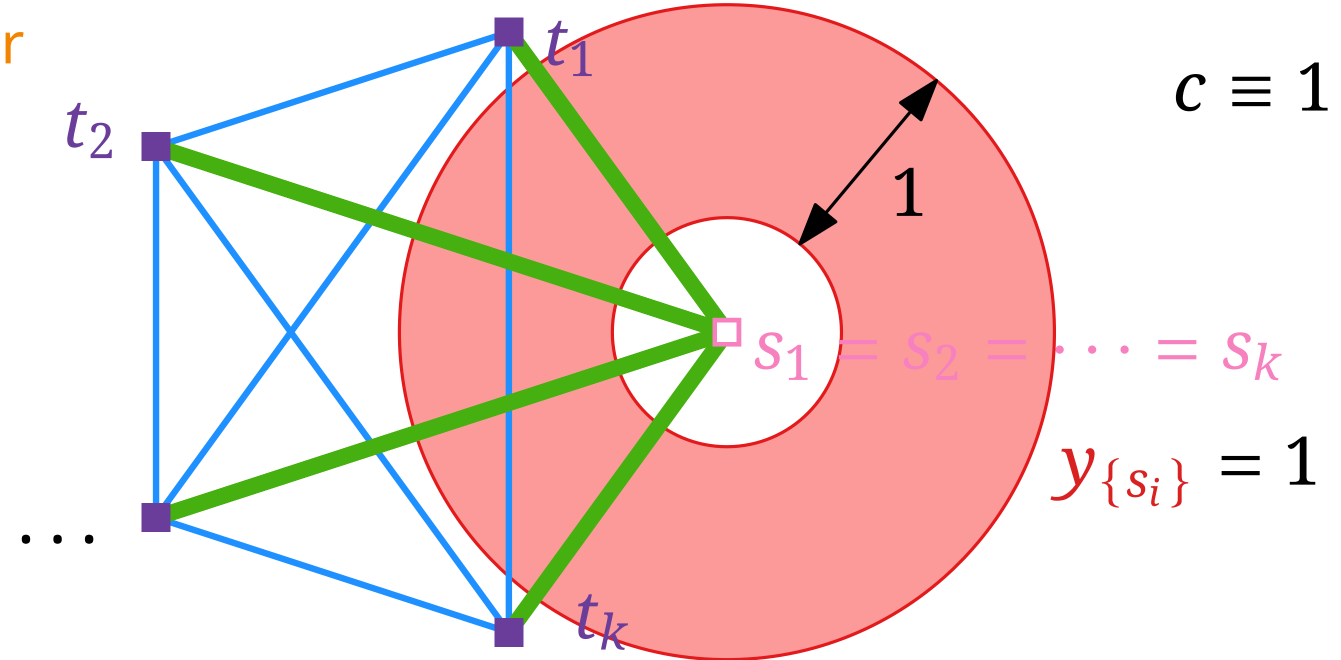
Compare to the value of the dual objective function $\sum_S y_S$

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

distribution of cost over more cuts?

Initially 1 component per terminal

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



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 $\sum_S y_S$

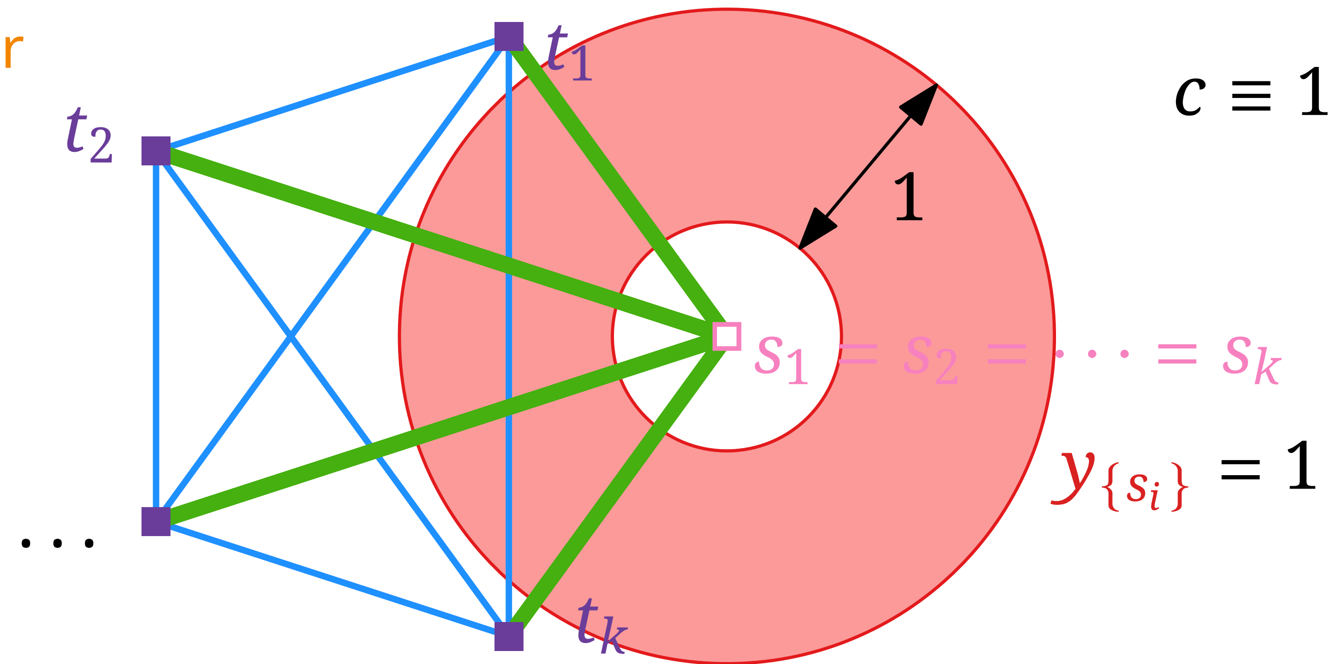
How to choose c_e in example to get large approximation factor?

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

distribution of cost over more cuts?

Initially 1 component per terminal

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



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 $\sum_S y_S$

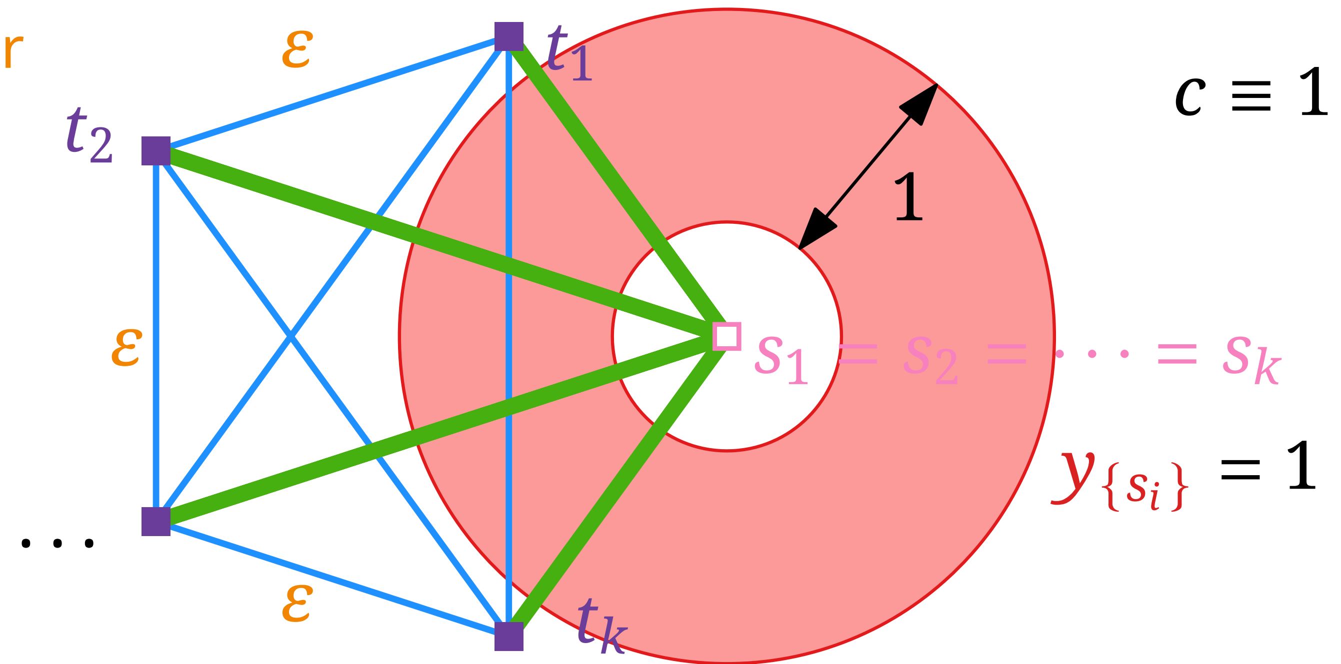
How to choose c_e in example to get large approximation factor?

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

distribution of cost over more cuts?

Initially 1 component per terminal

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



Analysis

The cost of the solution F can be written as

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

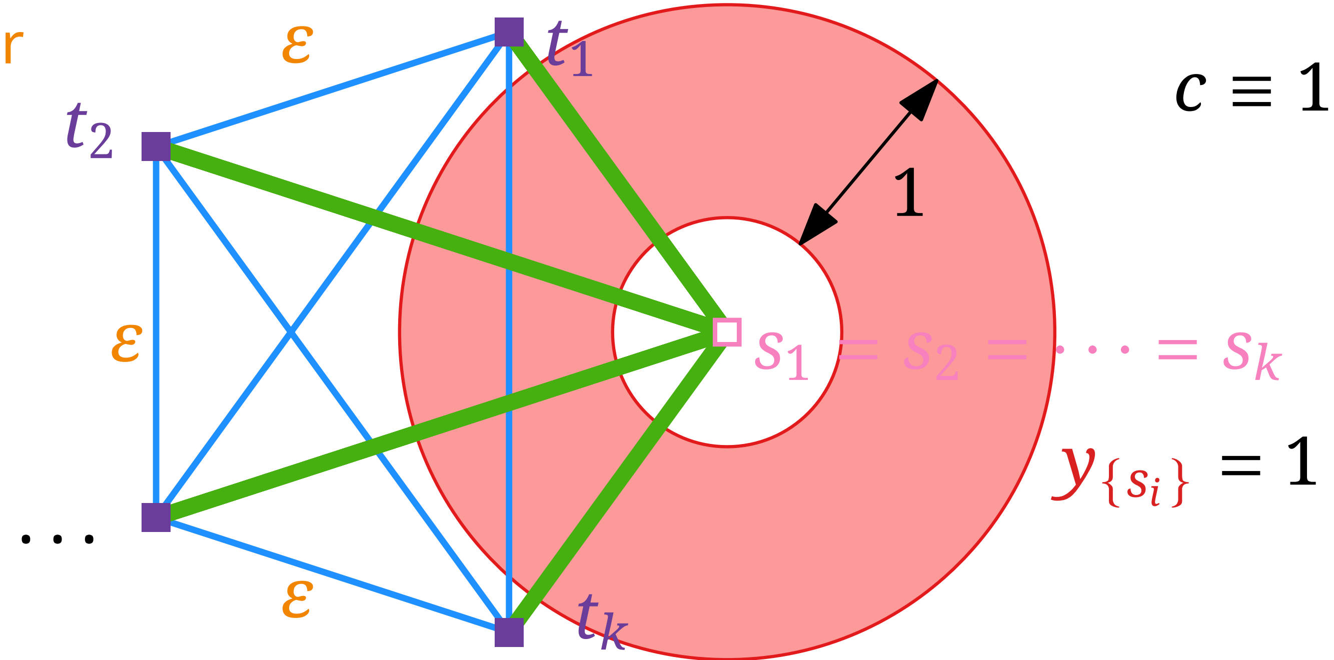
Compare to the value of the dual objective function $\sum_S y_S$

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

distribution of cost over more cuts?

Initially 1 component per terminal

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



$c \equiv 1$

How to choose c_e in example to get large approximation factor?

but again, simultaneous increase helps

Primal–Dual with Synchronized Increases

Primal–Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

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

while some $(s_j, t_j) \in R$ not connected in (V, F) **do**

$\quad \ell \leftarrow \ell + 1$

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

Primal–Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

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

while some $(s_j, t_j) \in R$ not connected in (V, F) **do**

  $\ell \leftarrow \ell + 1$

  $\mathcal{A} \leftarrow$ {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 $(s_j, t_j) \in R$ not connected in (V, F) **do**

$\ell \leftarrow \ell + 1$

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

Increase y_C for all $C \in \mathcal{A}$ 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 $(s_j, t_j) \in R$ not connected in (V, F) **do**

- $\ell \leftarrow \ell + 1$
- $\mathcal{A} \leftarrow$ {comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i}
- Increase y_C for all $C \in \mathcal{A}$ simultaneously
  until $\sum_{S\colon e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C)$, $C \in \mathcal{A}$.
- $F \leftarrow F \cup \{e_\ell\}$

Primal–Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

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

while some $(s_j, t_j) \in R$ not connected in (V, F) **do**

$\quad \ell \leftarrow \ell + 1$

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

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

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

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

$F' \leftarrow F$

return F'

Primal–Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

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

while some $(s_j, t_j) \in R$ not connected in (V, F) **do**

  $\ell \leftarrow \ell + 1$

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

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

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

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

$F' \leftarrow F$

// Pruning

return F'

Primal–Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

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

while some $(s_j, t_j) \in R$ not connected in (V, F) **do**

  $\ell \leftarrow \ell + 1$

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

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

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

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

$F' \leftarrow F$

// 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 $(s_j, t_j) \in R$ not connected in (V, F) **do**
  $\ell \leftarrow \ell + 1$
  $\mathcal{A} \leftarrow$ {comp. C in (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i}
  Increase y_C for all $C \in \mathcal{A}$ simultaneously
    until $\displaystyle\sum_{S\colon e_\ell \in \delta(S)} y_S = c_{e_\ell}$ for some $e_\ell \in \delta(C)$, $C \in \mathcal{A}$.
  $F \leftarrow F \cup \{e_\ell\}$

$F' \leftarrow F$
// Pruning
for $j \leftarrow \ell$ **down to** 1 **do**
  if $F' \setminus \{e_j\}$ is feasible solution **then**

return F'

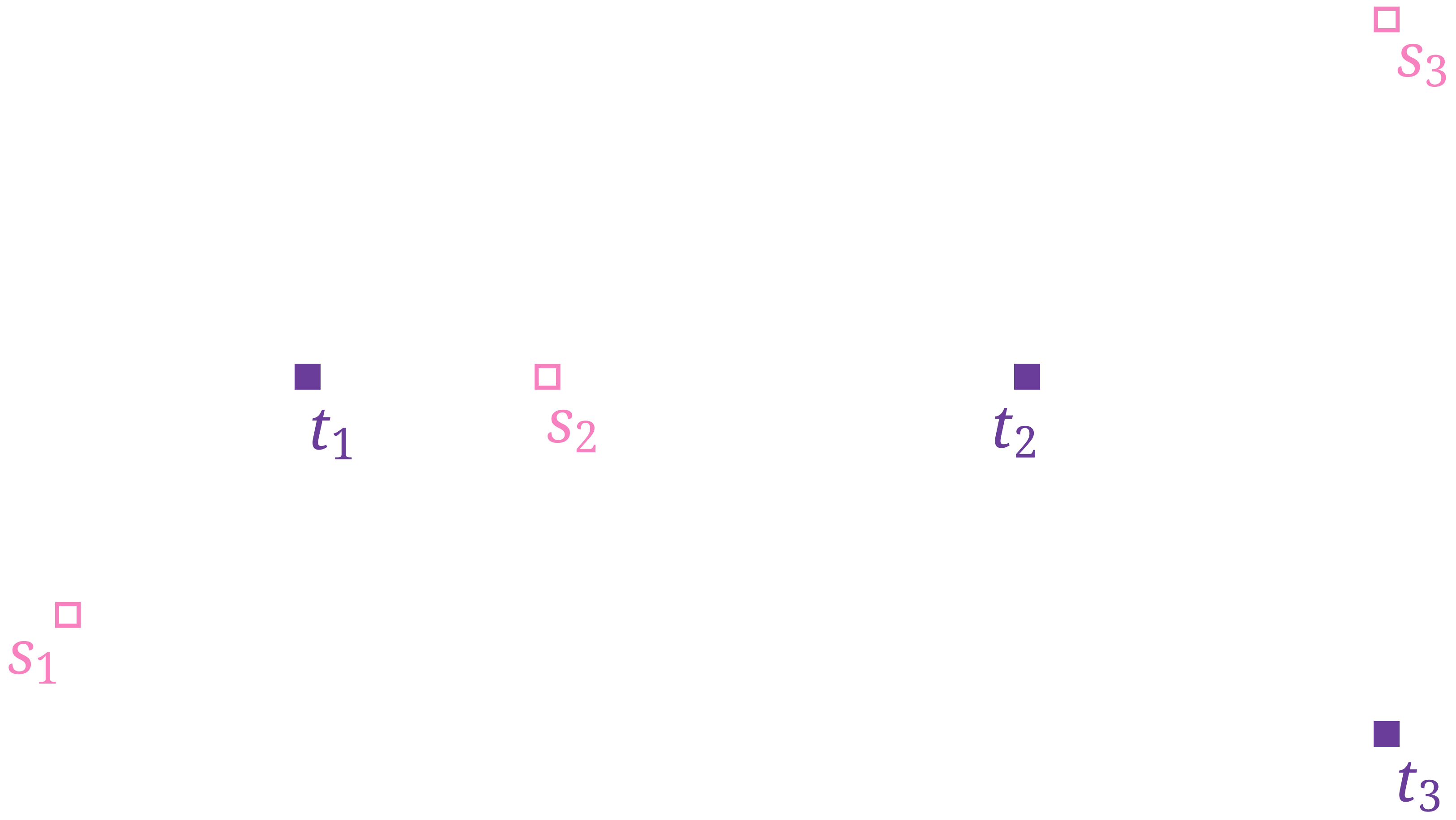
Primal–Dual with Synchronized Increases

PrimalDualSteinerForest(G, c, R)

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

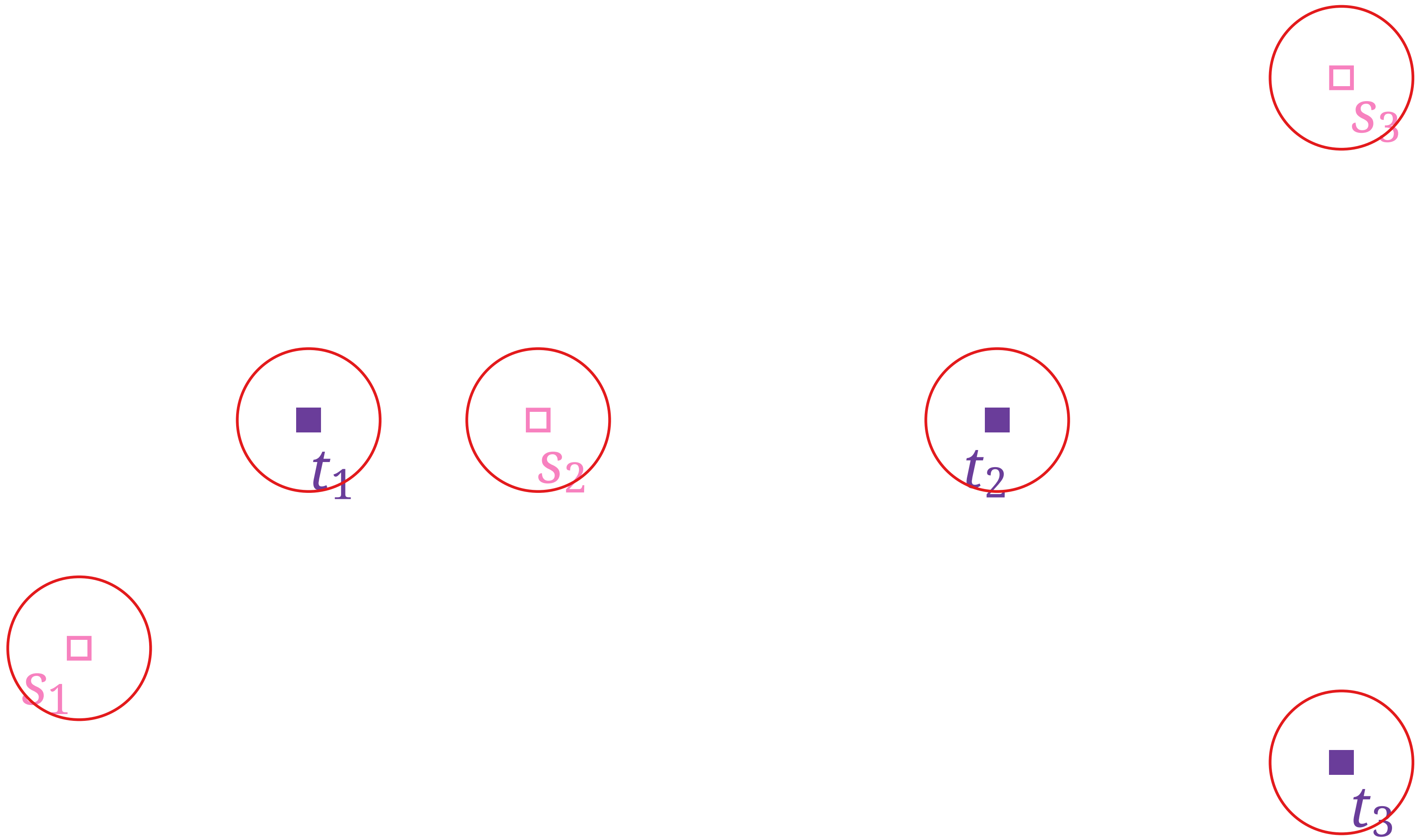
Illustration

$G = K_6$ with Euclidean edge costs



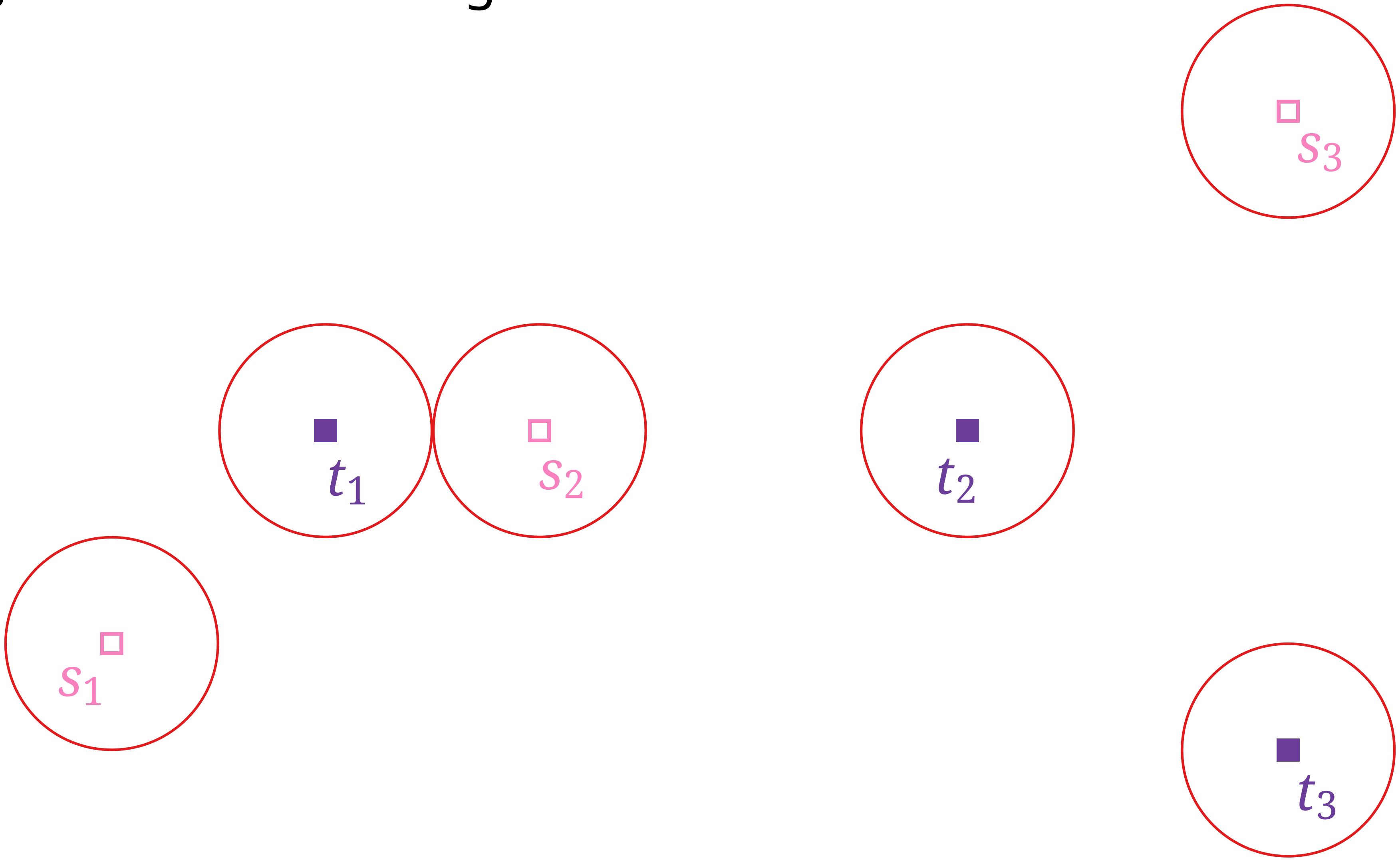
Illustration

$G = K_6$ with Euclidean edge costs



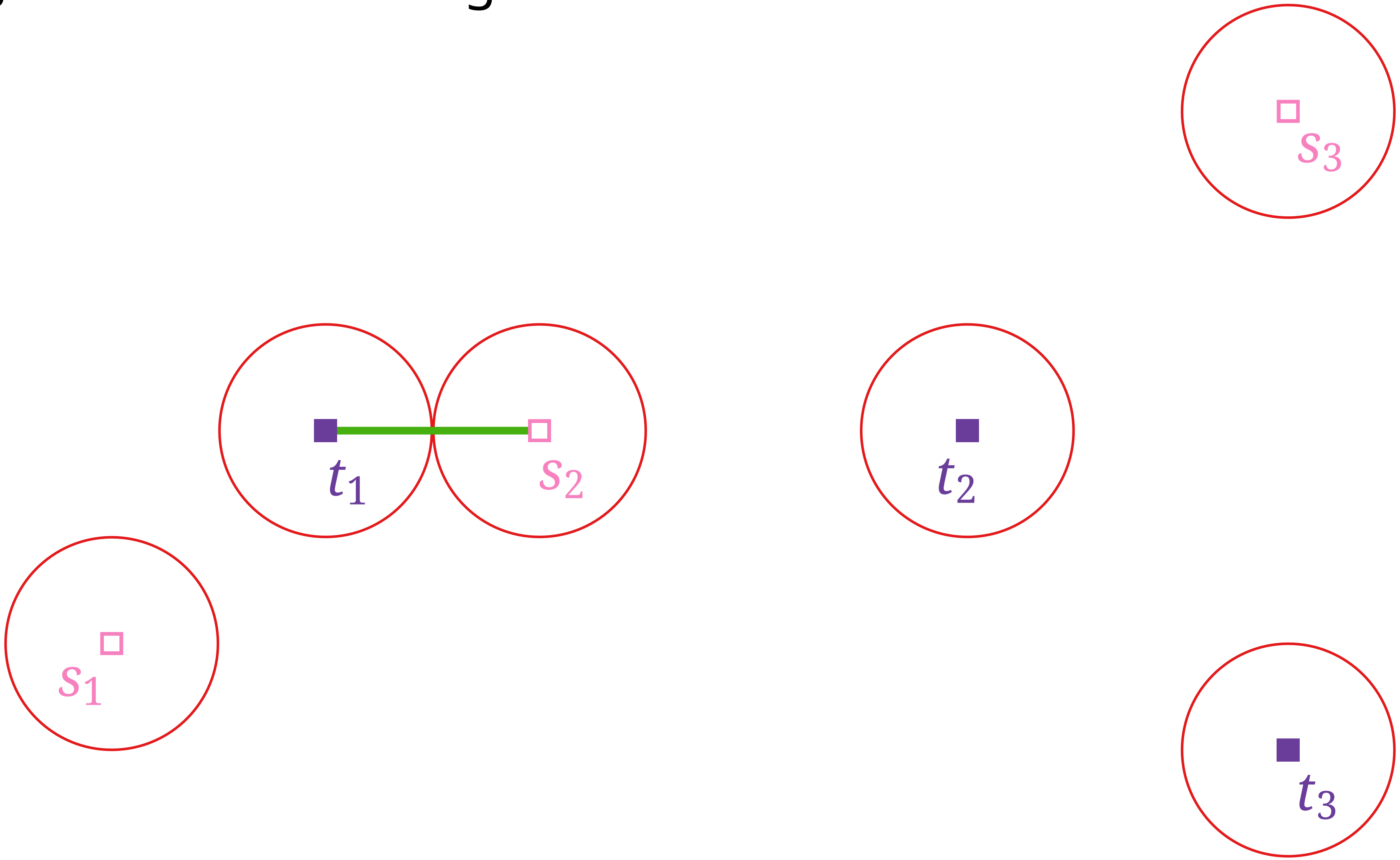
Illustration

$G = K_6$ with Euclidean edge costs



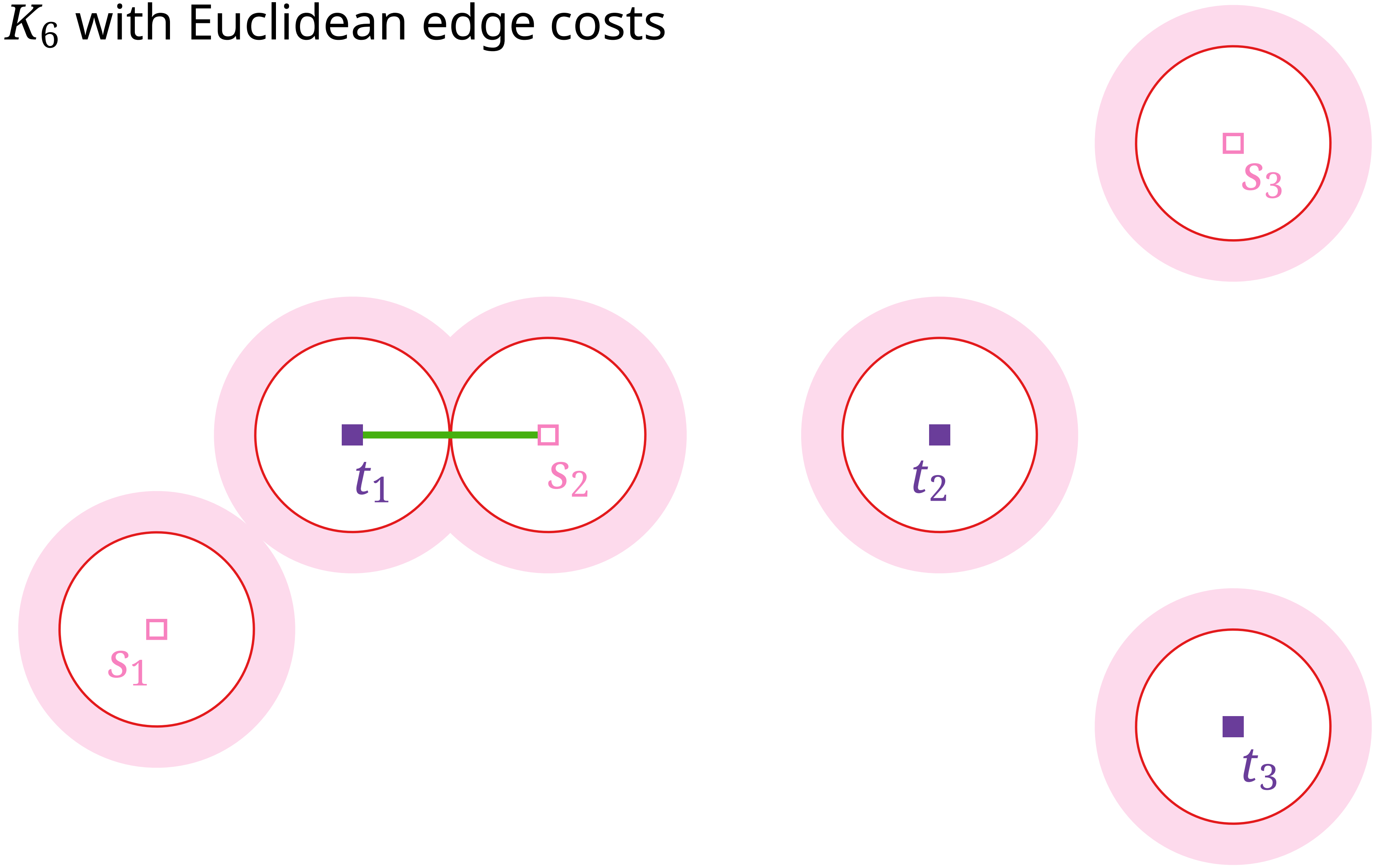
Illustration

$G = K_6$ with Euclidean edge costs



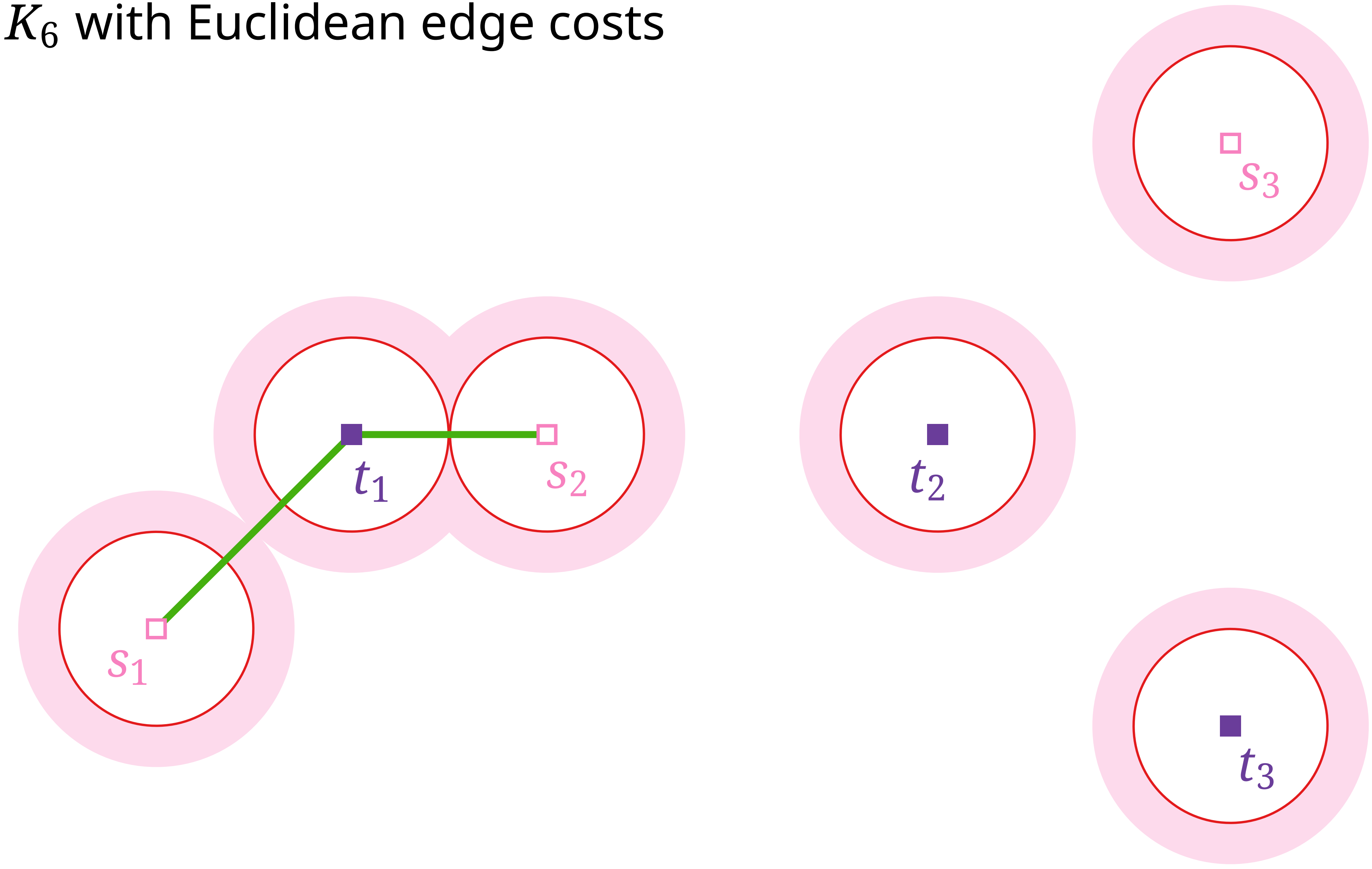
Illustration

$G = K_6$ with Euclidean edge costs



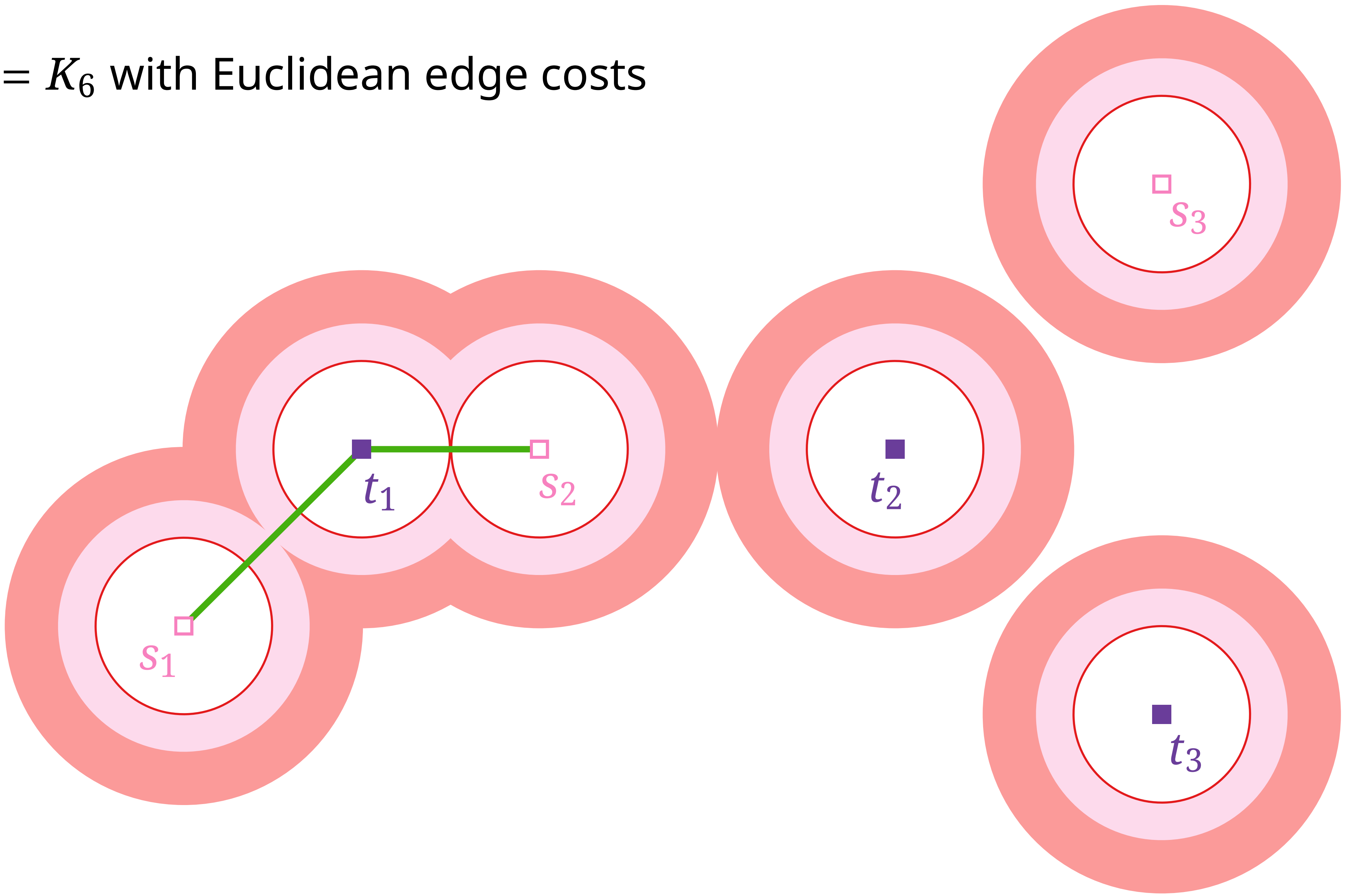
Illustration

$G = K_6$ with Euclidean edge costs



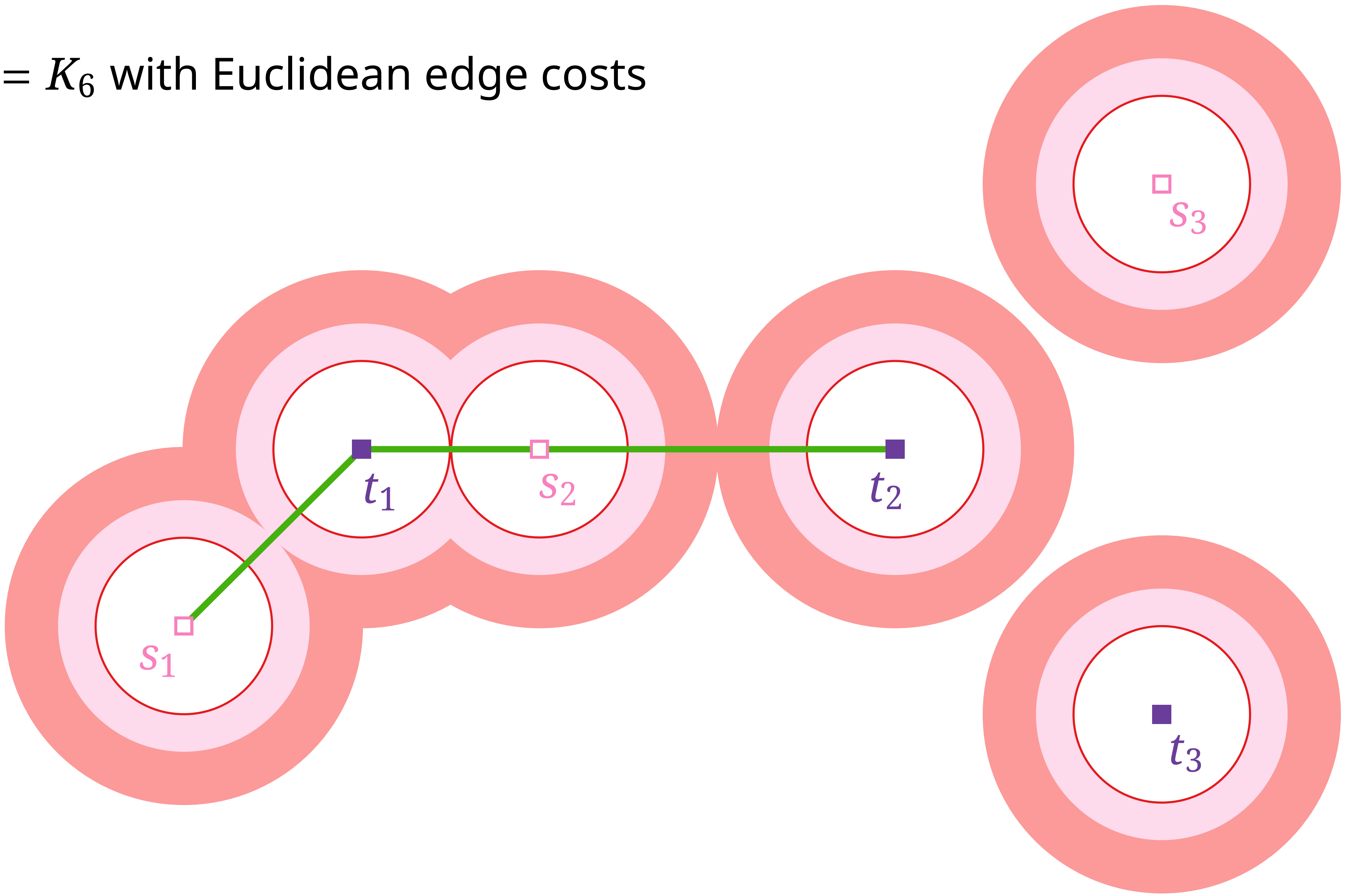
Illustration

$G = K_6$ with Euclidean edge costs



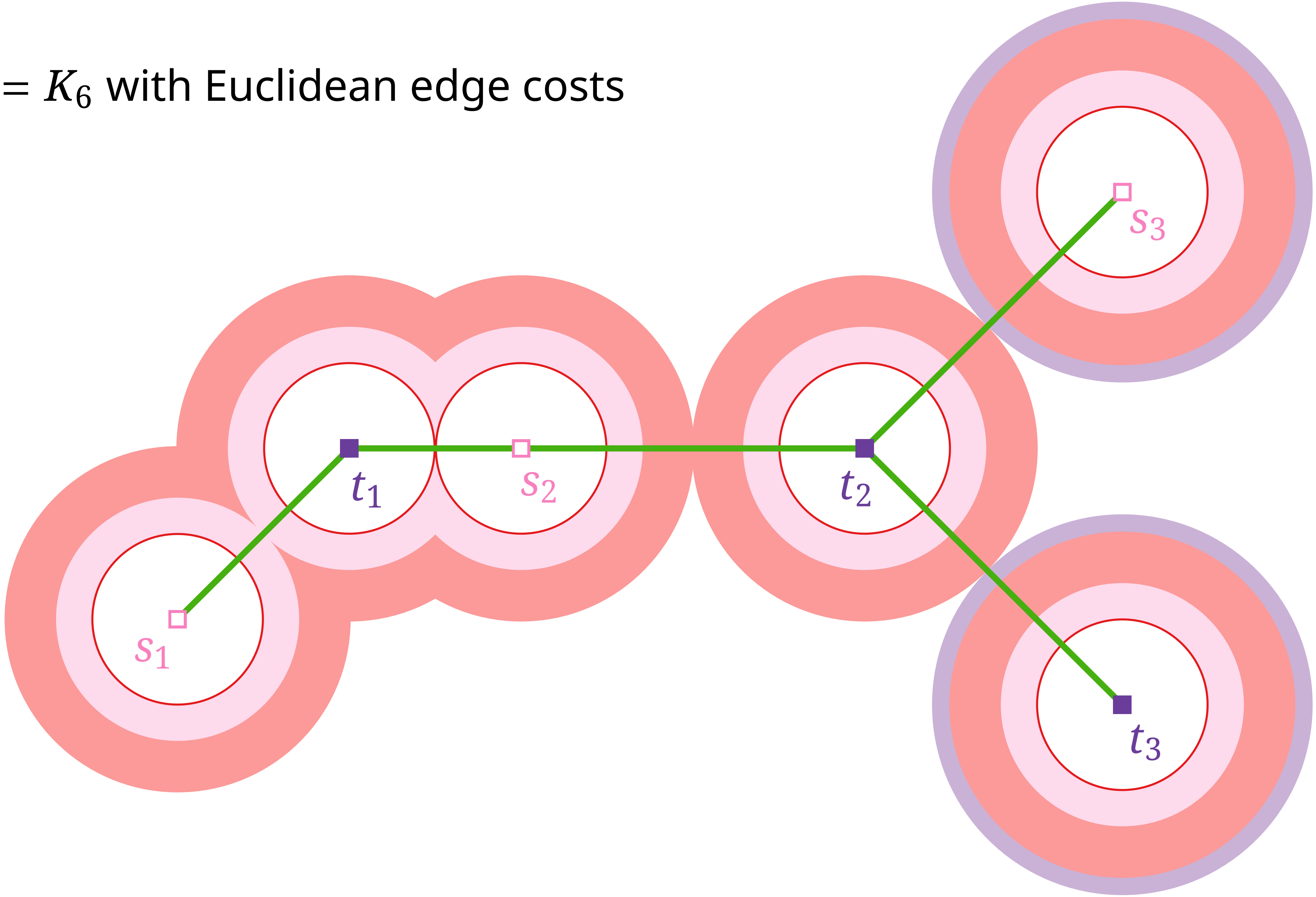
Illustration

$G = K_6$ with Euclidean edge costs



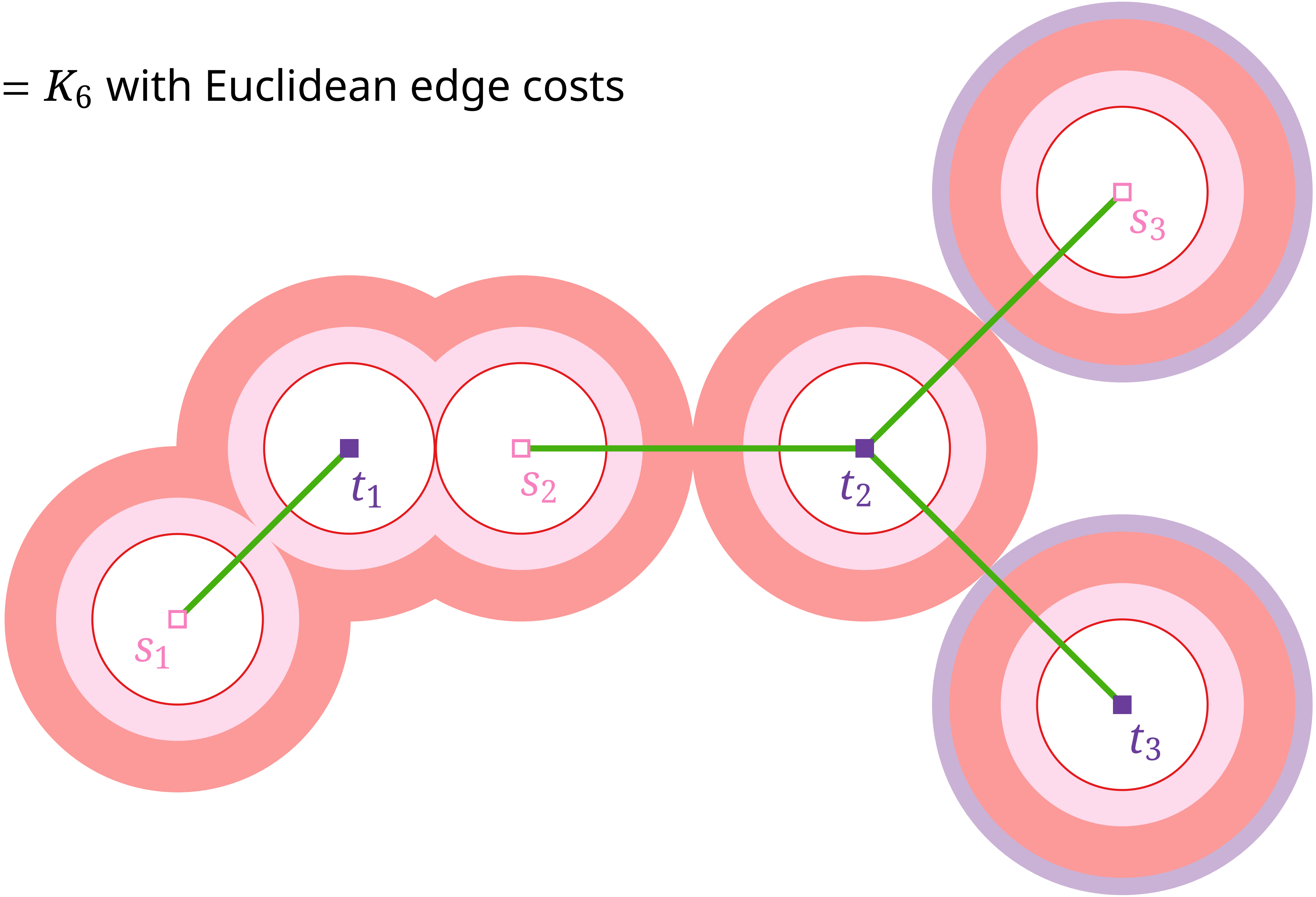
Illustration

$G = K_6$ with Euclidean edge costs



Illustration

$G = K_6$ with Euclidean edge costs



Structure Lemma

Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

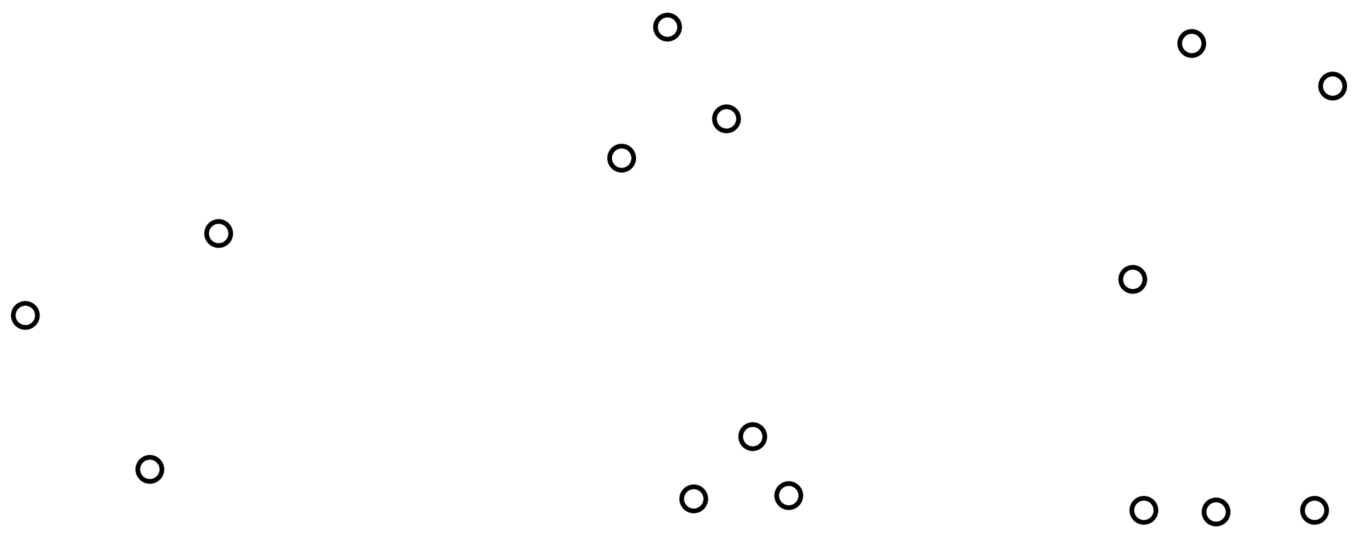
Proof. First the intuition...

Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition…

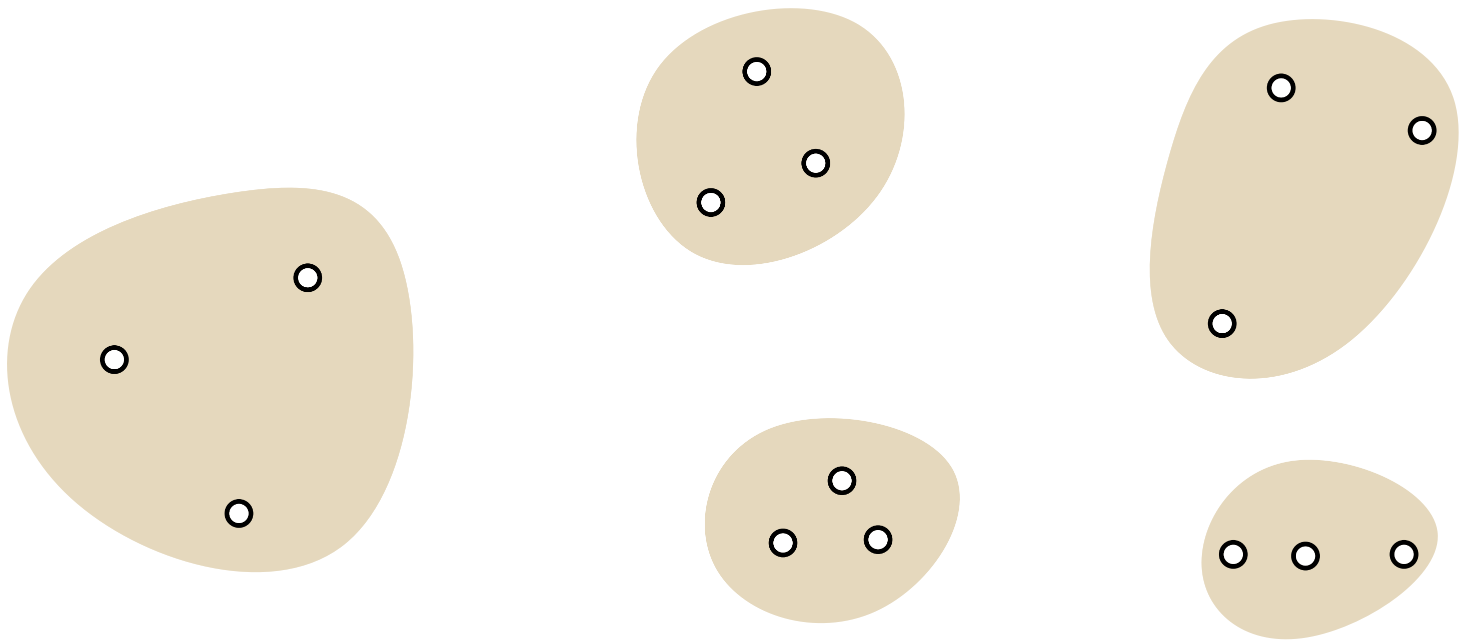


Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition...



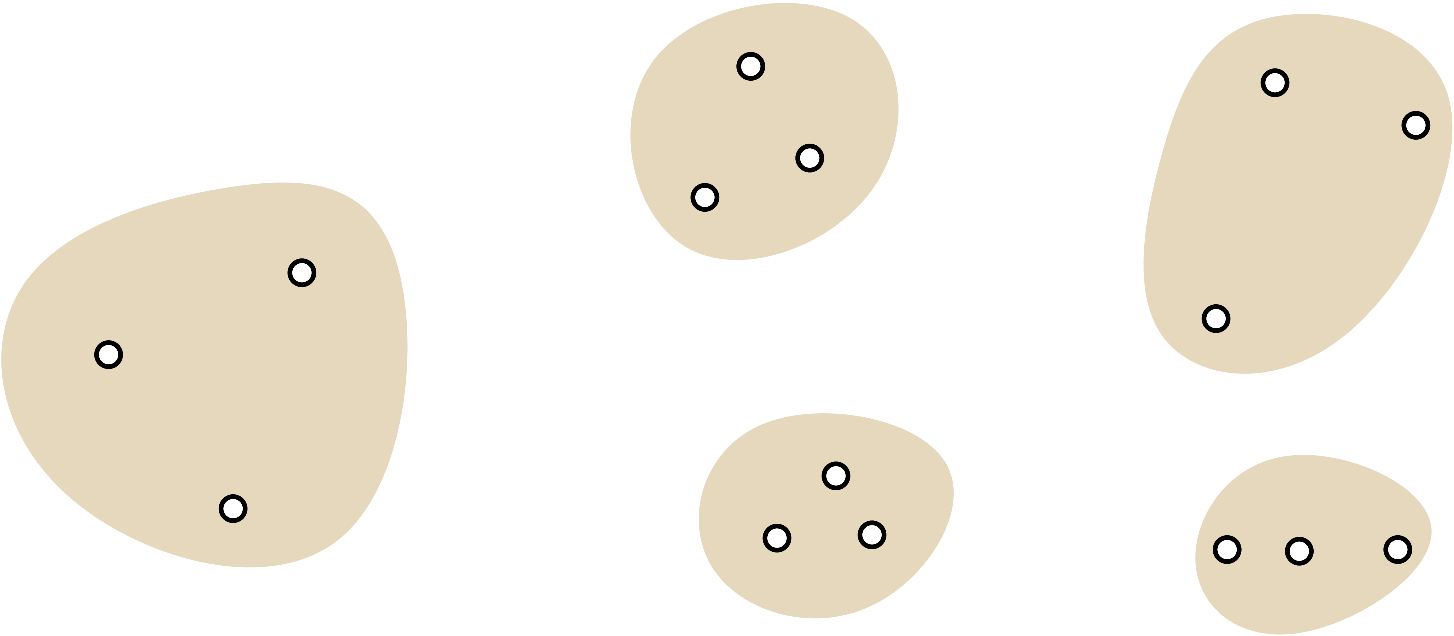
Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition…

$F' \cap C$

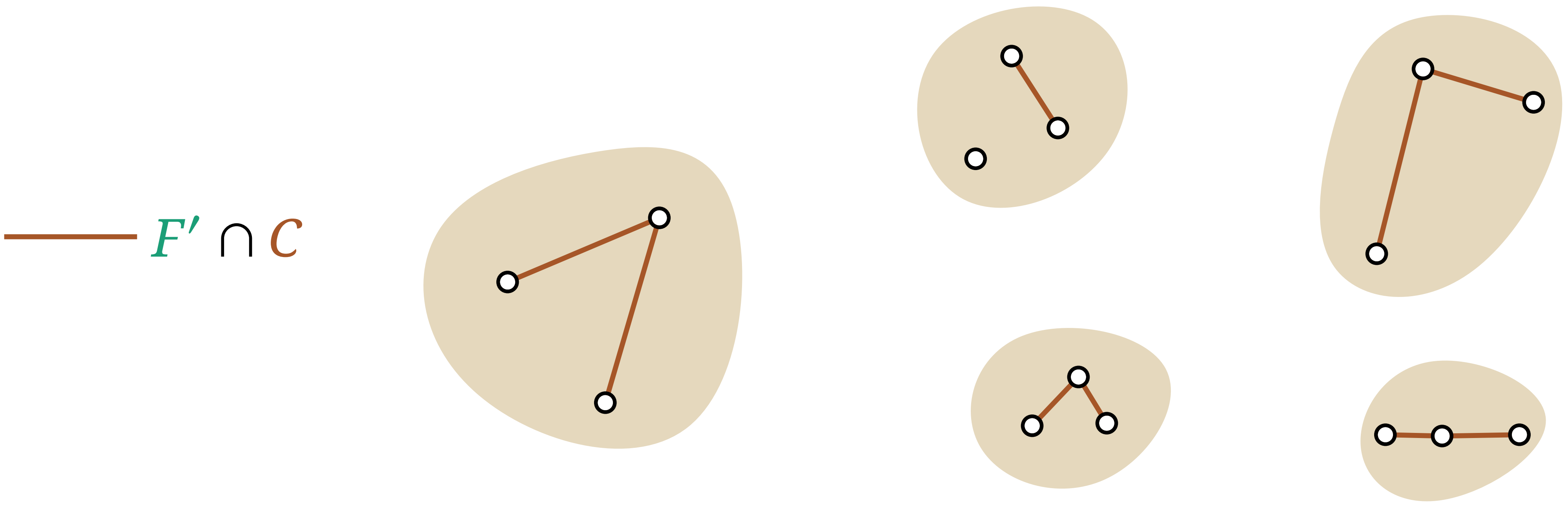


Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition...

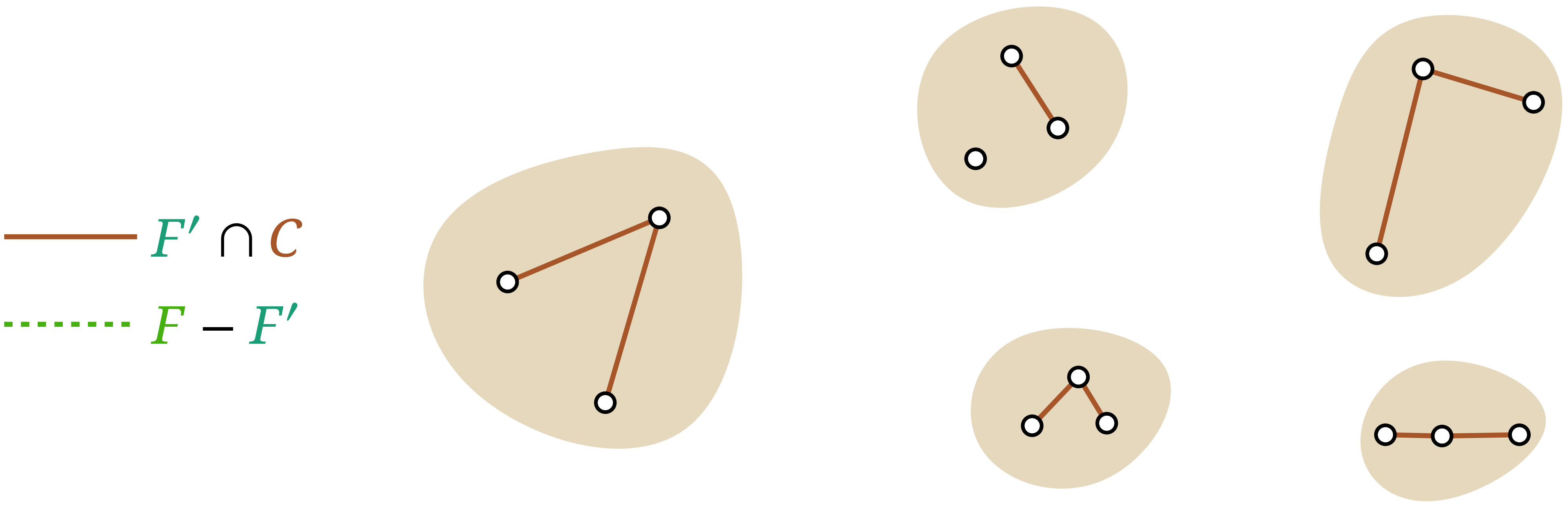


Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition...

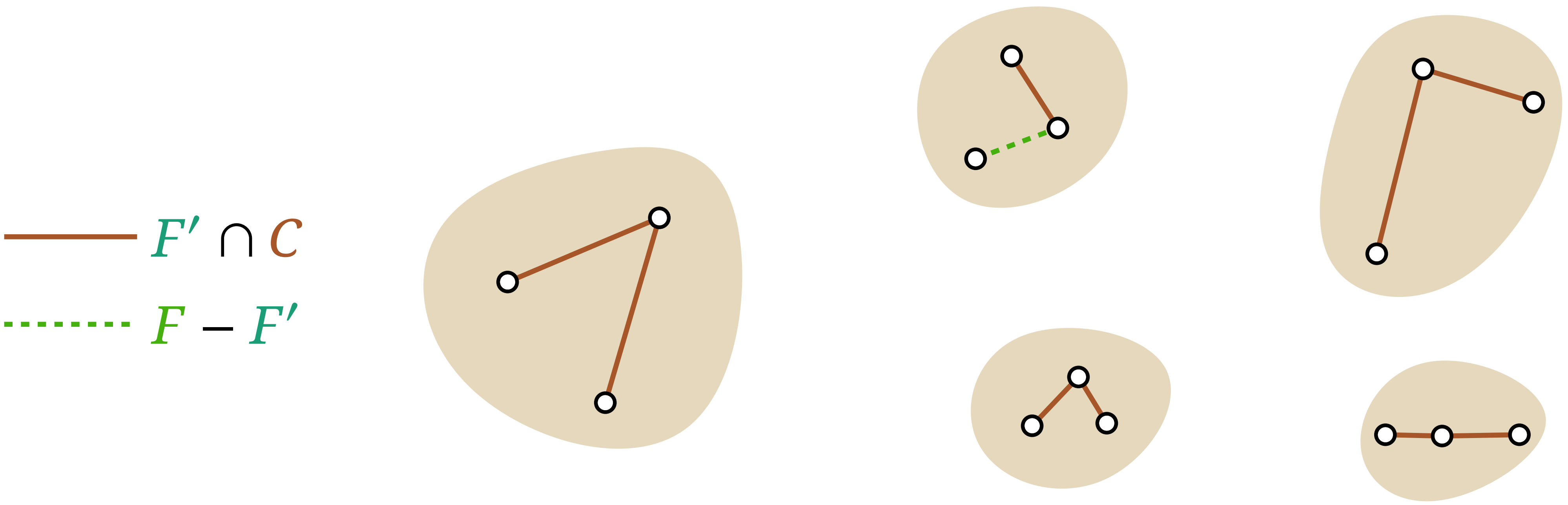


Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition...

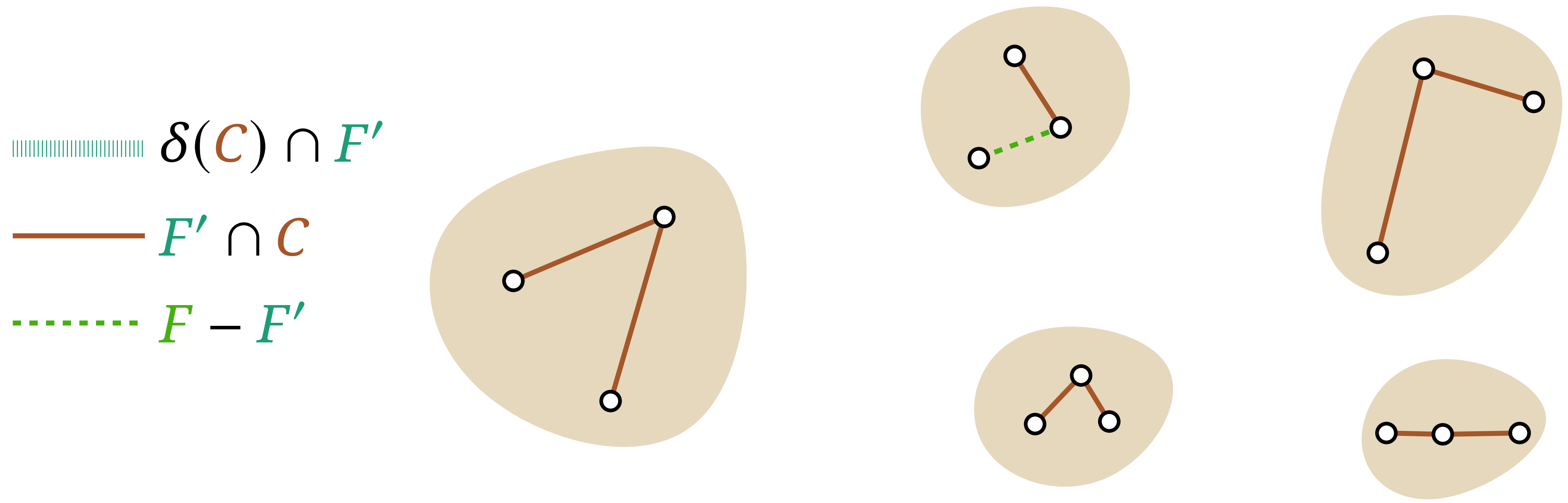


Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition…

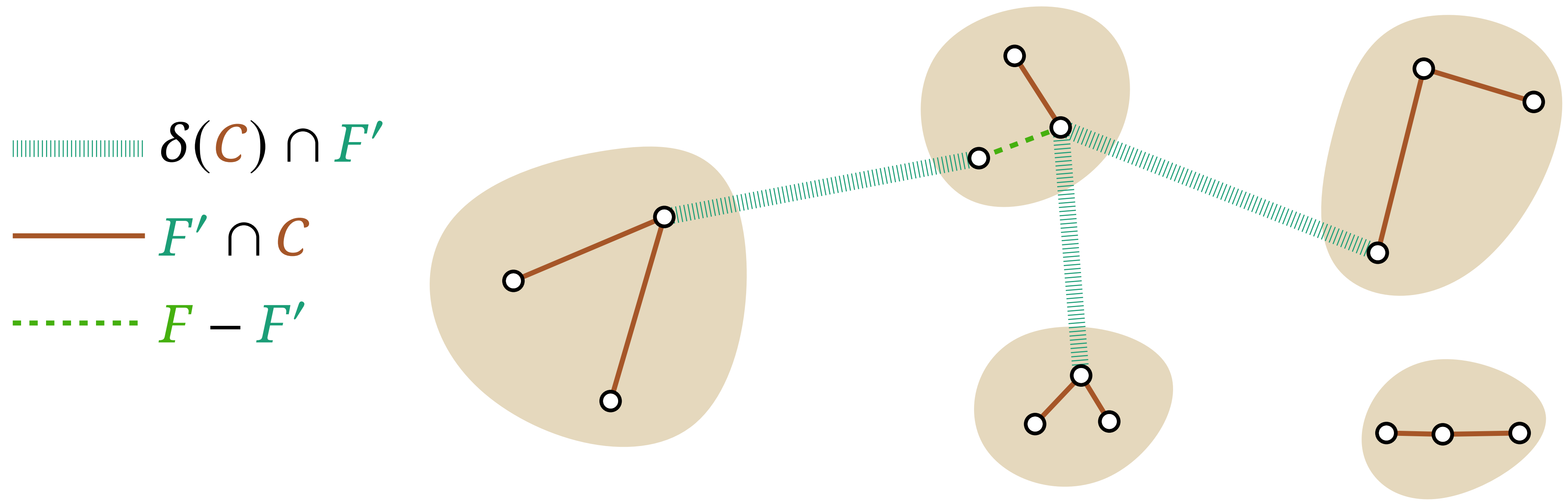


Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition…



Structure Lemma

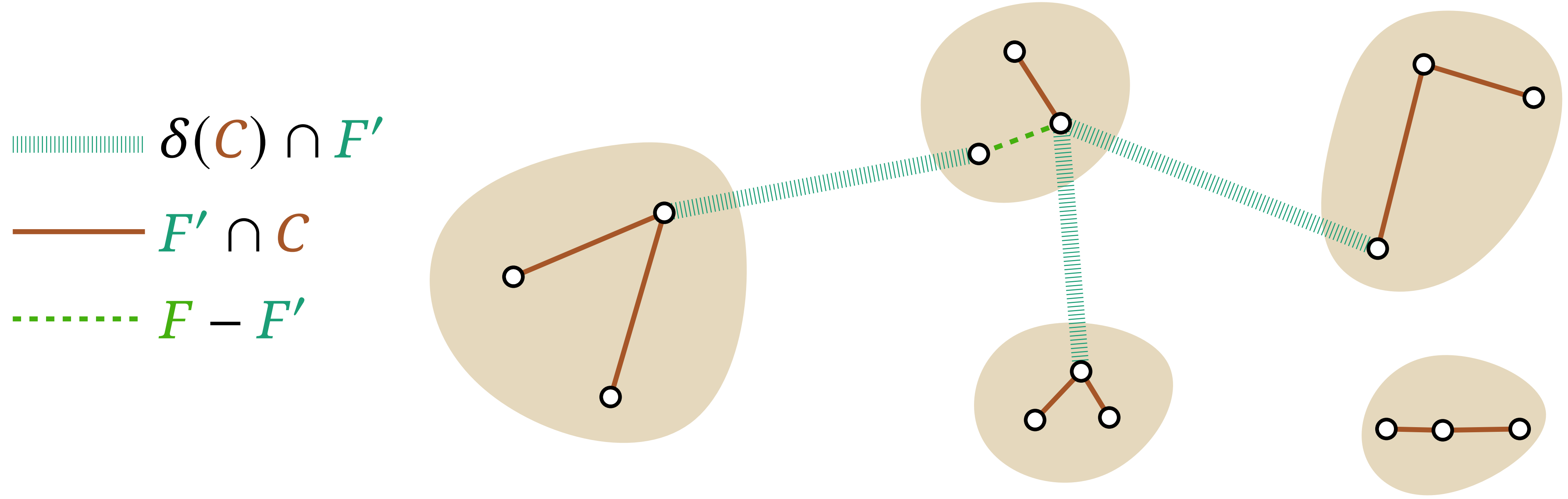
Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition…

Every connected component C of F is a forest in F'.

$\leadsto$ average degree $\leq$



Structure Lemma

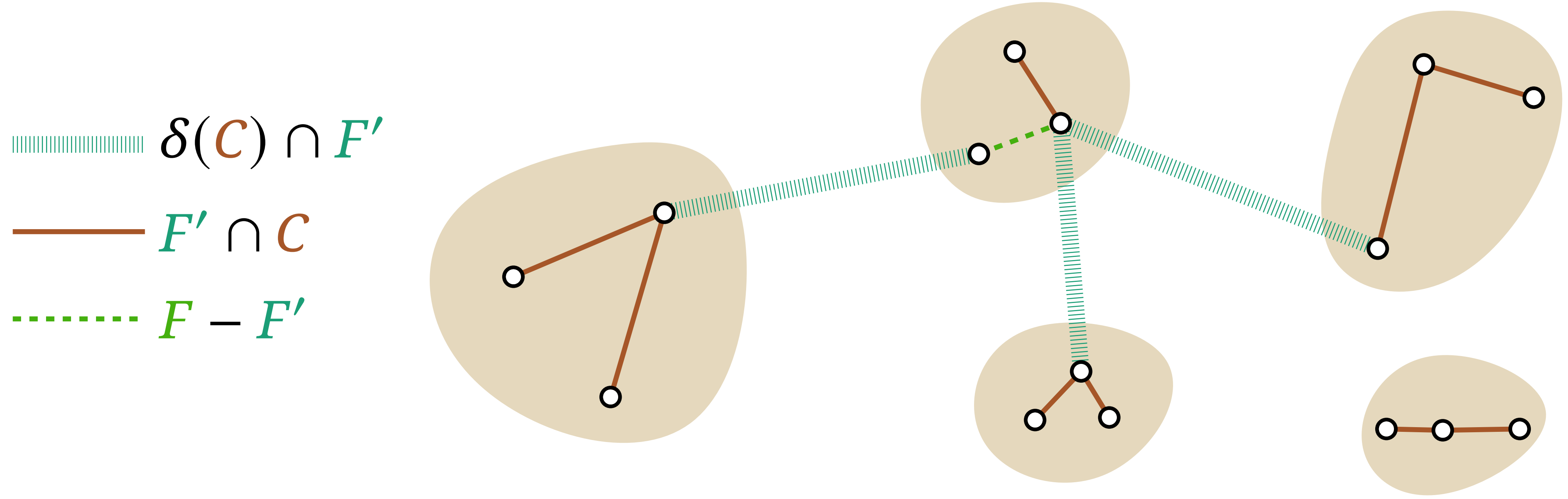
Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

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

Proof. First the intuition...

Every connected component C of F is a forest in F'.

$\rightsquigarrow$ average degree ≤ 2



Structure Lemma

Lemma. For the set $\mathcal{A}$ in any iteration of the algorithm:

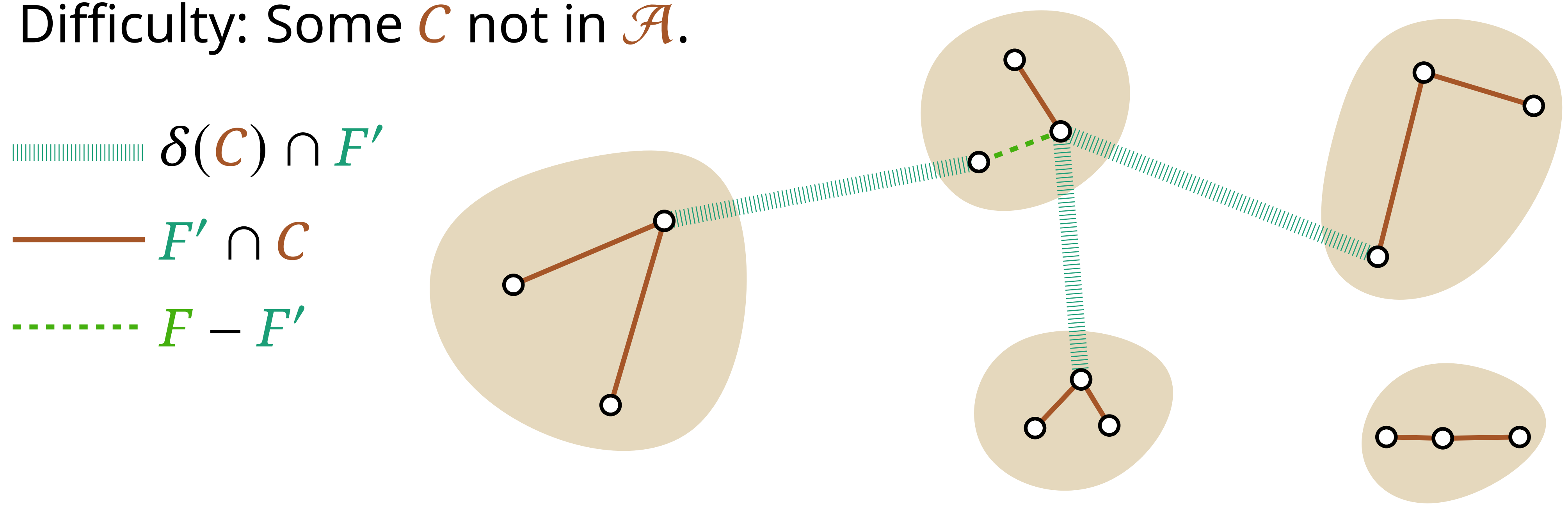
$$\sum_{C \in \mathcal{A}} |\delta(C) \cap F'| \leq 2|\mathcal{A}|\,.$$

Proof. First the intuition...

Every connected component C of F is a forest in F'.

$\rightsquigarrow$ average degree ≤ 2

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

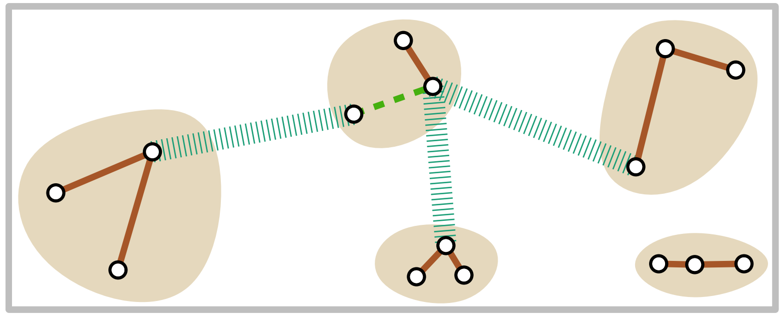


Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.



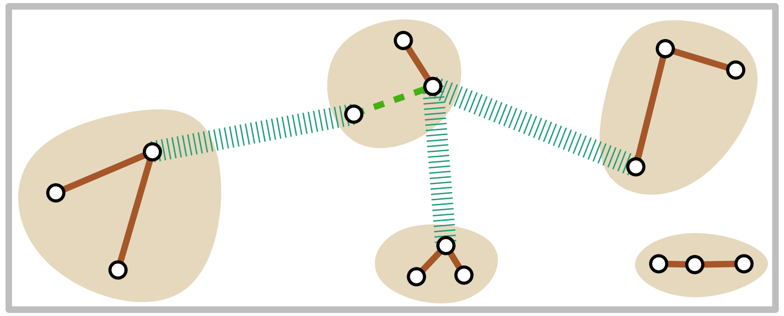
Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).



Proof of the Structure Lemma

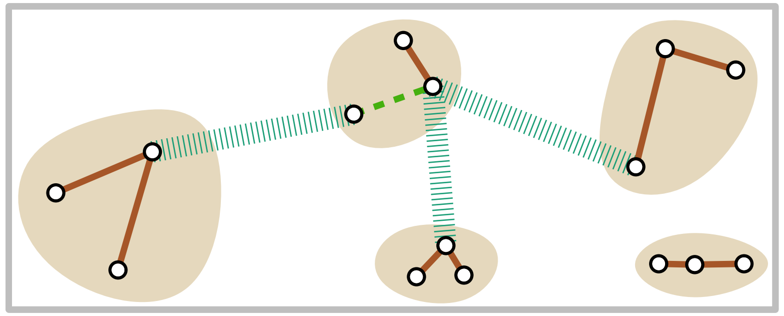
Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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



Proof of the Structure Lemma

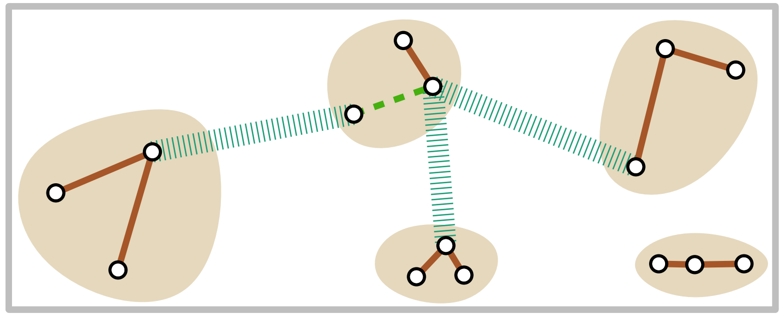
Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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



Proof of the Structure Lemma

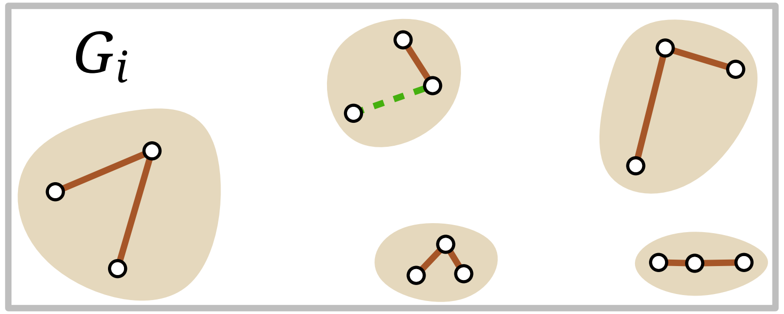
Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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



Proof of the Structure Lemma

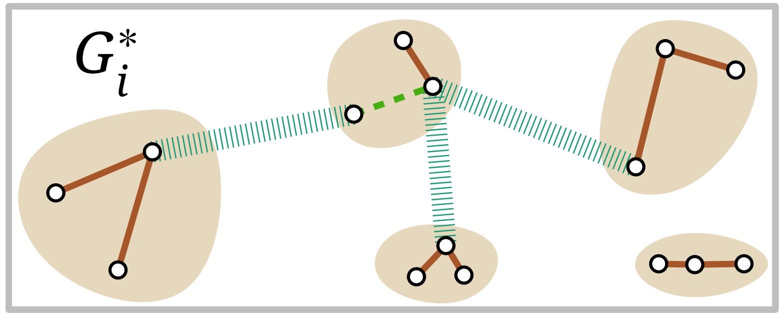
Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was 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 the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

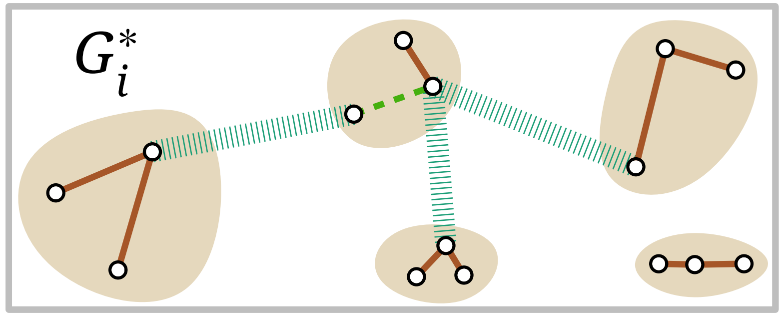
$$\sum_{C \in \mathcal{A}} |\delta(C) \cap F'| \leq 2|\mathcal{A}|\,.$$

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was 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 every component C of G_i in G_i^* to a single vertex $\leadsto G_i'$.



Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

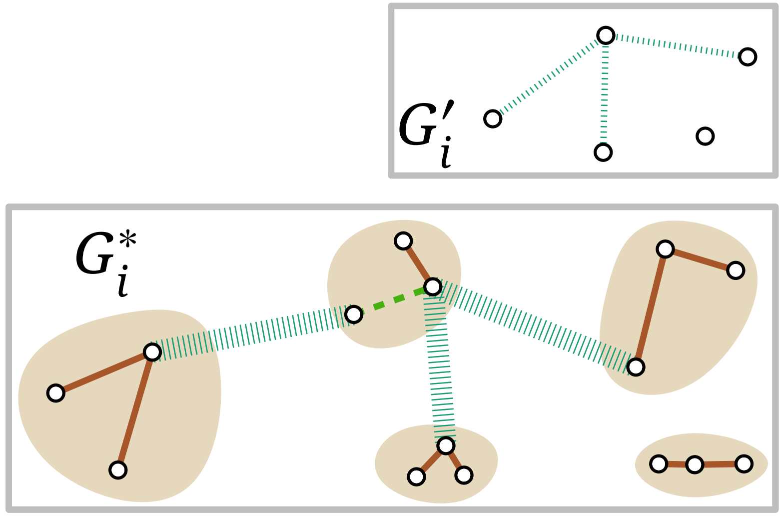
$$\sum_{C \in \mathcal{A}} |\delta(C) \cap F'| \leq 2|\mathcal{A}|\,.$$

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was 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 every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.



Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

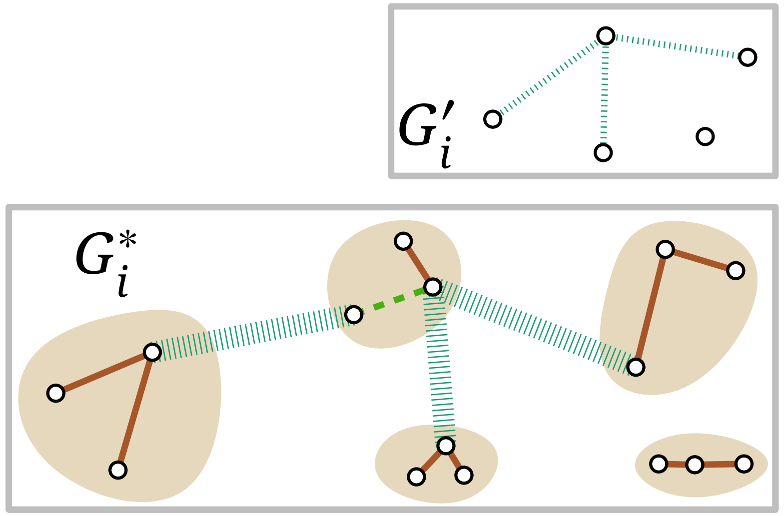
Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was 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 every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G'_i$.

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



Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

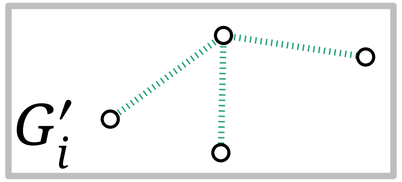
Proof.

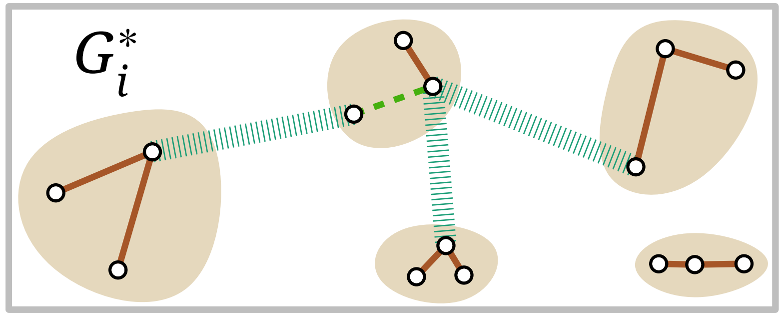
For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

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

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





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

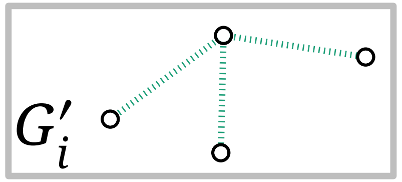
For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

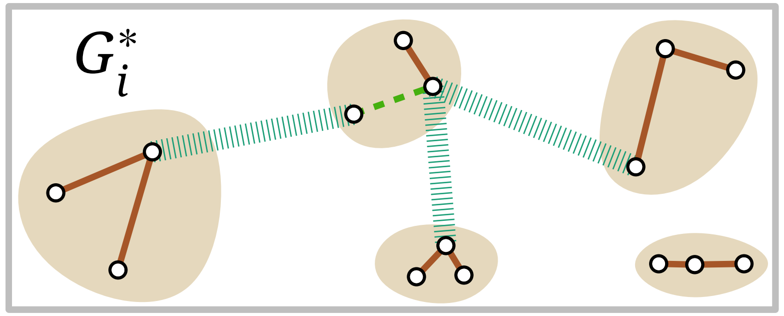
Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

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

Claim. G_i' is a forest.

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





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

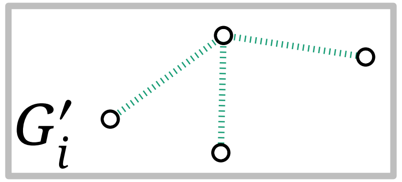
Let $F_i = \{e_1, \ldots, e_i\}$, $G_i = (V, F_i)$, and $G_i^* = (V, F_i \cup F')$.

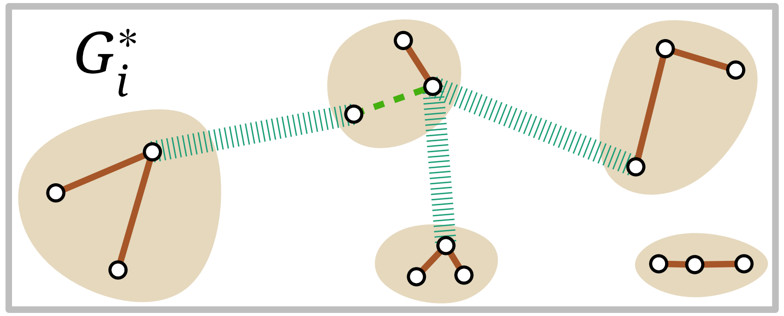
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

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

Claim. G_i' is a forest.

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





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

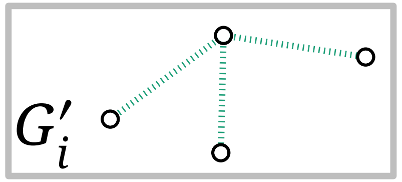
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

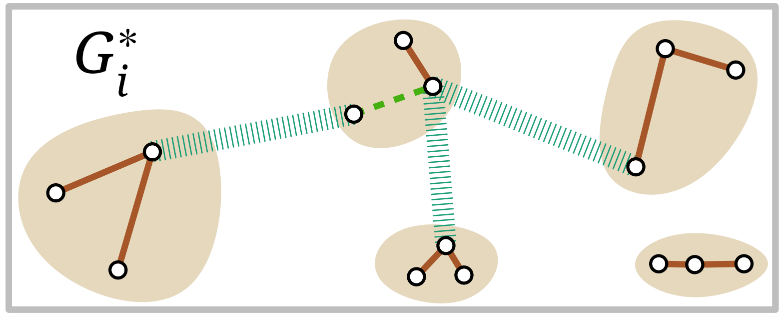
(Ignore components C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

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

$= 2|E(G_i')|$





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

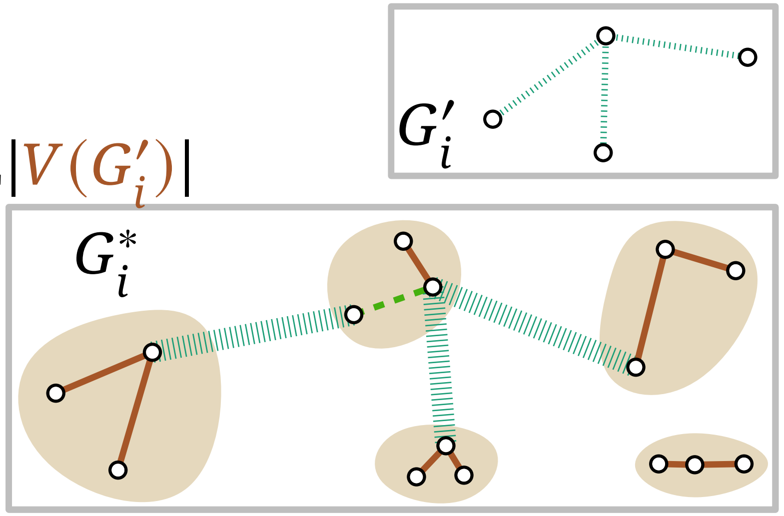
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

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

Claim. G_i' is a forest.

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

$$= 2|E(G_i')| \quad \le 2|V(G_i')|$$



Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

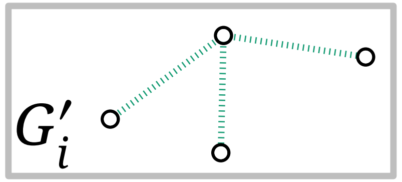
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

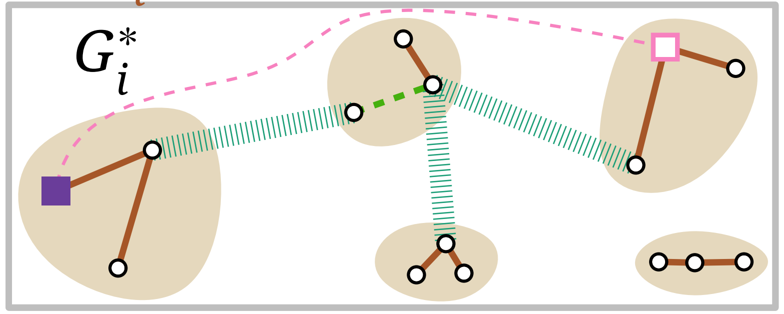
Claim. G_i' is a forest.

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

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

$= 2|E(G_i')| \quad \le 2|V(G_i')|$





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

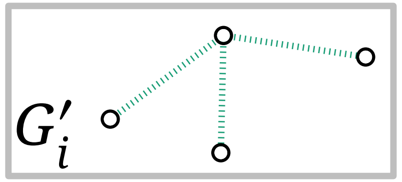
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

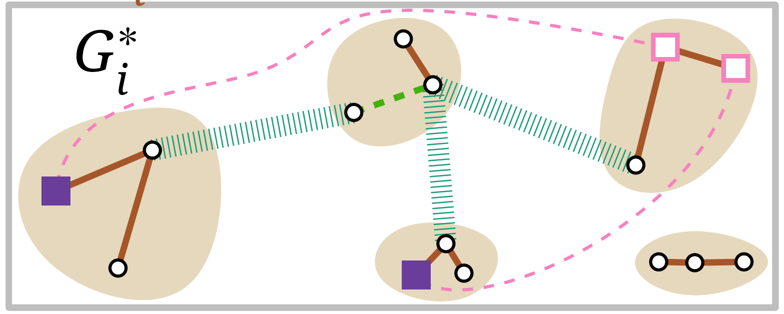
(Ignore components C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

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

$= 2|E(G_i')| \quad \leq 2|V(G_i')|$





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

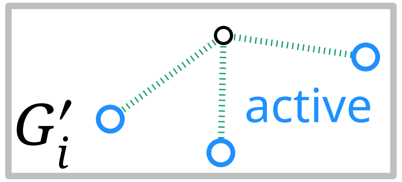
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

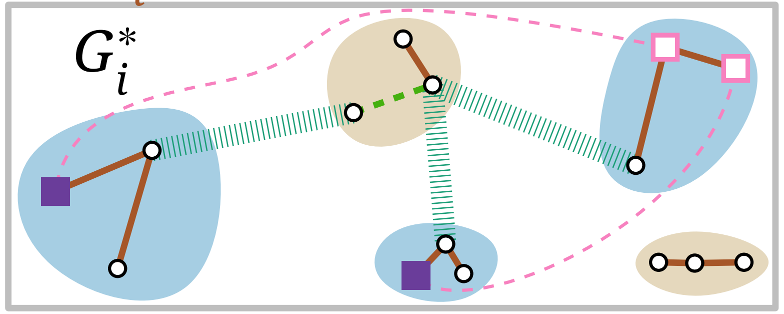
Claim. G_i' is a forest.

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

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

$$= 2|E(G_i')| \quad \leq 2|V(G_i')|$$





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

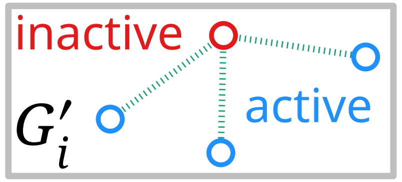
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

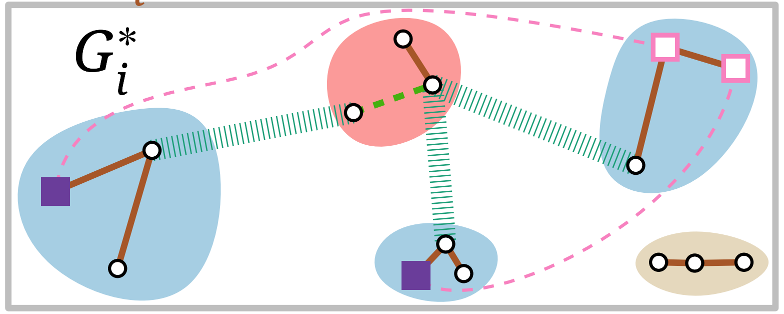
(Ignore components C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.

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

$= 2|E(G_i')| \quad \leq 2|V(G_i')|$





Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

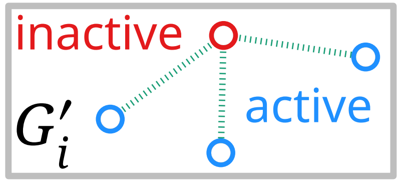
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

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

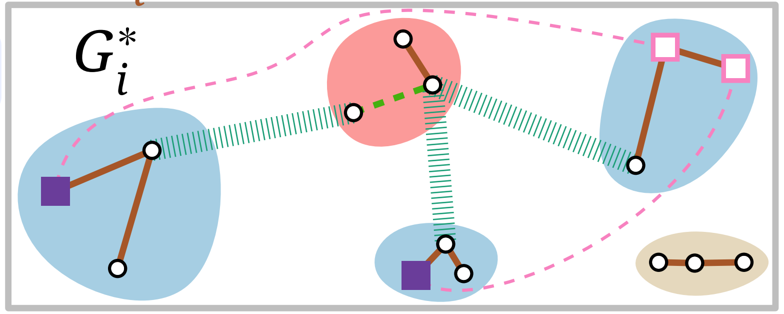
Claim. G_i' is a forest.

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

$= 2|E(G_i')| \quad \leq 2|V(G_i')|$



Claim. Inactive vertices have degree ≥ 2.



Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

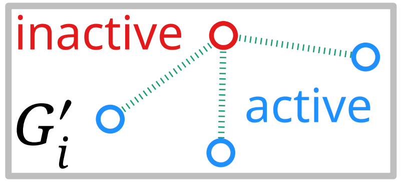
For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

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

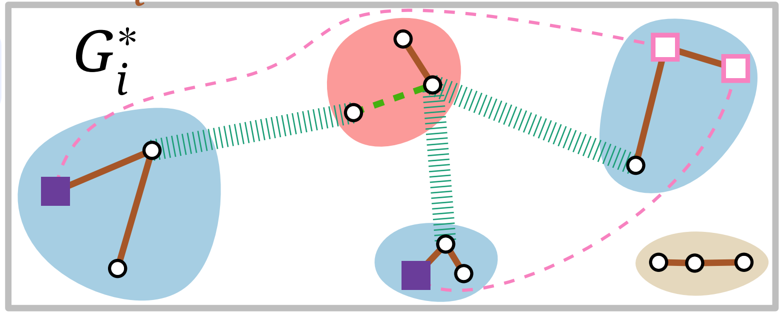
(Ignore components C with $\delta(C) \cap F' = \emptyset$.)

Claim. G_i' is a forest.



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

$$= 2|E(G_i')| \quad \leq 2|V(G_i')|$$



Claim. Inactive vertices have degree ≥ 2.

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

Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

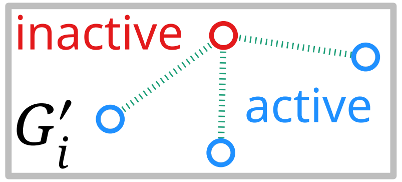
For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

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

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

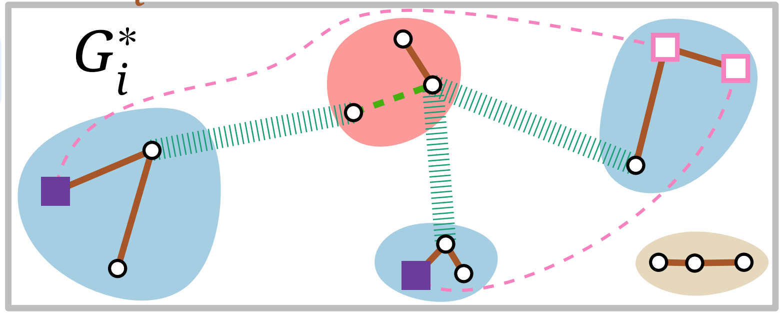
Claim. G_i' is a forest.



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

$= 2|E(G_i')| \quad \leq 2|V(G_i')|$

Claim. Inactive vertices have degree ≥ 2.



$\Rightarrow \sum_{v \text{ active}} \deg_{G'}(v) \leq$
$2 \cdot |V(G')| - 2 \cdot \#(\text{inactive}) =$

Proof of the Structure Lemma

Lemma. For the set C in any iteration of the algorithm:

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

Proof.

For $i = 1, \ldots, \ell$, consider i-th iteration (when e_i was added to F).

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

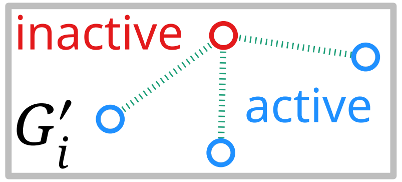
Contract every component C of G_i in G_i^* to a single vertex $\rightsquigarrow G_i'$.

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

Claim. G_i' is a forest.

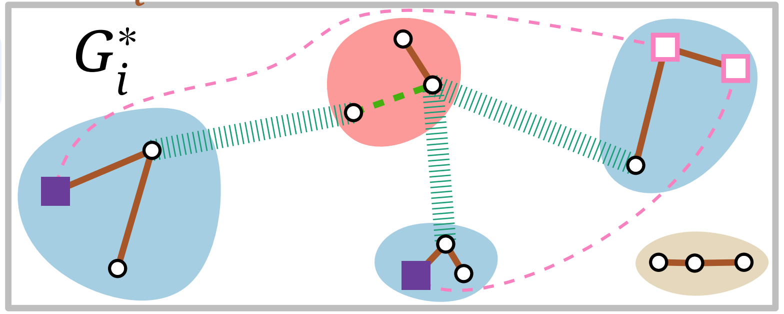
Note: $\sum_{C \text{ comp.}} |\delta(C) \cap F'| = \sum_{v \in V(G_i')} \deg_{G_i'}(v)$

$= 2|E(G_i')| \quad \leq 2|V(G_i')|$



Claim. Inactive vertices have degree ≥ 2.

$\Rightarrow \sum_{v \text{ active}} \deg_{G'}(v) \leq$
$2 \cdot |V(G')| - 2 \cdot \#(\text{inactive}) = 2|\mathcal{A}|$. $\square$



Analysis

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 mentioned 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 mentioned 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

Analysis

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

Proof.

As mentioned 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 mentioned 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

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

Analysis

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

Proof.

As mentioned 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

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

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 every 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 every S.

Assume that $(*)$ holds at the start of the current 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 every S.

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

In the current iteration, we increase y_C for every $C \in \mathcal{A}$ 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 every S.

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

In the current iteration, we increase y_C for every $C \in \mathcal{A}$ 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 every S.

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

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

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

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 every S.

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

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

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

Analysis

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

Proof.

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

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

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

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

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

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 every S.

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

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

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

Structure lemma $\Longrightarrow$

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 every S.

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

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

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

Structure lemma $\Rightarrow (*)$ also holds after the current iteration.

$\square$

Summary

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

Summary

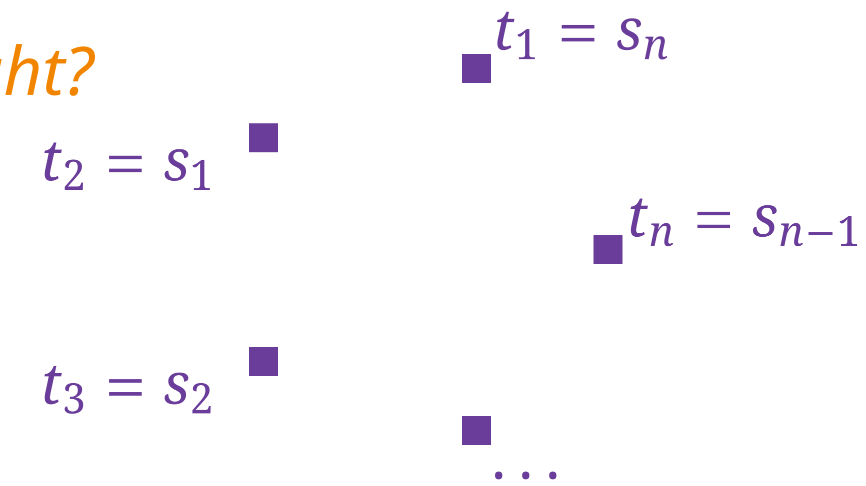
Theorem. The Primal–Dual algorithm with synchronized increases gives a 2-approximation for SteinerForest.

Is our analysis tight?

Summary

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

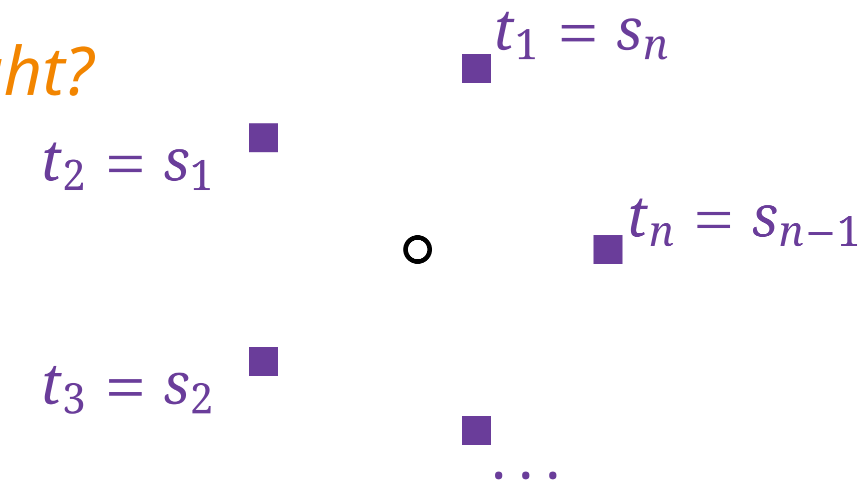
Is our analysis tight?



Summary

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

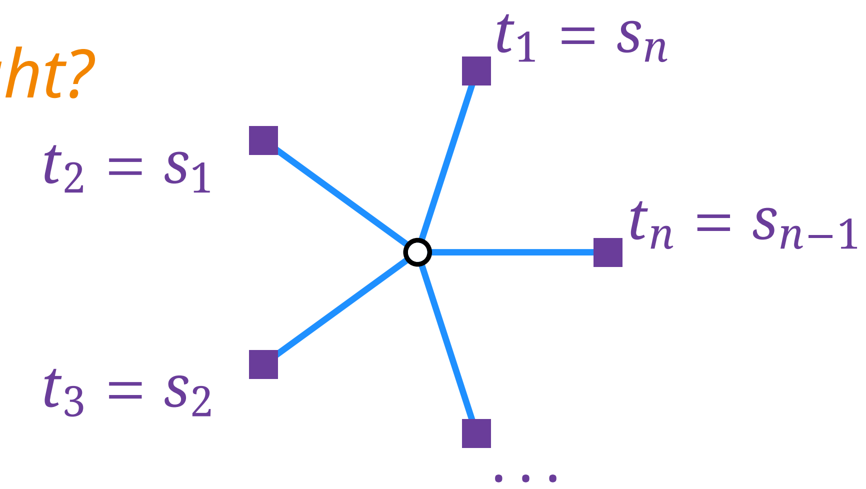
Is our analysis tight?



Summary

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

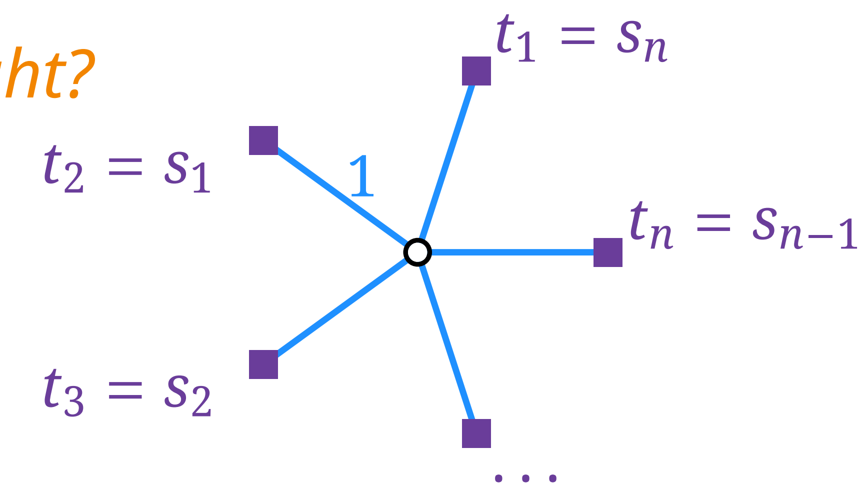
Is our analysis tight?



Summary

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

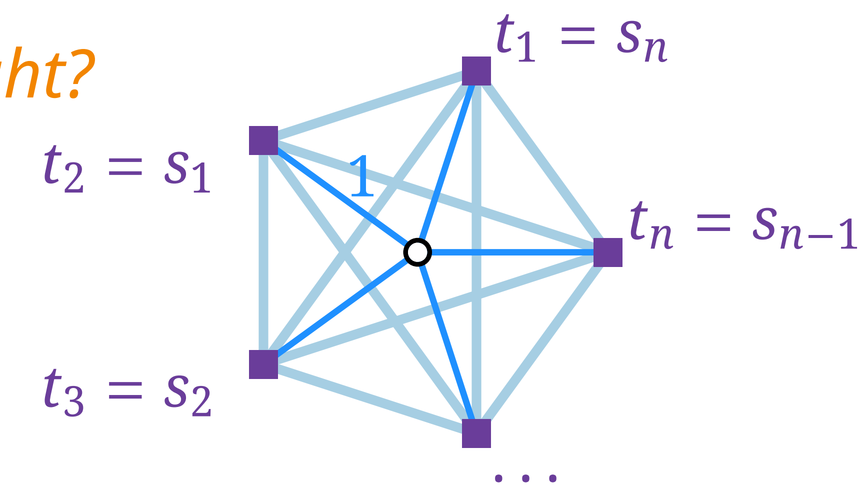
Is our analysis tight?



Summary

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

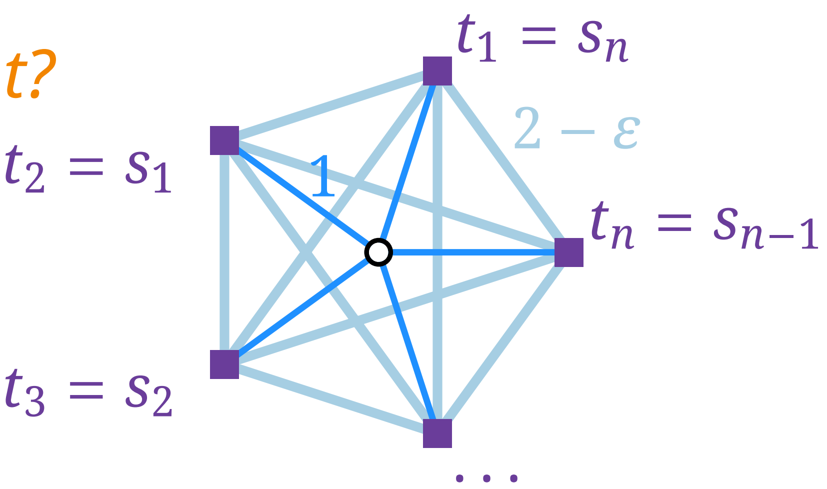
Is our analysis tight?



Summary

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

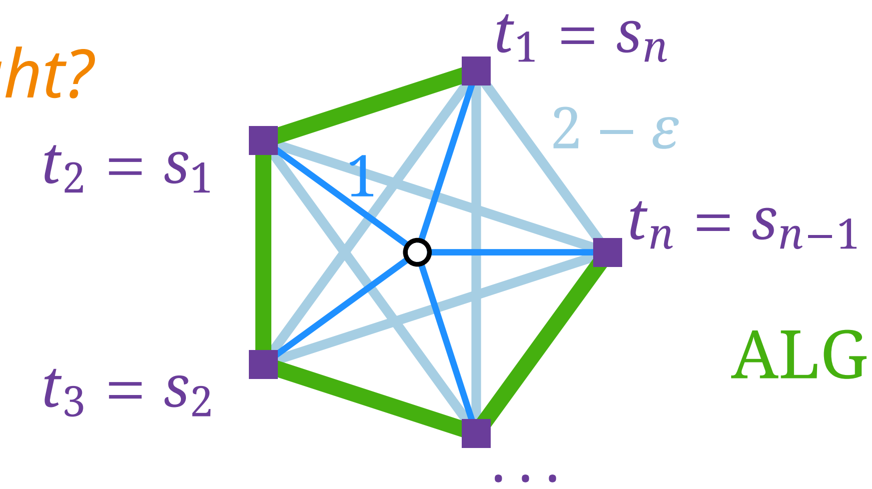
Is our analysis tight?



Summary

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

Is our analysis tight?

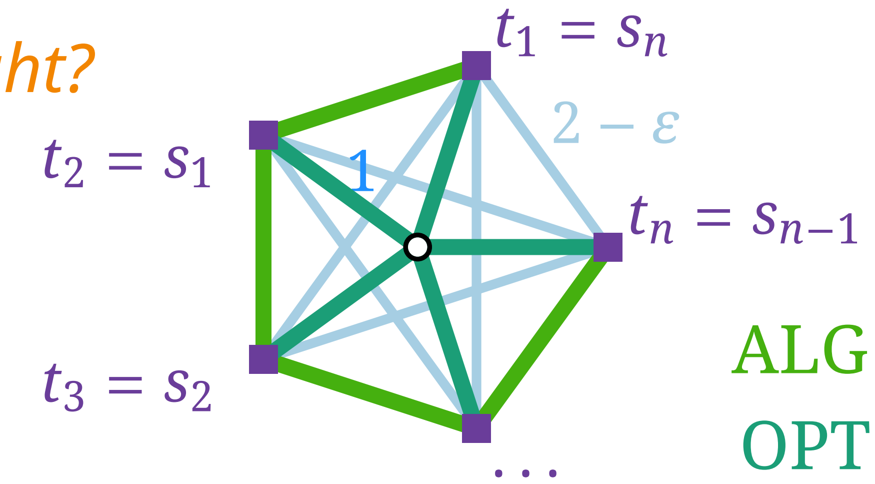


$\mathrm{ALG} = (2 - \varepsilon)(n - 1)$

Summary

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

Is our analysis tight?

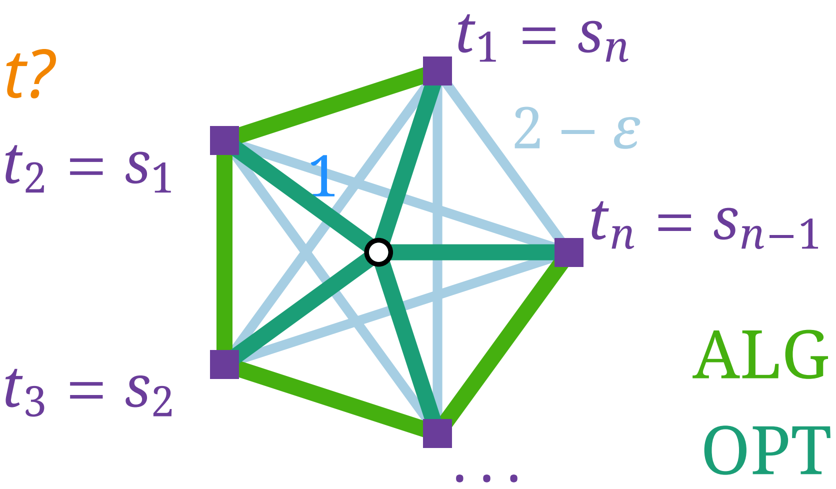


$$\text{ALG} = (2 - \varepsilon)(n - 1)$$
$$\text{OPT} = n$$

Summary

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

Is our analysis tight?



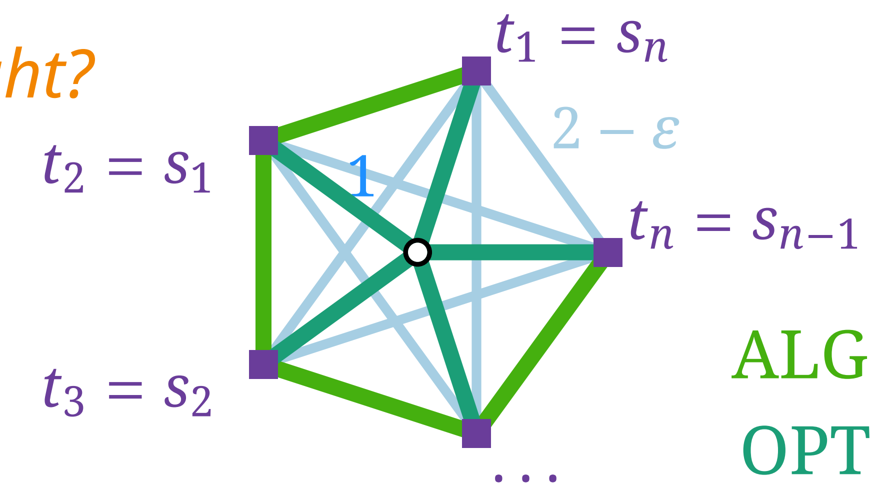
$$\mathrm{ALG} = (2 - \varepsilon)(n - 1)$$
$$\mathrm{OPT} = n$$

Can we do better?

Summary

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

Is our analysis tight?



$\text{ALG} = (2 - \varepsilon)(n - 1)$

$\text{OPT} = n$

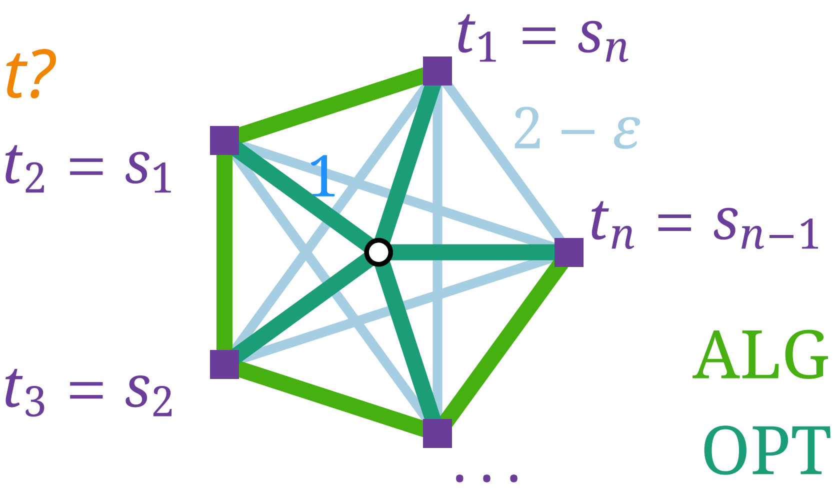
Can we do better?

No better approximation factor is known. :-(

Summary

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

Is our analysis tight?



$\text{ALG} = (2 - \varepsilon)(n - 1)$

$\text{OPT} = n$

Can we do better?

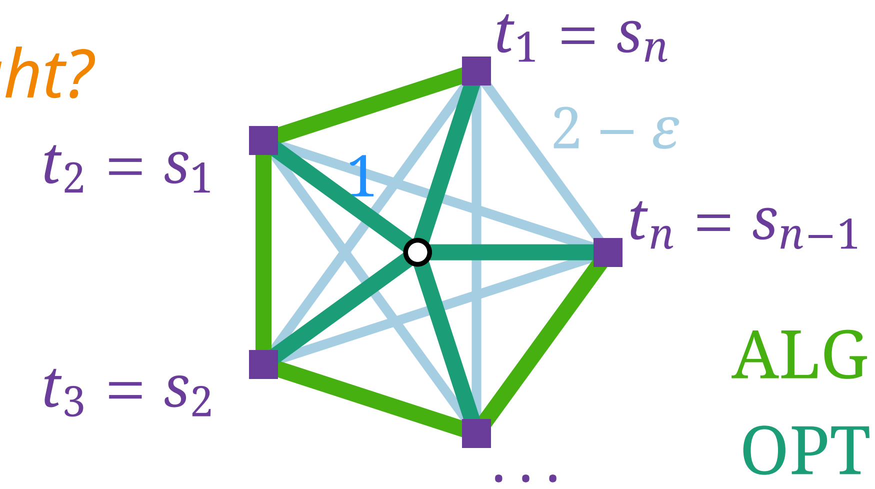
No better approximation factor is known. :-(

The integrality gap is $2 - 1/n$.

Summary

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

Is our analysis tight?



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

Can we do better?

No better approximation factor is known. :-(

The integrality gap is $2 - 1/n$.

SteinerForest (as SteinerTree) cannot be approximated within factor $\frac{96}{95} \approx 1.0105$ (unless P = NP). [Chlebík, Chlebíková '08]