Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Chair for
INFORMATICS I
Efficient Algorithms and
Knowledge-Based Systems

Institute for Informatics

Approximationsalgorithmen

SteinerWald primal-dual

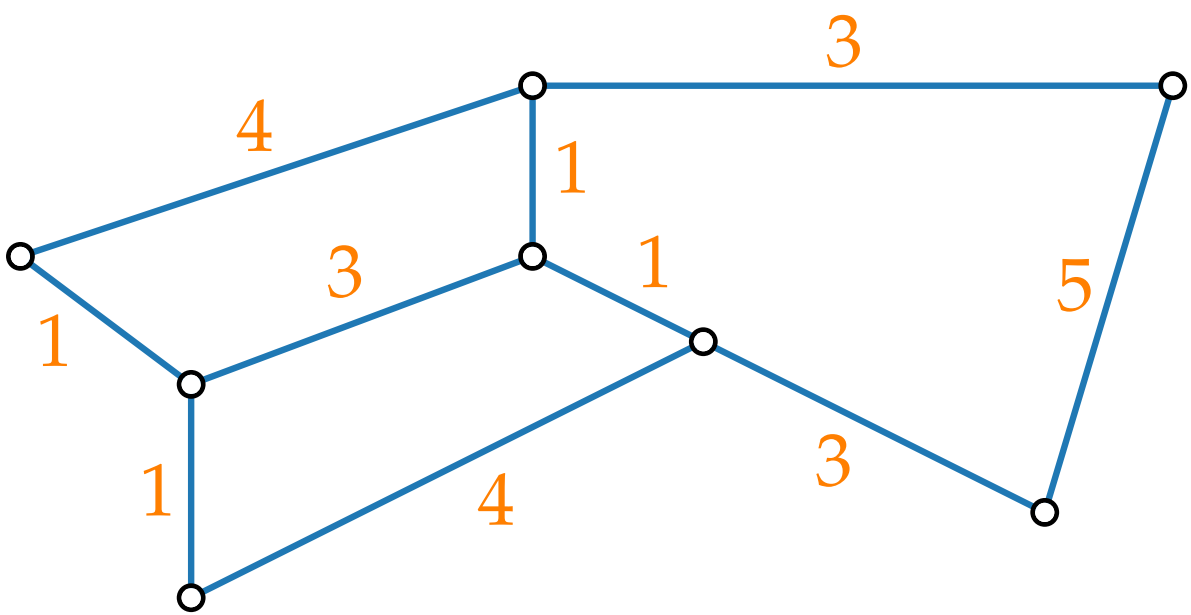
12. Vorlesung

Philipp Kindermann

Wintersemester 2019/20

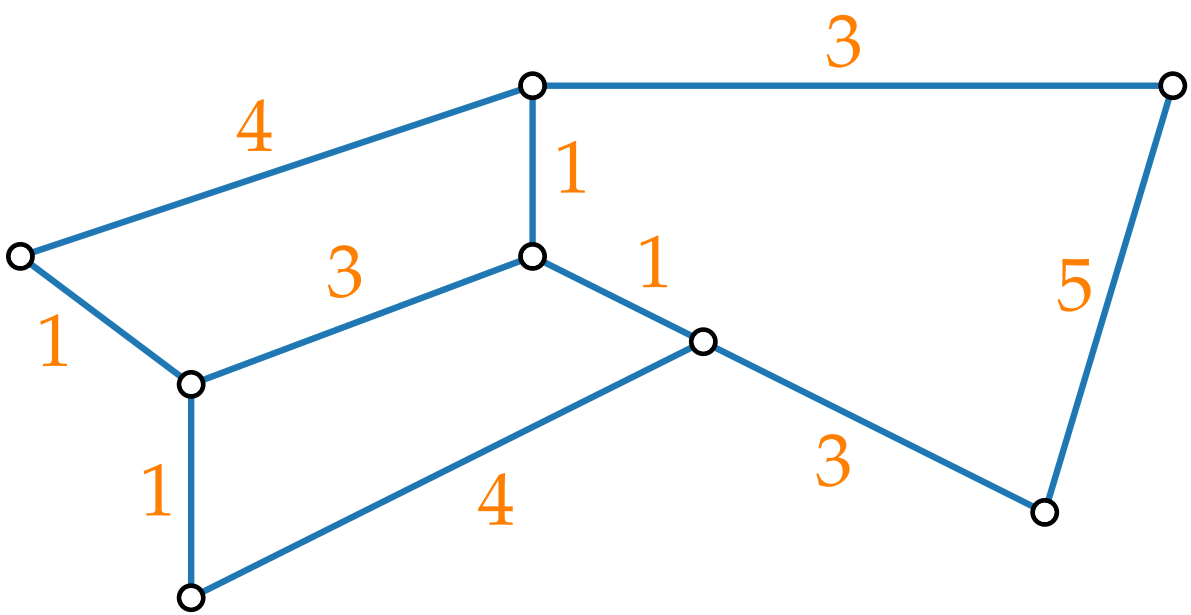
SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$



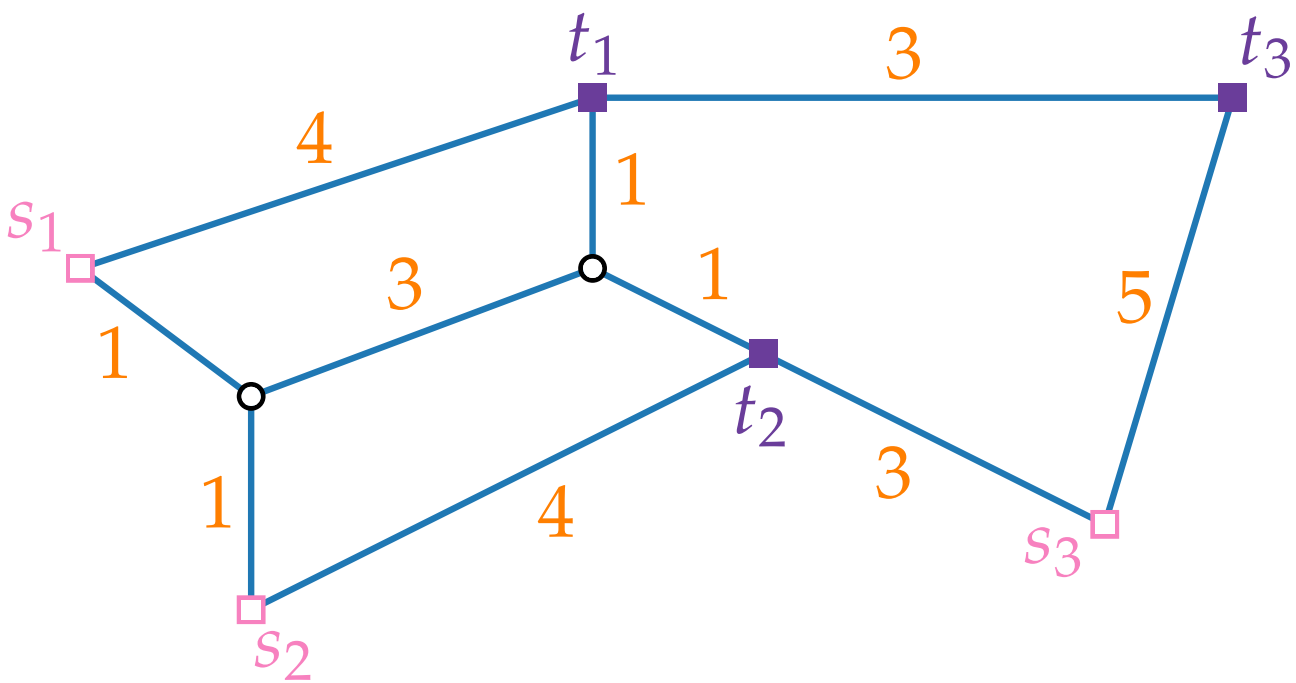
SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.



SteinerWald

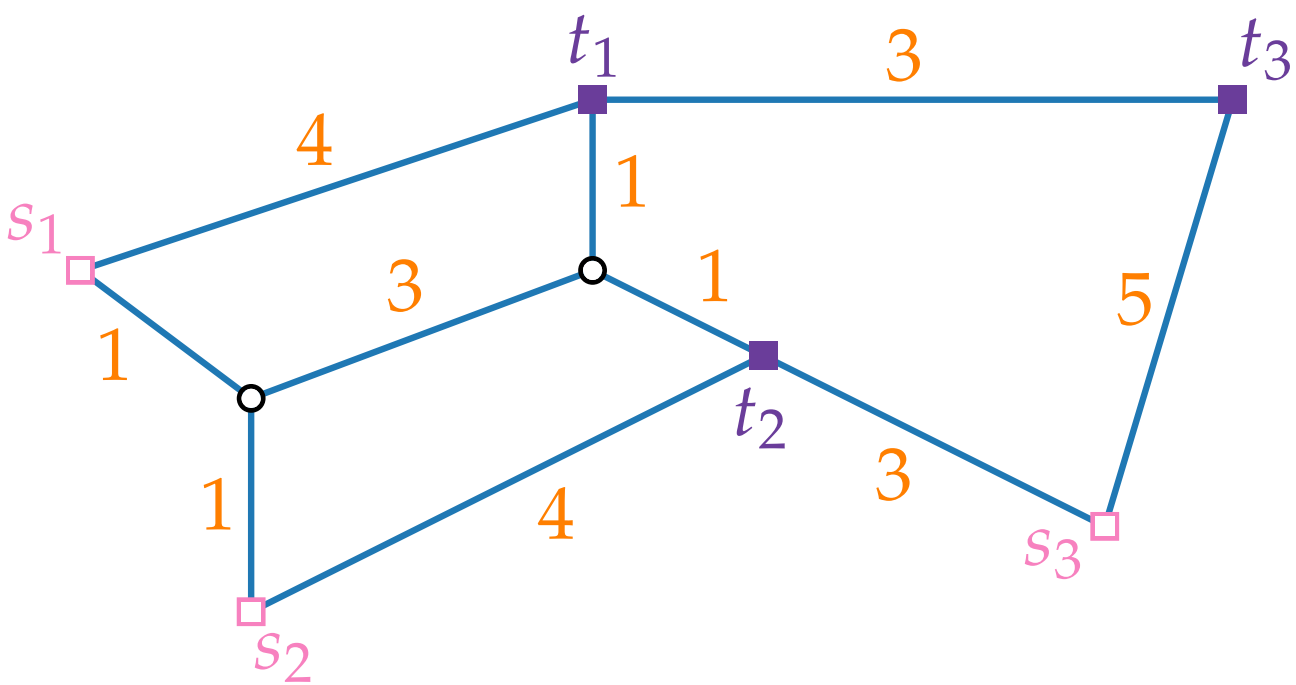
Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.



SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.

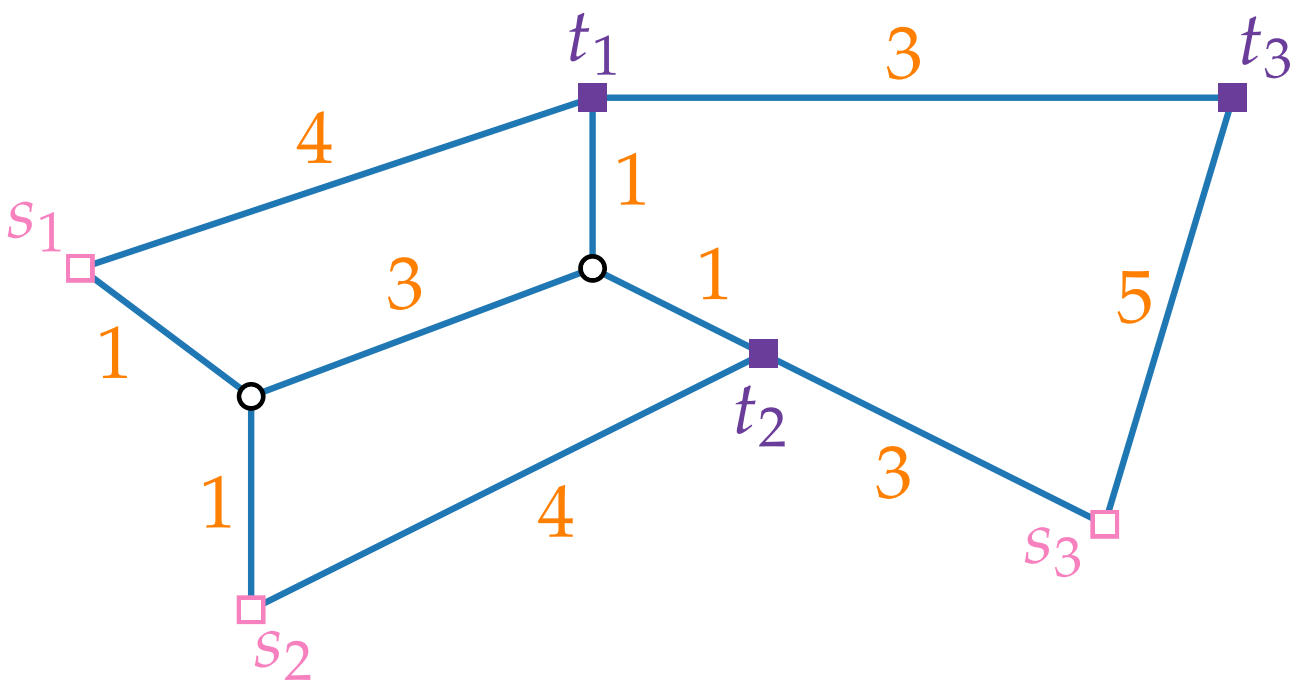
Gesucht: Eine Kantenmenge $F \subseteq E$ mit minimalen Gesamtkosten $c(F)$



SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.

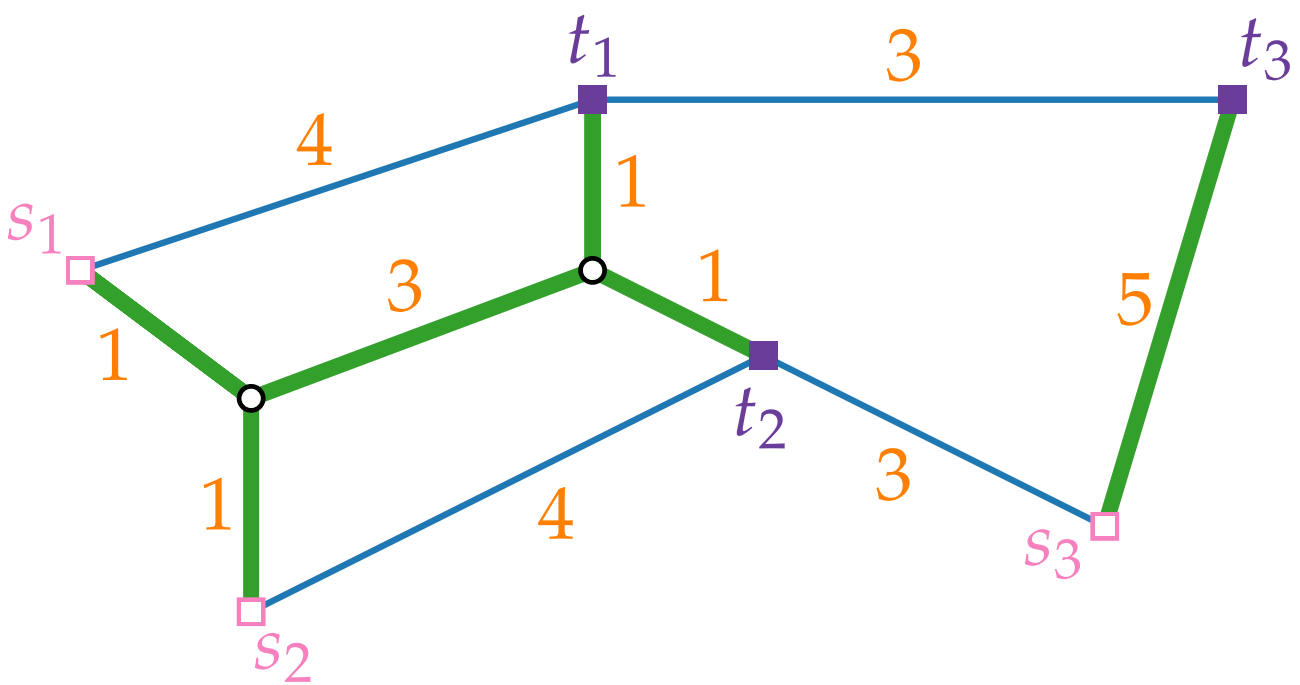
Gesucht: Eine Kantenmenge $F \subseteq E$ mit minimalen Gesamtkosten $c(F)$, so dass der Teilgraph (V, F) jedes der Paare (s_i, t_i), $i = 1, \ldots, k$ verbindet.



SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.

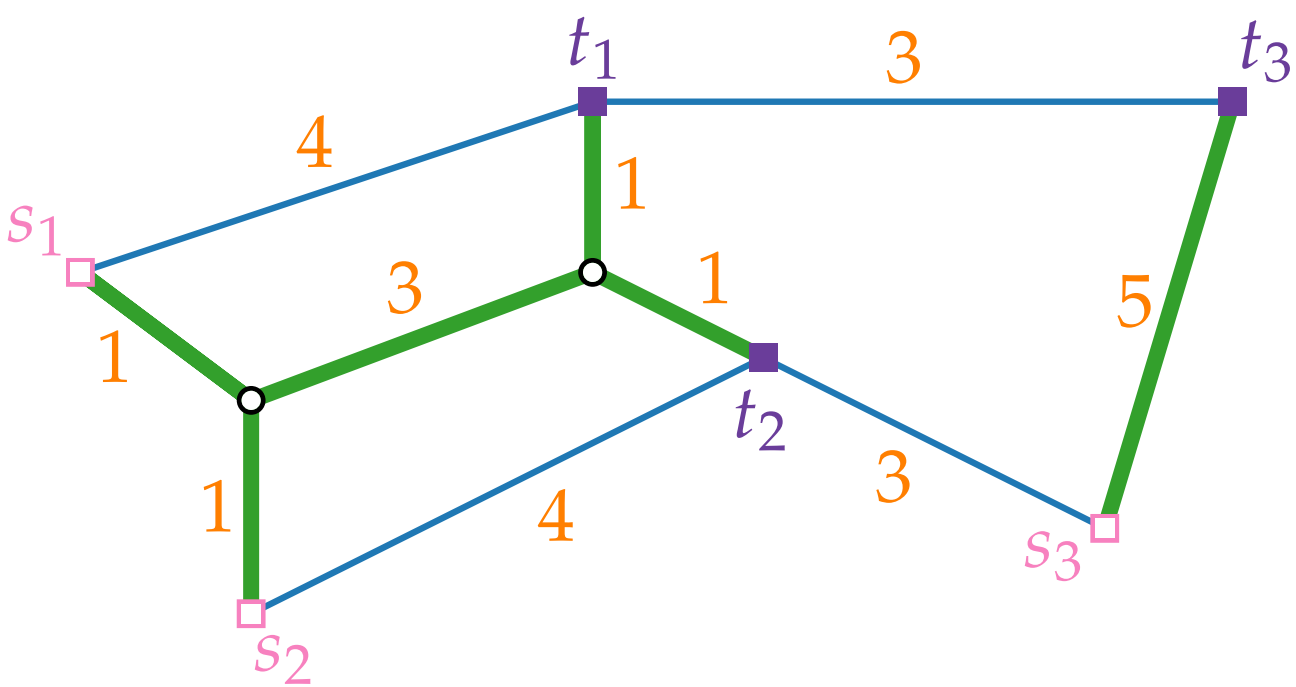
Gesucht: Eine Kantenmenge $F \subseteq E$ mit minimalen Gesamtkosten $c(F)$, so dass der Teilgraph (V, F) jedes der Paare (s_i, t_i), $i = 1, \ldots, k$ verbindet.



SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.

Gesucht: Eine Kantenmenge $F \subseteq E$ mit minimalen Gesamtkosten $c(F)$, so dass der Teilgraph (V, F) jedes der Paare (s_i, t_i), $i = 1, \ldots, k$ verbindet.

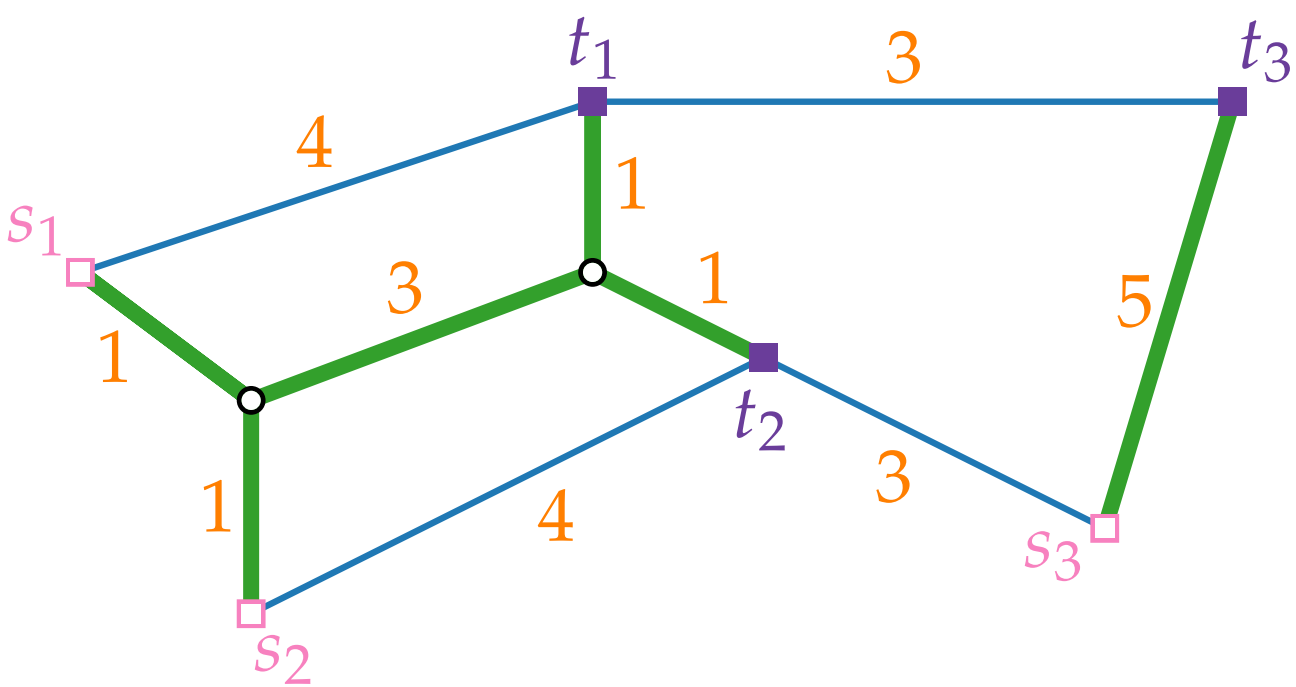


Spezialfälle?

SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.

Gesucht: Eine Kantenmenge $F \subseteq E$ mit minimalen Gesamtkosten $c(F)$, so dass der Teilgraph (V, F) jedes der Paare (s_i, t_i), $i = 1, \ldots, k$ verbindet.



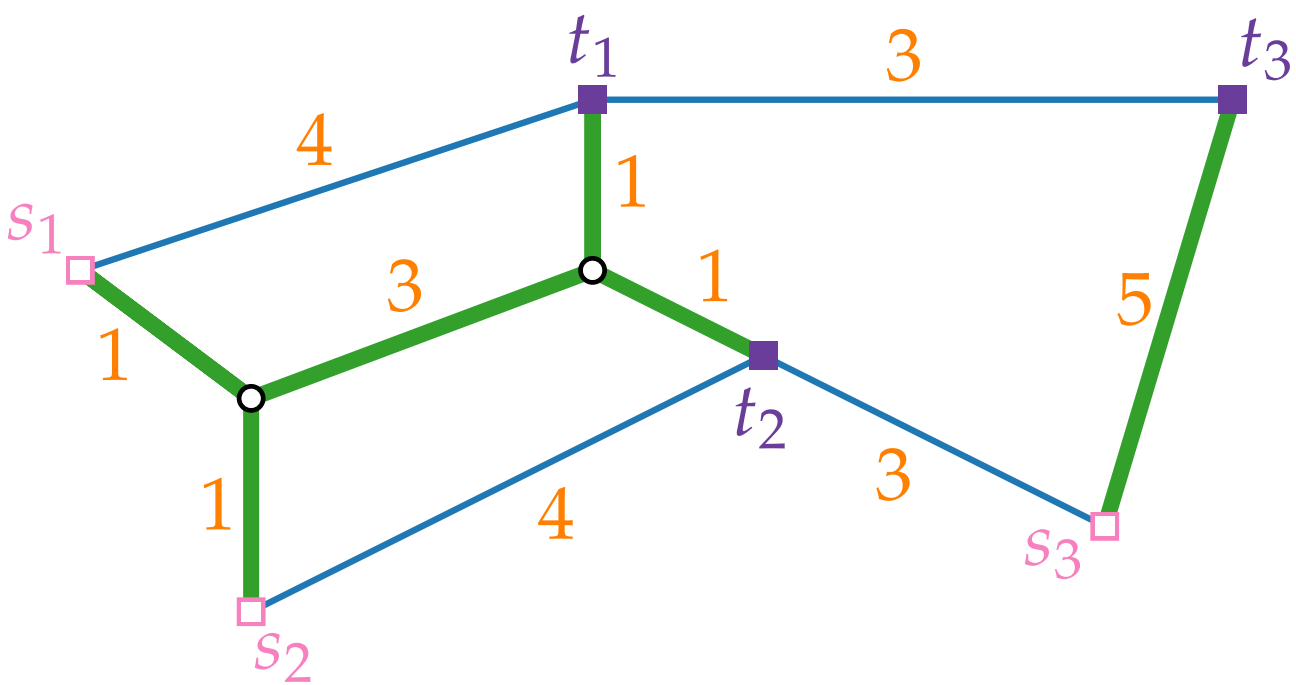
Spezialfälle?

Kürzeste Wege ($k = 1$)

SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.

Gesucht: Eine Kantenmenge $F \subseteq E$ mit minimalen Gesamtkosten $c(F)$, so dass der Teilgraph (V, F) jedes der Paare (s_i, t_i), $i = 1, \ldots, k$ verbindet.



Spezialfälle?

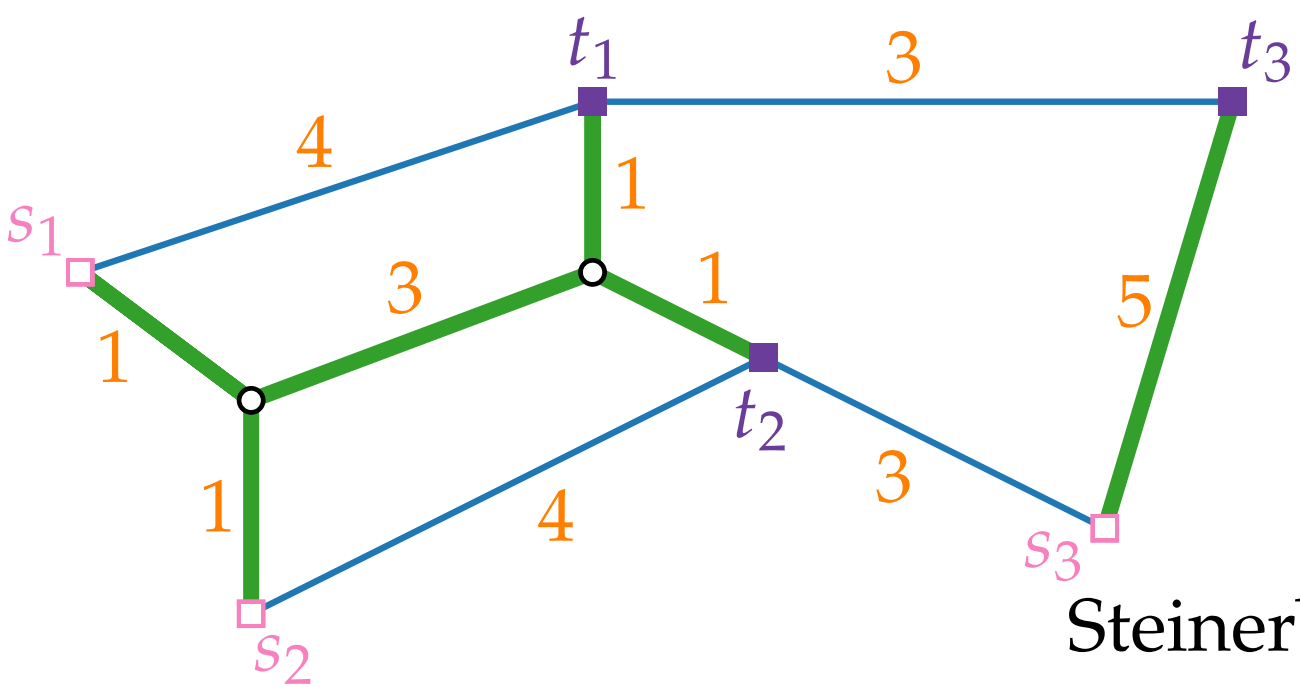
Kürzeste Wege ($k = 1$)

MST (Paare $= V \times V$)

SteinerWald

Gegeben: Ein Graph $G = (V, E)$ mit Kantengewichten $c\colon E \to \mathbb{N}$ sowie Menge $R = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ von k Paaren von Knoten.

Gesucht: Eine Kantenmenge $F \subseteq E$ mit minimalen Gesamtkosten $c(F)$, so dass der Teilgraph (V, F) jedes der Paare (s_i, t_i), $i = 1, \ldots, k$ verbindet.



Spezialfälle?

Kürzeste Wege ($k = 1$)

MST (Paare $= V \times V$)

Steinerbäume (Paare $= T \times T$)

Ansätze?

- Vereinigung k kürzester s_i-t_i-Wege

Ansätze?

- Vereinigung k kürzester s_i-t_i-Wege
- SteinerTree auf der Terminalmenge

Ansätze?

- Vereinigung k kürzester s_i-t_i-Wege
- SteinerTree auf der Terminalmenge

Obige Ansätze sind beliebig schlecht :-(

Ansätze?

- Vereinigung k kürzester s_i-t_i-Wege
- SteinerTree auf der Terminalmenge

Obige Ansätze sind beliebig schlecht :-(

Schwierigkeit: Welche Terminalmengen liegen in der gleichen Zusammenhangskomponente?

Ein ILP

minimize

subject to

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

Ein ILP

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

subject to

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

Ein ILP

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

subject to

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

t_i

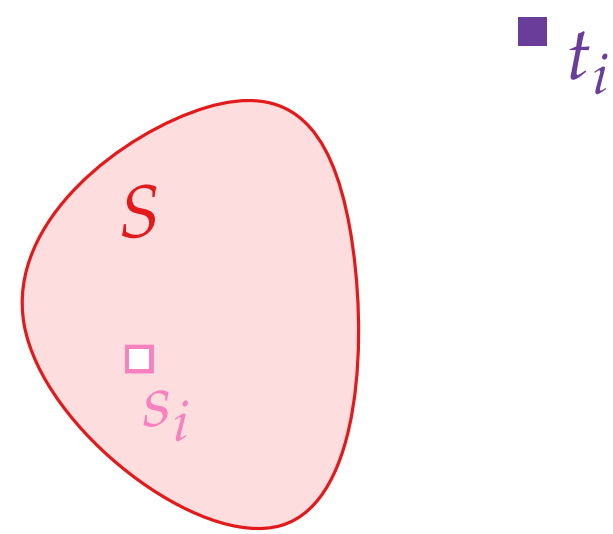
s_i

Ein ILP

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

subject to

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

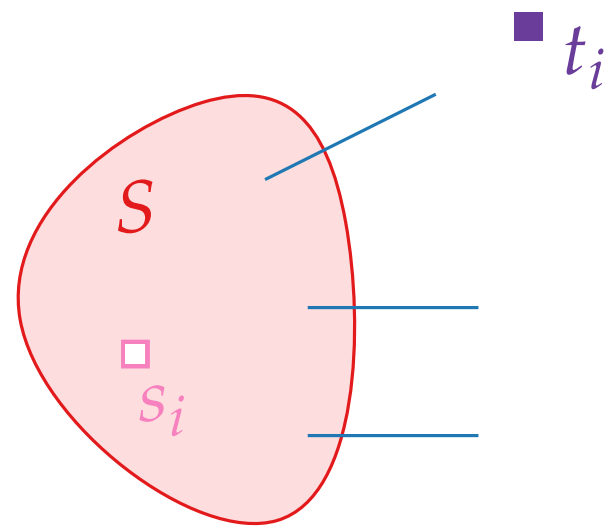


Ein ILP

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

subject to

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

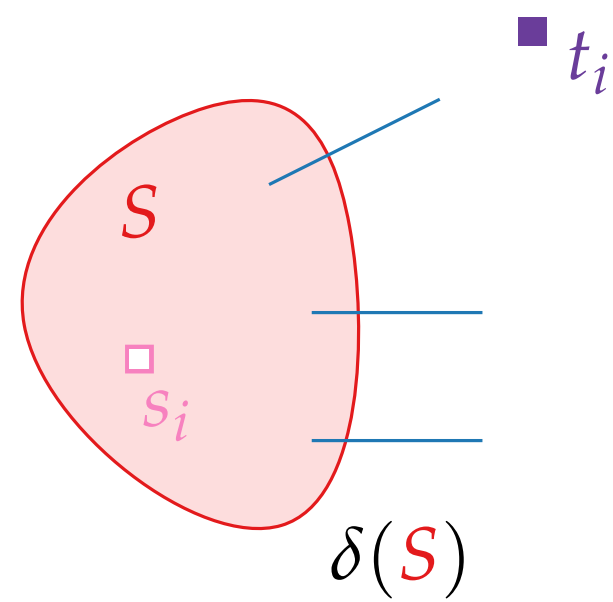


Ein ILP

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

subject to

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



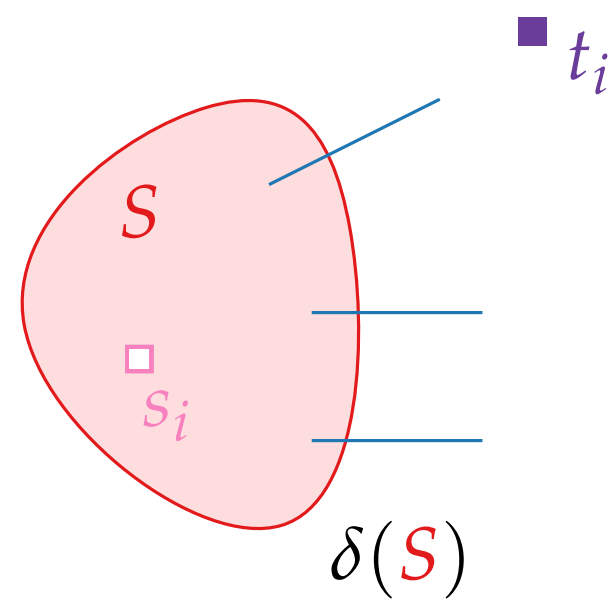
Ein ILP

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

subject to

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

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



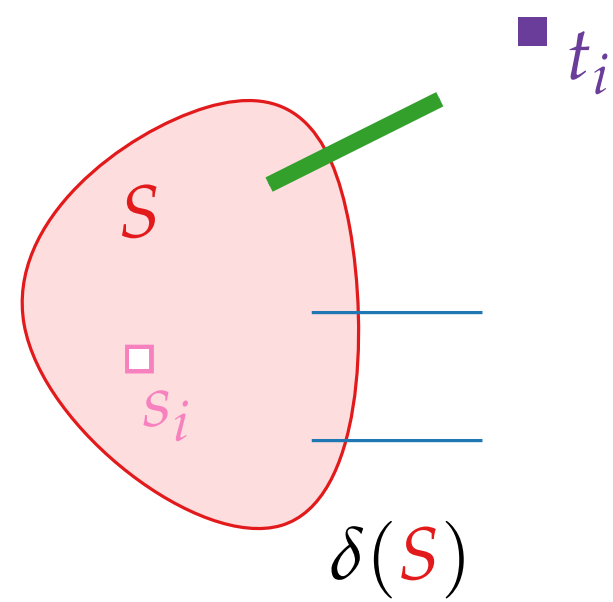
Ein ILP

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

subject to

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

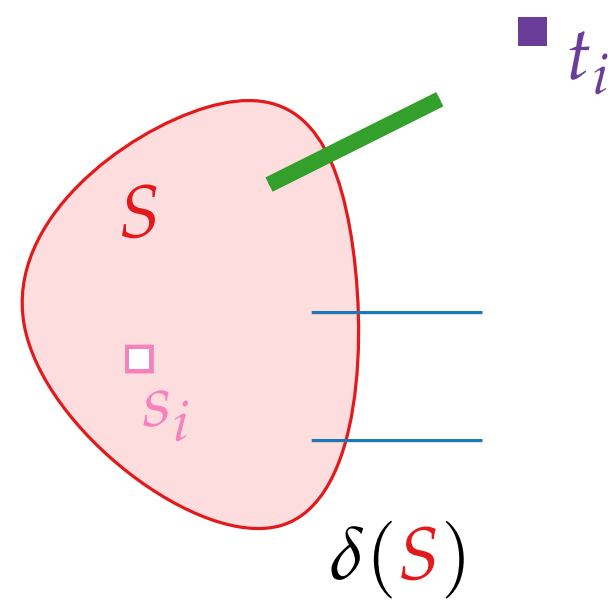
$$\delta(S) := \{(u,v) \in E \colon u \in S \text{ und } v \notin S\}$$



Ein ILP

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

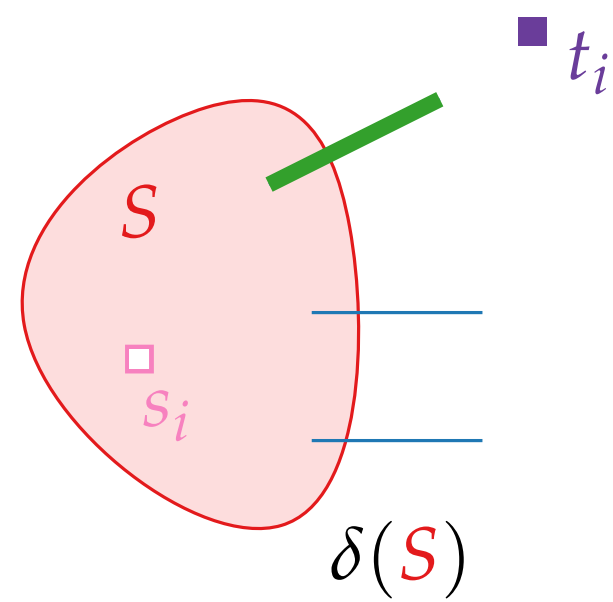
$$\delta(S) := \{(u,v) \in E \colon u \in S \text{ und } v \notin S\}$$



Ein ILP

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

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

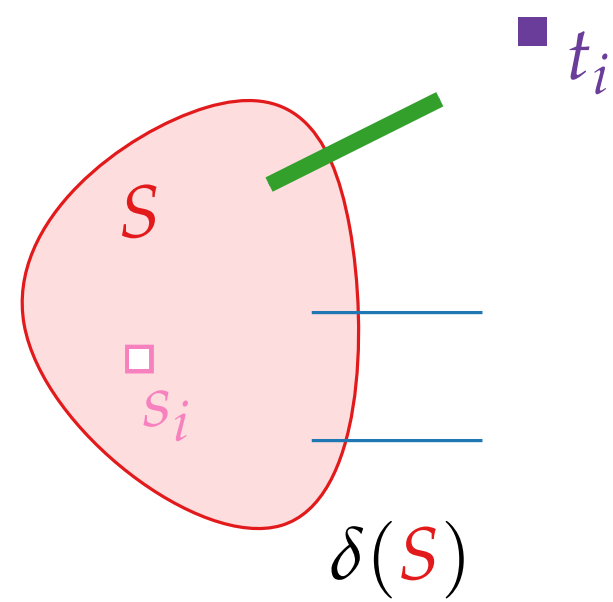


Ein ILP

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

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

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



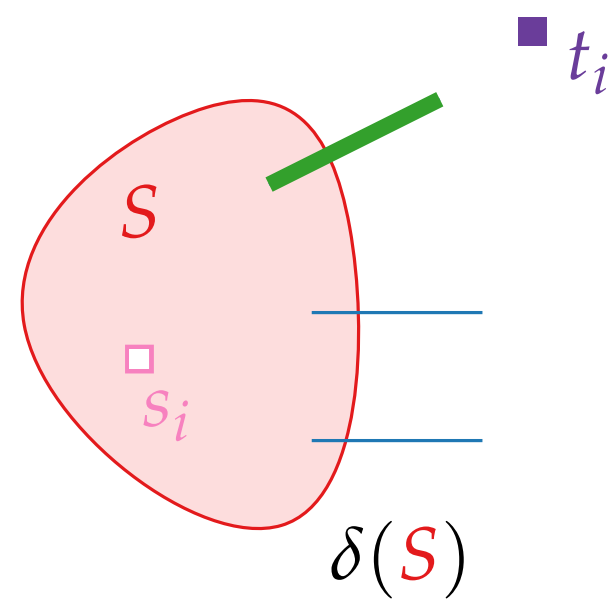
Ein ILP

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

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

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

$\rightsquigarrow$ exponentiell großes ILP!



LP-Relaxierung und Duales LP

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

LP-Relaxierung und Duales 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-Relaxierung und Duales LP

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

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

LP-Relaxierung und Duales LP

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

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

LP-Relaxierung und Duales LP

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

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

Intuition für Duales

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

Intuition für Duales

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:

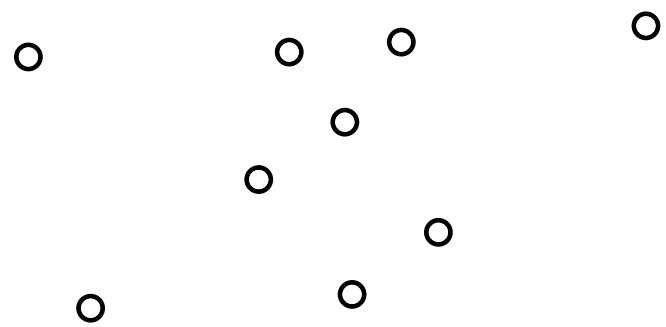
Intuition für Duales

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

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:



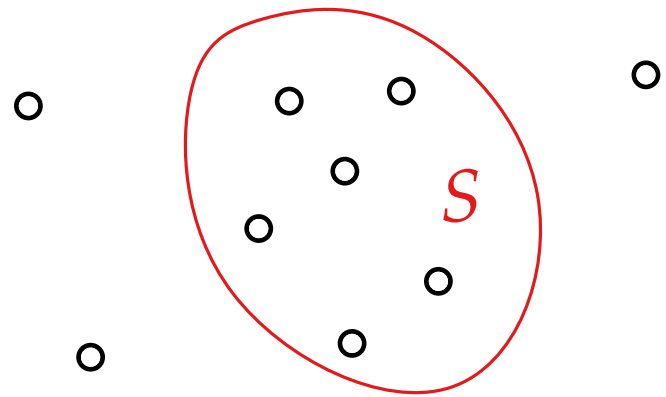
Intuition für Duales

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

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:



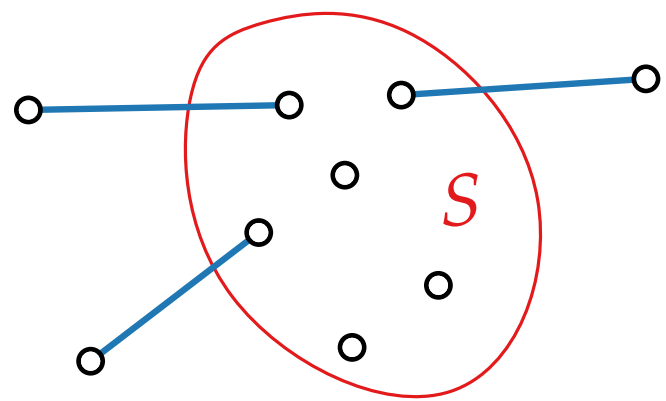
Intuition für Duales

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

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:



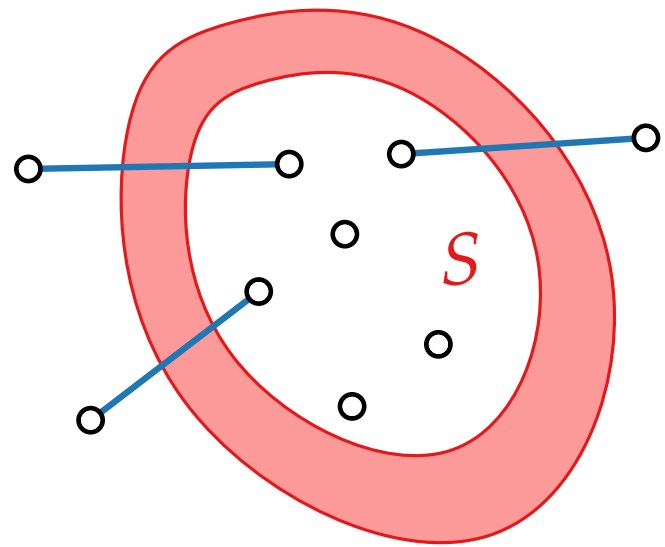
Intuition für Duales

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

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:



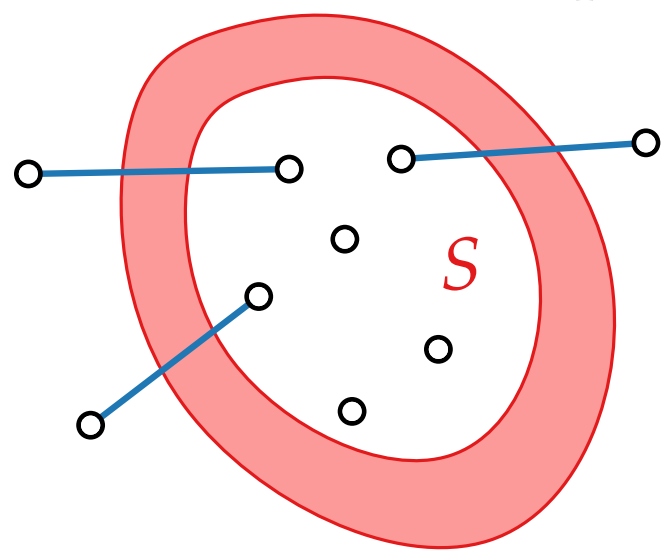
Intuition für Duales

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

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:



$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

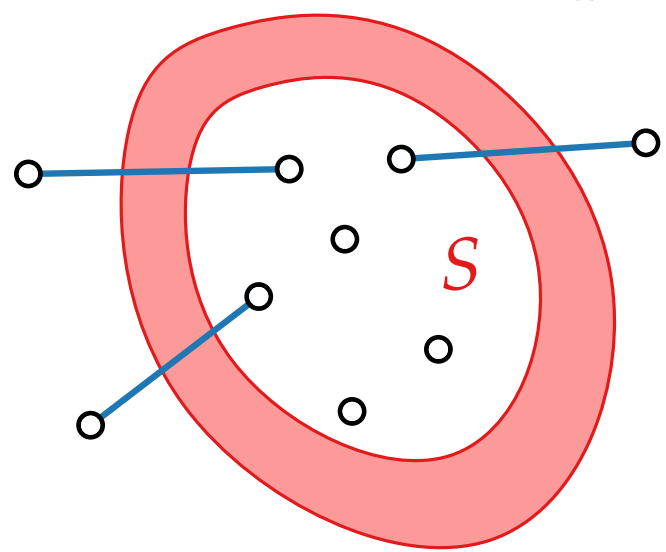
Intuition für Duales

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

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:



$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

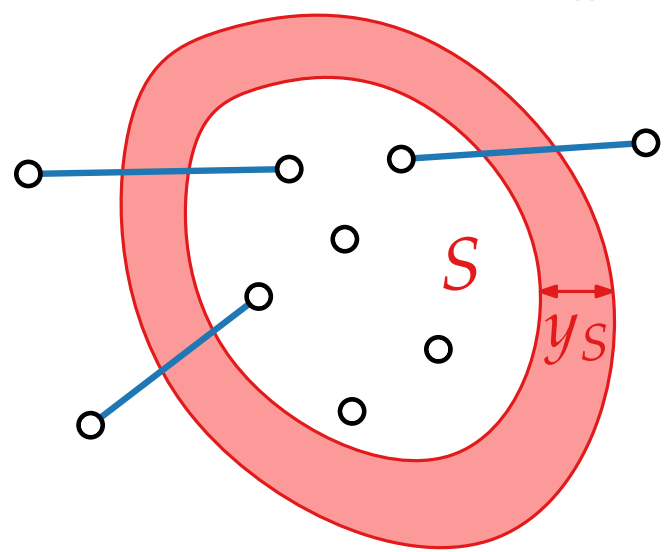
Intuition für Duales

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

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:



$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

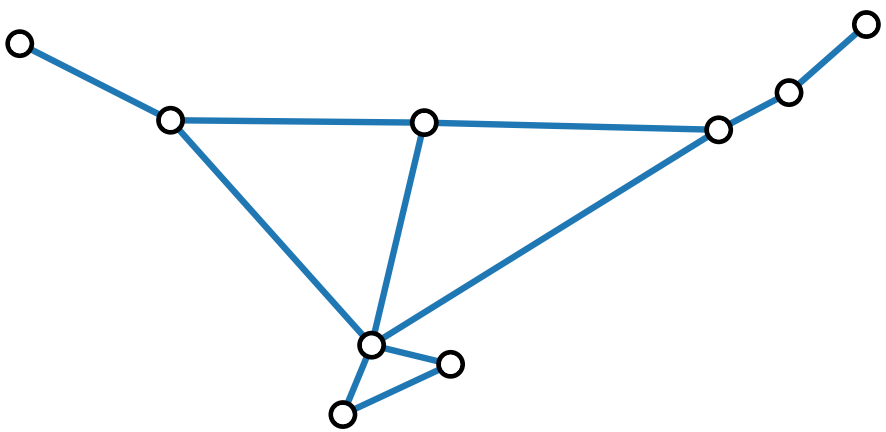
Intuition für Duales

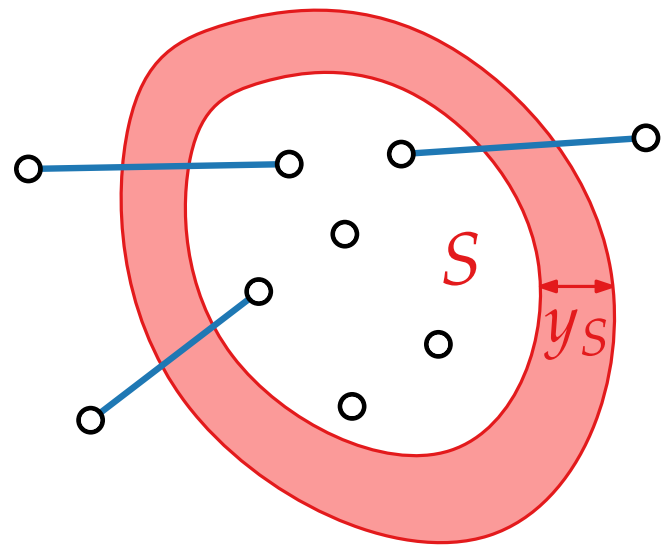
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

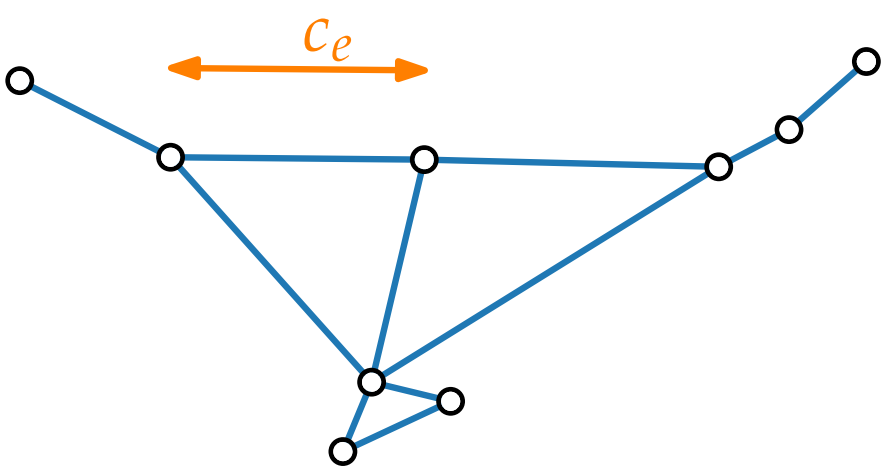
Intuition für Duales

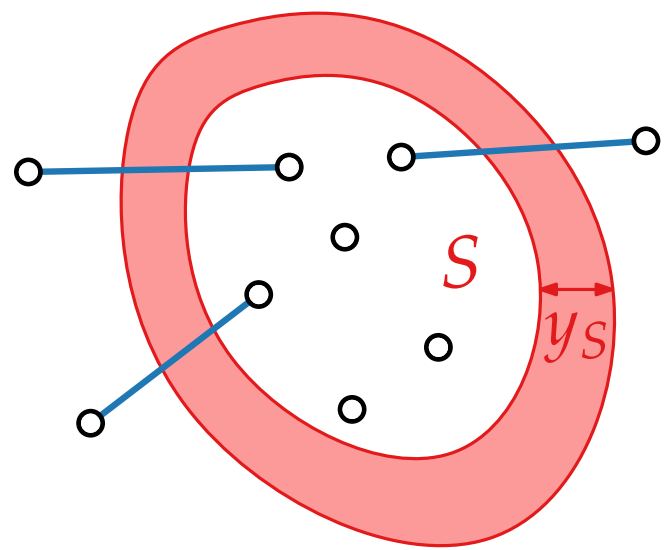
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

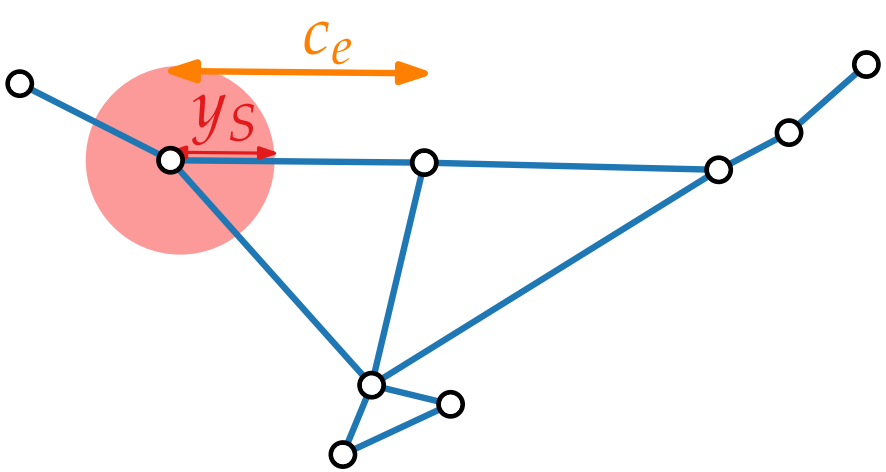
Intuition für Duales

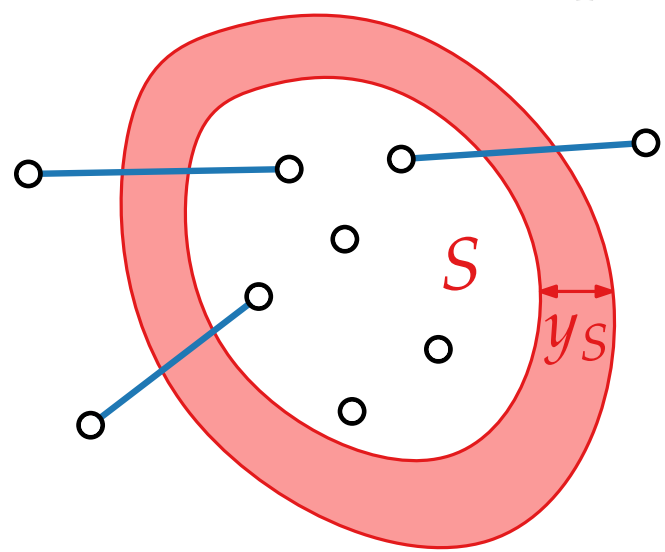
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

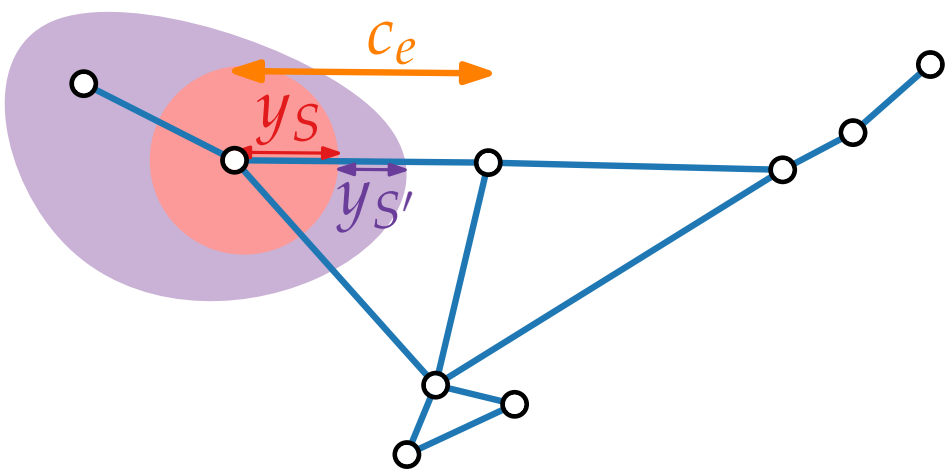
Intuition für Duales

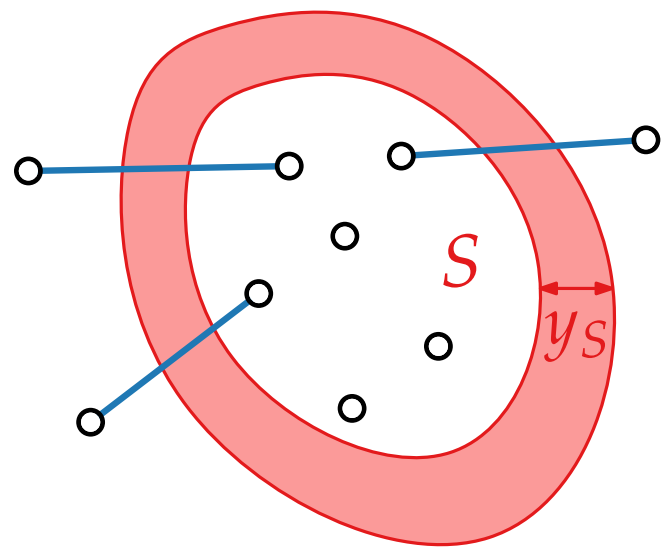
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

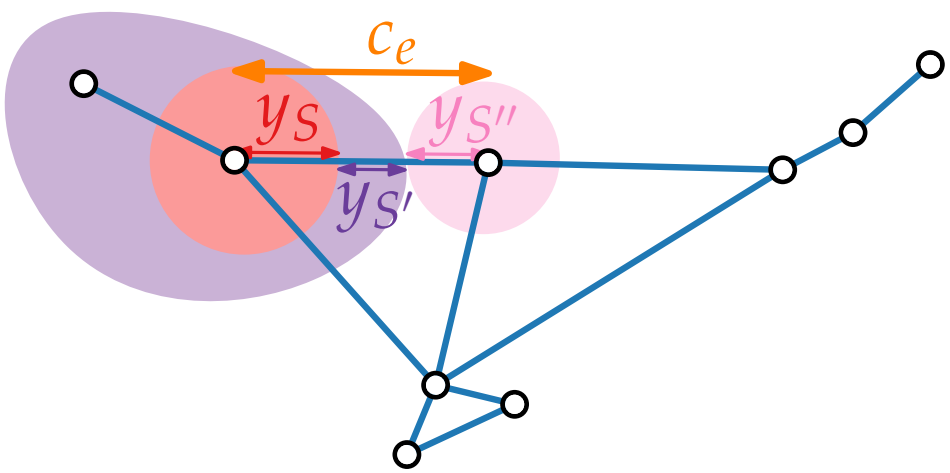
Intuition für Duales

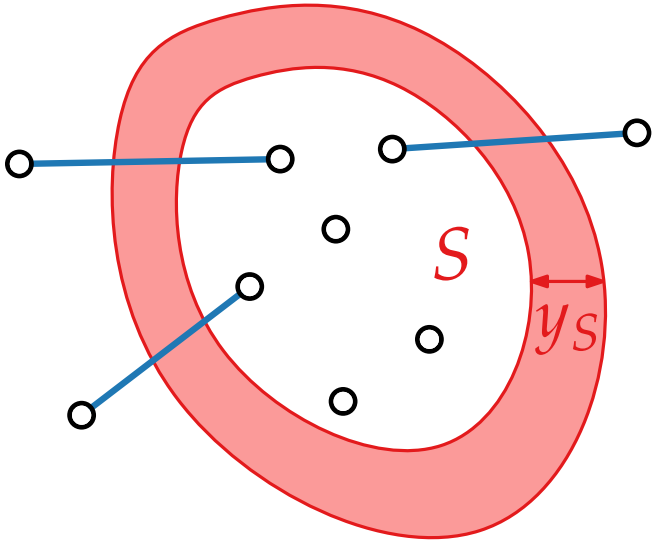
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

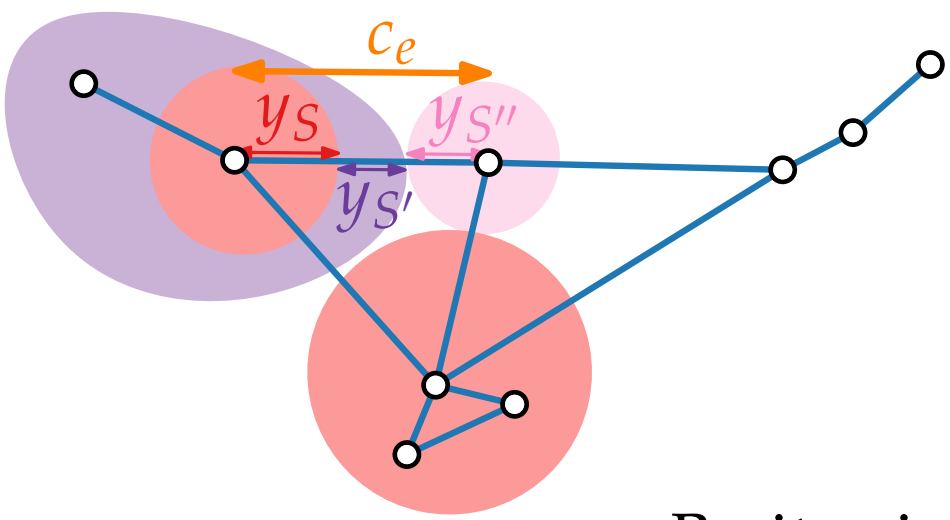
Intuition für Duales

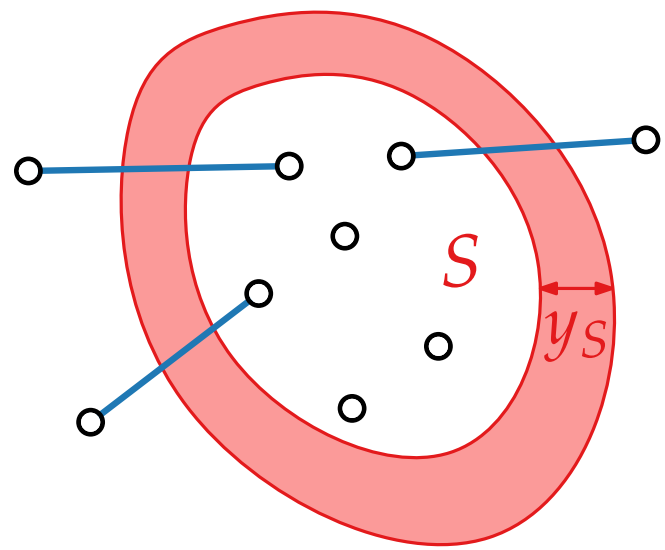
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

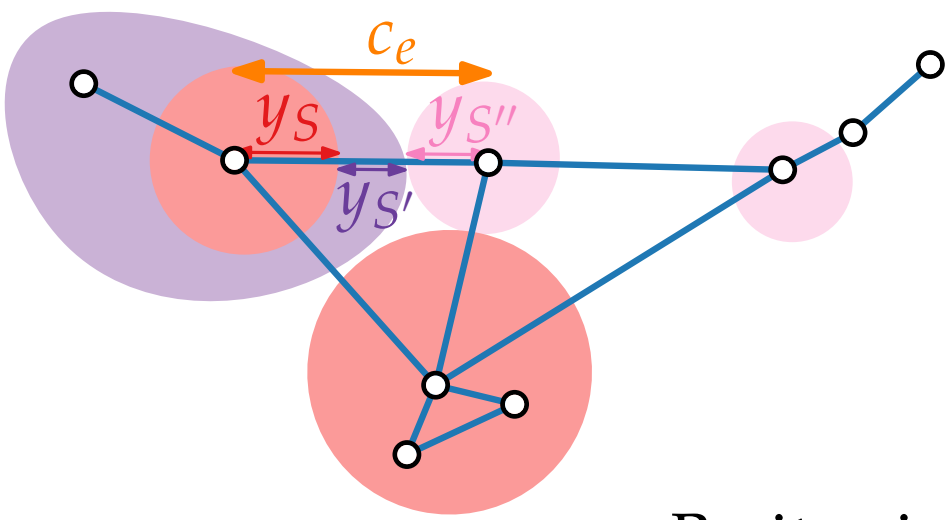
Intuition für Duales

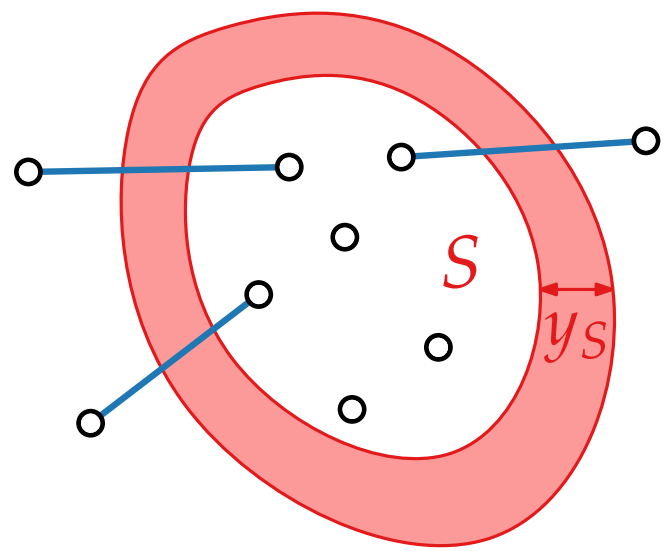
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

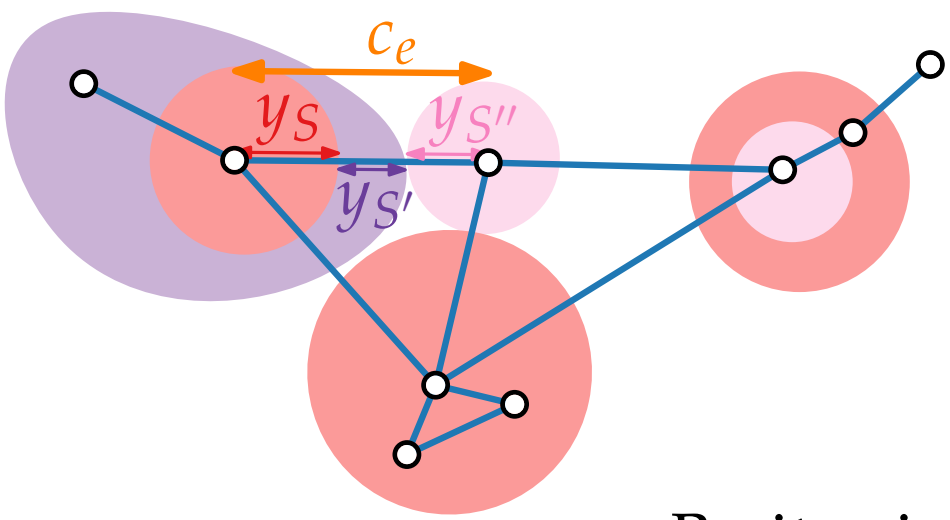
Intuition für Duales

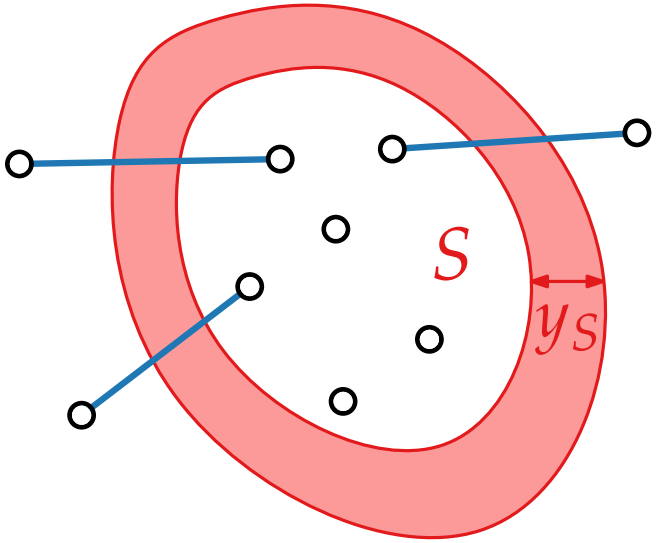
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

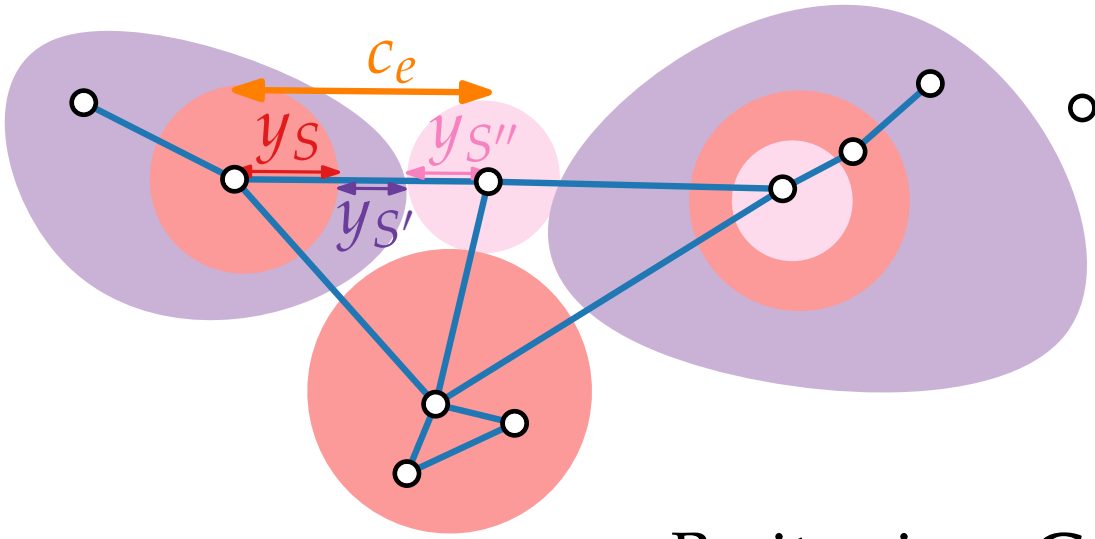
Intuition für Duales

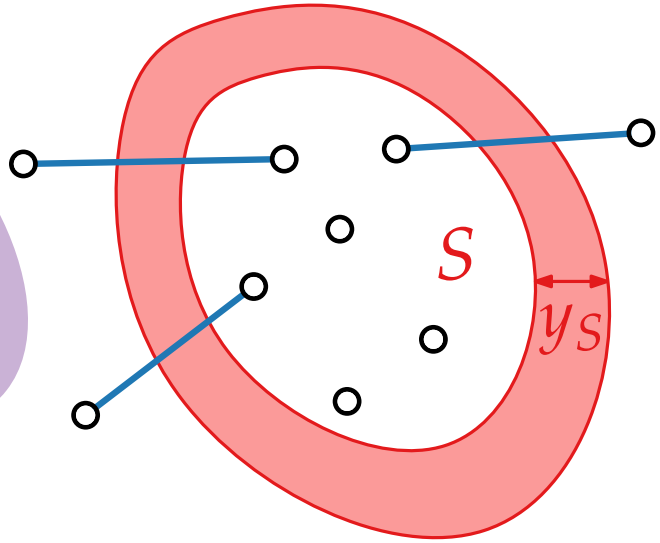
$$\textbf{maximize} \quad \sum_{\substack{S \in \mathcal{S}_i \\ i=1,\ldots,k}} y_S$$

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

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

Graph ist Netzwerk von **Brücken**, die **Gräben** überspannen:





$\delta(S) =$ Menge der Kanten / Brücken über Graben um S

$y_S =$ Breite eines **Grabens** um Menge S

Komplementärer Schlupf (Wdh.)

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

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

Satz. Seien $x = (x_1, \ldots, x_n)$ und $y = (y_1, \ldots, y_m)$ zulässige Lösungen für das primale bzw. duale Programm. Dann sind x und y genau dann optimal, wenn die folgenden Bedingungen erfüllt sind:

Primaler KS:
Für jedes $j = 1, \ldots, n$: Entweder $x_j = 0$ oder $\sum_{i=1}^{m} a_{ij} y_i = c_j$

Dualer KS:
Für jedes $i = 1, \ldots, m$: Entweder $y_i = 0$ oder $\sum_{j=1}^{n} a_{ij} x_j = b_i$

Ein erster Primal-Dual-Ansatz

Komplementärer Schlupf: $x_e > 0 \quad \Rightarrow$

Ein erster Primal-Dual-Ansatz

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

Ein erster Primal-Dual-Ansatz

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

$\Rightarrow$ Wähle „scharfe“ Kanten (und nur solche)!

Ein erster Primal-Dual-Ansatz

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

$\Rightarrow$ Wähle „scharfe“ Kanten (und nur solche)!

Wollen iterativ zulässige ganzzahlige Primal-Lösung aufbauen.

Ein erster Primal-Dual-Ansatz

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

$\Rightarrow$ Wähle „scharfe“ Kanten (und nur solche)!

Wollen iterativ zulässige ganzzahlige Primal-Lösung aufbauen.

Wie finden wir verletzte primale Beschränkung?

$\left(\sum_{e \in \delta(S)} x_e < 1\right)$

Ein erster Primal-Dual-Ansatz

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

$\Rightarrow$ Wähle „scharfe“ Kanten (und nur solche)!

Wollen iterativ zulässige ganzzahlige Primal-Lösung aufbauen.

Wie finden wir verletzte primale Beschränkung?

$\rightsquigarrow$ Zusammenhangskomponente C! $\left(\sum_{e \in \delta(S)} x_e < 1\right)$

Ein erster Primal-Dual-Ansatz

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

$\Rightarrow$ Wähle „scharfe“ Kanten (und nur solche)!

Wollen iterativ zulässige ganzzahlige Primal-Lösung aufbauen.

Wie finden wir verletzte primale Beschränkung?

$\rightsquigarrow$ Zusammenhangskomponente C! $\left(\sum_{e \in \delta(S)} x_e < 1\right)$

Wie verbessern wir (iterativ) Dual-Lösung?

Ein erster Primal-Dual-Ansatz

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

$\Rightarrow$ Wähle „scharfe“ Kanten (und nur solche)!

Wollen iterativ zulässige ganzzahlige Primal-Lösung aufbauen.

Wie finden wir verletzte primale Beschränkung?

$\rightsquigarrow$ Zusammenhangskomponente C! $\left(\sum_{e \in \delta(S)} x_e < 1\right)$

Wie verbessern wir (iterativ) Dual-Lösung?

$\rightsquigarrow$ Erhöhe y_C! (Möglich, weil keine Kante aus $\delta(C)$ scharf.)

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

return F

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

return F

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

  Bestimme Zusammenhangskomponente C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i.

return F

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

Bestimme Zusammenhangskomponente C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i.

Erhöhe y_C bis

return F

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

Bestimme Zusammenhangskomponente C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i.

Erhöhe y_C bis $\sum_{S:\, e' \in \delta(S)} y_S = c_{e'}$ für ein $e' \in \delta(C)$.

return F

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

Bestimme Zusammenhangskomponente C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i.

Erhöhe y_C bis $\sum_{S:\, e' \in \delta(S)} y_S = c_{e'}$ für ein $e' \in \delta(C)$.

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

return F

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

Bestimme Zusammenhangskomponente C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i.

Erhöhe y_C bis $\sum_{S:\, e' \in \delta(S)} y_S = c_{e'}$ für ein $e' \in \delta(C)$.

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

return F

Laufzeit?

Ein erster Primal-Dual-Ansatz

PrimalDualSteinerwaldNaiv(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

  Bestimme Zusammenhangskomponente C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i.

  Erhöhe y_C bis $\sum_{S:\, e' \in \delta(S)} y_S = c_{e'}$ für ein $e' \in \delta(C)$.

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

return F

Laufzeit?

Trick: Verwalte alle y_S mit $y_S = 0$ implizit.

Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

$$\sum_{e \in F} c_e \overset{\text{k. S.}}{=} \sum_{e \in F}$$

Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

$$\sum_{e \in F} c_e \overset{\text{k. S.}}{=} \sum_{e \in F} \sum_{S \colon e \in \delta(S)} y_S =$$

Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Es gibt leider Beispiele mit $|\delta(S) \cap F| = k$ für alle $y_S > 0$:

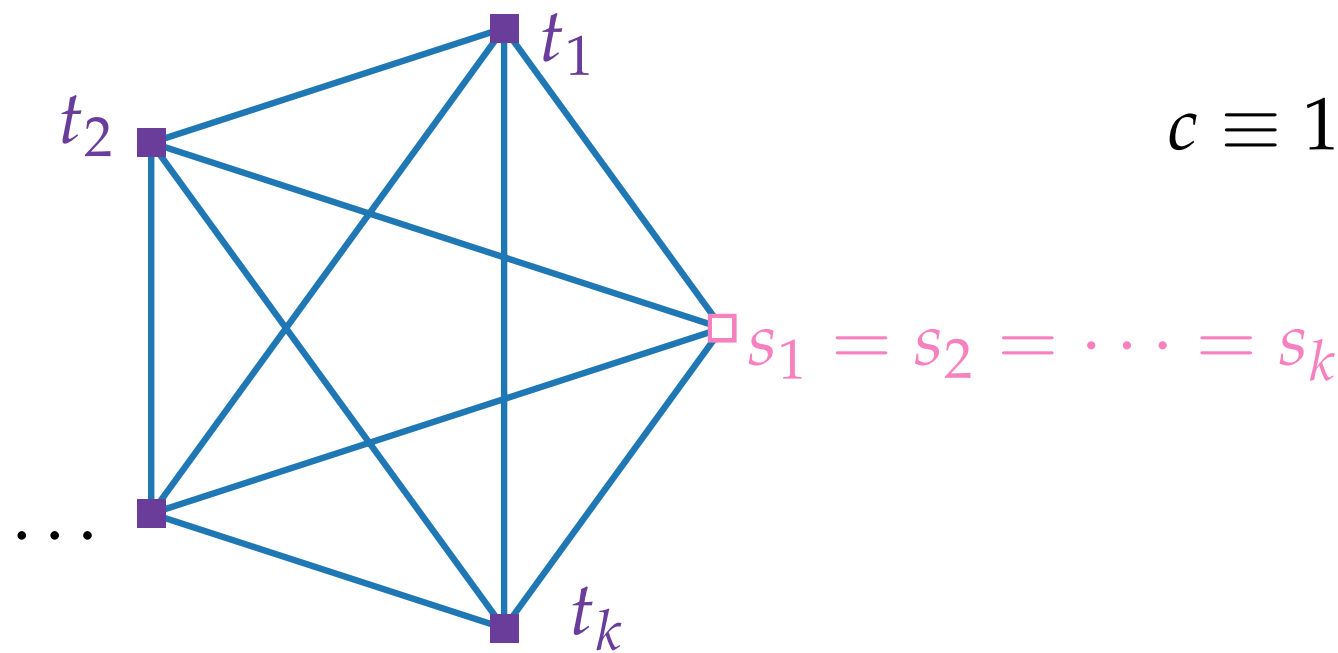
Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Es gibt leider Beispiele mit $|\delta(S) \cap F| = k$ für alle $y_S > 0$:



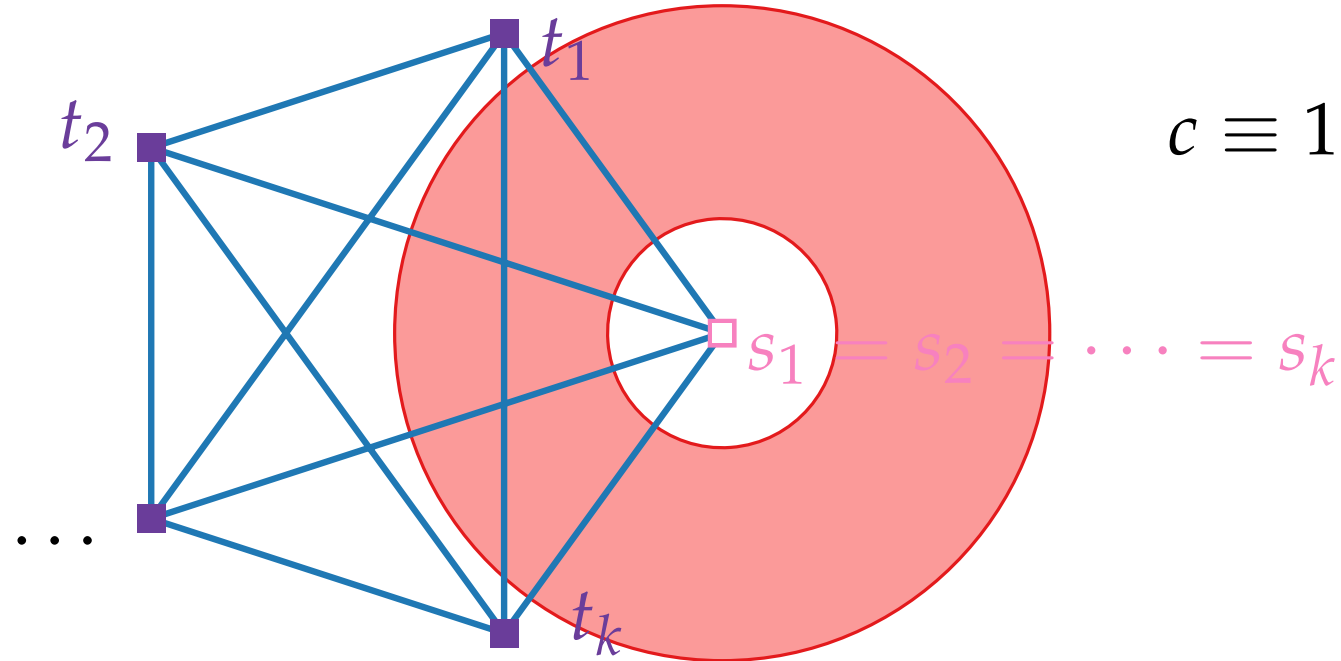
Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Es gibt leider Beispiele mit $|\delta(S) \cap F| = k$ für alle $y_S > 0$:



$c \equiv 1$

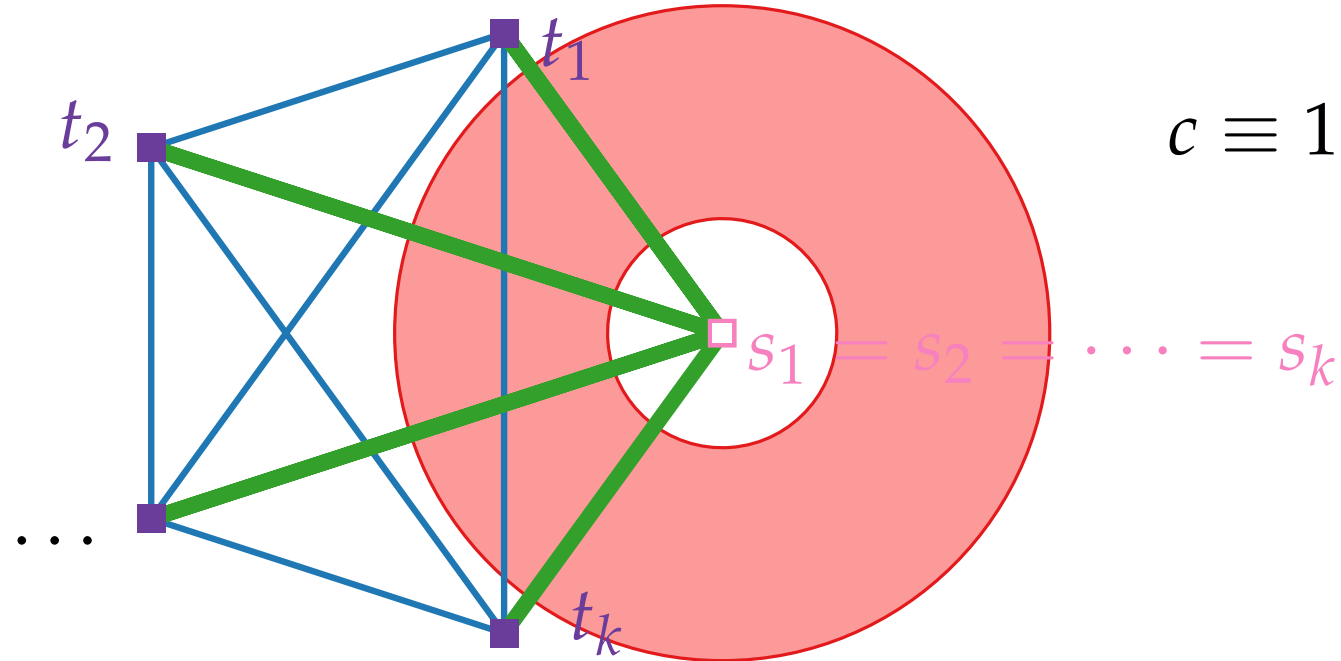
Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Es gibt leider Beispiele mit $|\delta(S) \cap F| = k$ für alle $y_S > 0$:



$c \equiv 1$

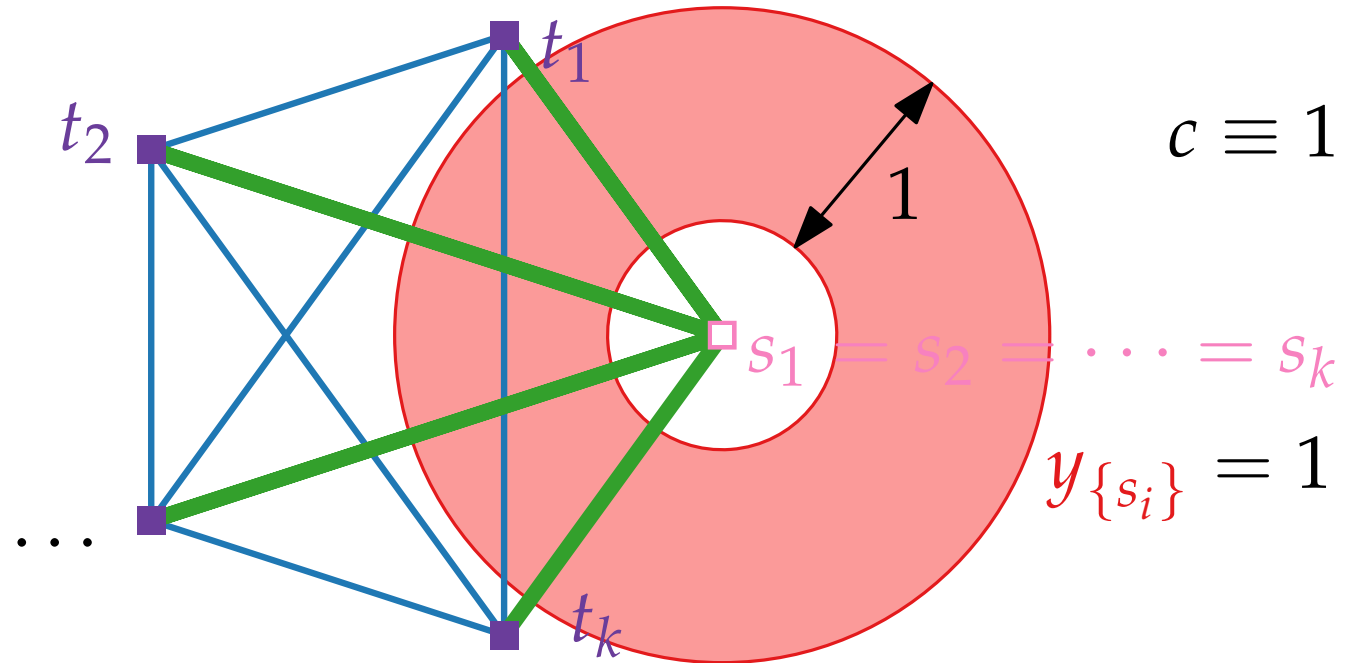
Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

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

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Es gibt leider Beispiele mit $|\delta(S) \cap F| = k$ für alle $y_S > 0$:



Analyse

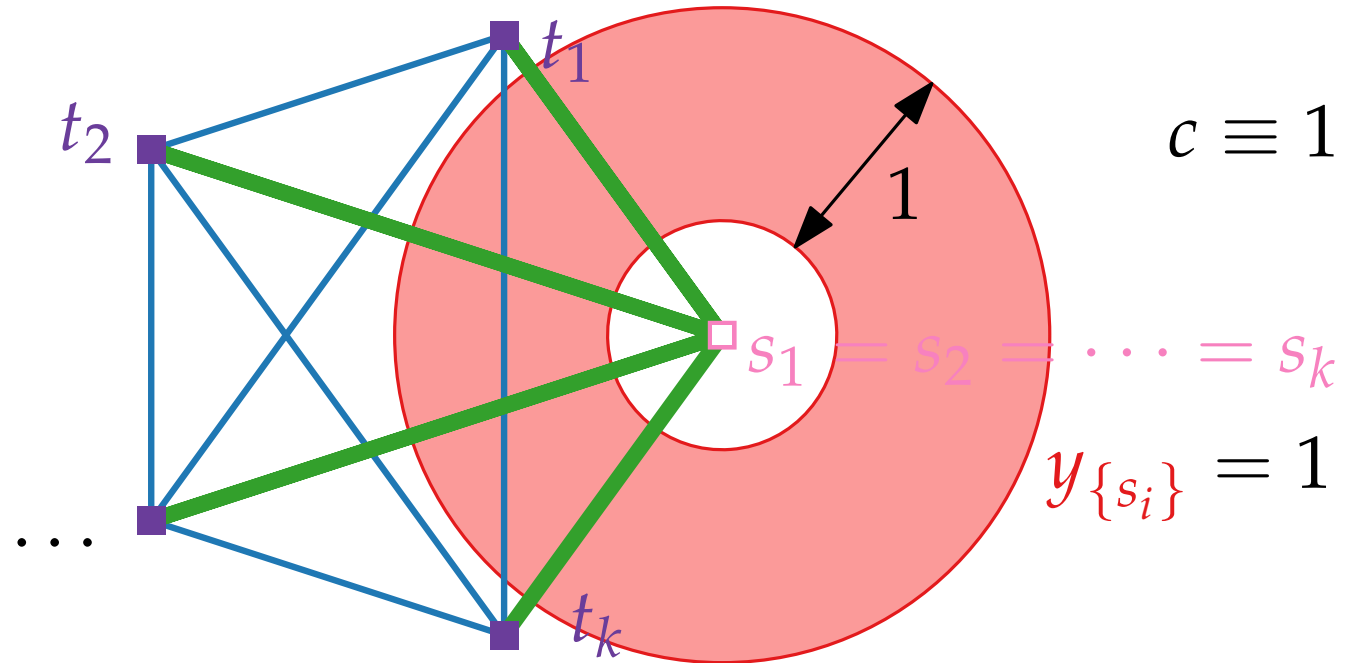
Die Kosten der finalen Lösung F lassen sich schreiben als

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

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Es gibt leider Beispiele mit $|\delta(S) \cap F| = k$ für alle $y_S > 0$:

Aber:
Durchschnittsgrad
der ZK $= 2$!



Analyse

Die Kosten der finalen Lösung F lassen sich schreiben als

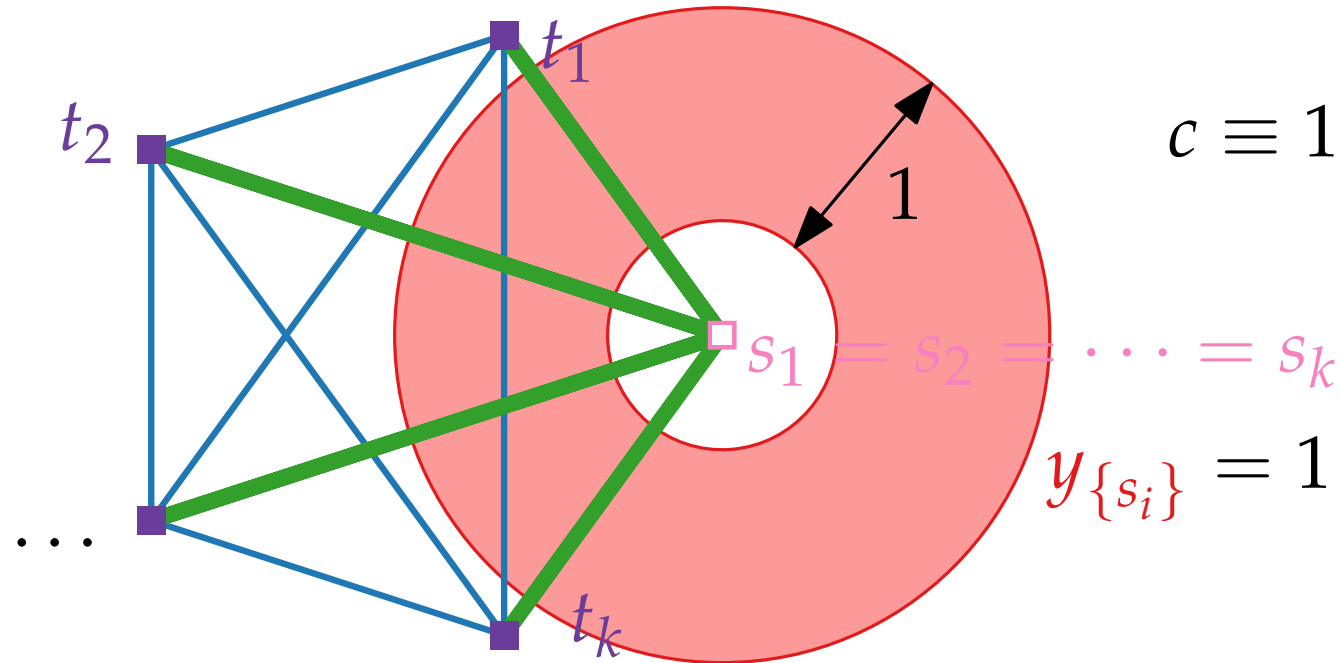
$$\sum_{e \in F} c_e \overset{\text{k. S.}}{=} \sum_{e \in F} \sum_{S \colon e \in \delta(S)} y_S = \sum_S |\delta(S) \cap F| \cdot y_S.$$

Vergleiche dies mit dem dualen Zielfunktionswert $\sum_S y_S$

Es gibt leider Beispiele mit $|\delta(S) \cap F| = k$ für alle $y_S > 0$:

Aber:
Durchschnittsgrad der ZK $= 2$!

$\Rightarrow$ Erhöhe y_C für alle ZK C simultan!



Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

  $\ell \leftarrow \ell + 1$

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

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

- $\ell \leftarrow \ell + 1$
- $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i
- $F \leftarrow F \cup \{e_\ell\}$

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

- $\ell \leftarrow \ell + 1$
- $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i
- Erhöhe y_C für alle $C \in \mathcal{C}$ einheitlich,
- $F \leftarrow F \cup \{e_\ell\}$

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

- $\ell \leftarrow \ell + 1$
- $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i
- Erhöhe y_C für alle $C \in \mathcal{C}$ einheitlich,
  bis $\displaystyle\sum_{S:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ für ein $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.
- $F \leftarrow F \cup \{e_\ell\}$

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

- $\ell \leftarrow \ell + 1$
- $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i
- Erhöhe y_C für alle $C \in \mathcal{C}$ einheitlich,
  bis $\sum_{S:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ für ein $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.
- $F \leftarrow F \cup \{e_\ell\}$

$F' \leftarrow F$

return F'

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

  $\ell \leftarrow \ell + 1$

  $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i

  Erhöhe y_C für alle $C \in \mathcal{C}$ einheitlich,

  bis $\displaystyle\sum_{S:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ für ein $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.

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

$F' \leftarrow F$

// Pruning

return F'

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

  $\ell \leftarrow \ell + 1$

  $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i

  Erhöhe y_C für alle $C \in \mathcal{C}$ einheitlich,

  bis $\displaystyle\sum_{S:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ für ein $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.

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

$F' \leftarrow F$

// Pruning

for $j \leftarrow \ell$ **down to** 1 **do**

return F'

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

  $\ell \leftarrow \ell + 1$

  $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i

  Erhöhe y_C für alle $C \in \mathcal{C}$ einheitlich,

  bis $\displaystyle\sum_{S:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ für ein $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.

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

$F' \leftarrow F$

// Pruning

for $j \leftarrow \ell$ **down to** 1 **do**

  if $F' \setminus \{e_j\}$ ist zulässige Lösung **then**

return F'

Primal-Dual mit simultaner Erhöhung

PrimalDualSteinerwald(G, c, R)

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

while nicht alle $(s_i, t_i) \in R$ sind in (V, F) verbunden **do**

  $\ell \leftarrow \ell + 1$

  $\mathcal{C} \leftarrow$ Menge aller ZK C in (V, F) mit $|C \cap \{s_i, t_i\}| = 1$ für ein i

  Erhöhe y_C für alle $C \in \mathcal{C}$ einheitlich,

  bis $\displaystyle\sum_{S:\, e_\ell \in \delta(S)} y_S = c_{e_\ell}$ für ein $e_\ell \in \delta(C)$, $C \in \mathcal{C}$.

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

$F' \leftarrow F$

// Pruning

for $j \leftarrow \ell$ **down to** 1 **do**

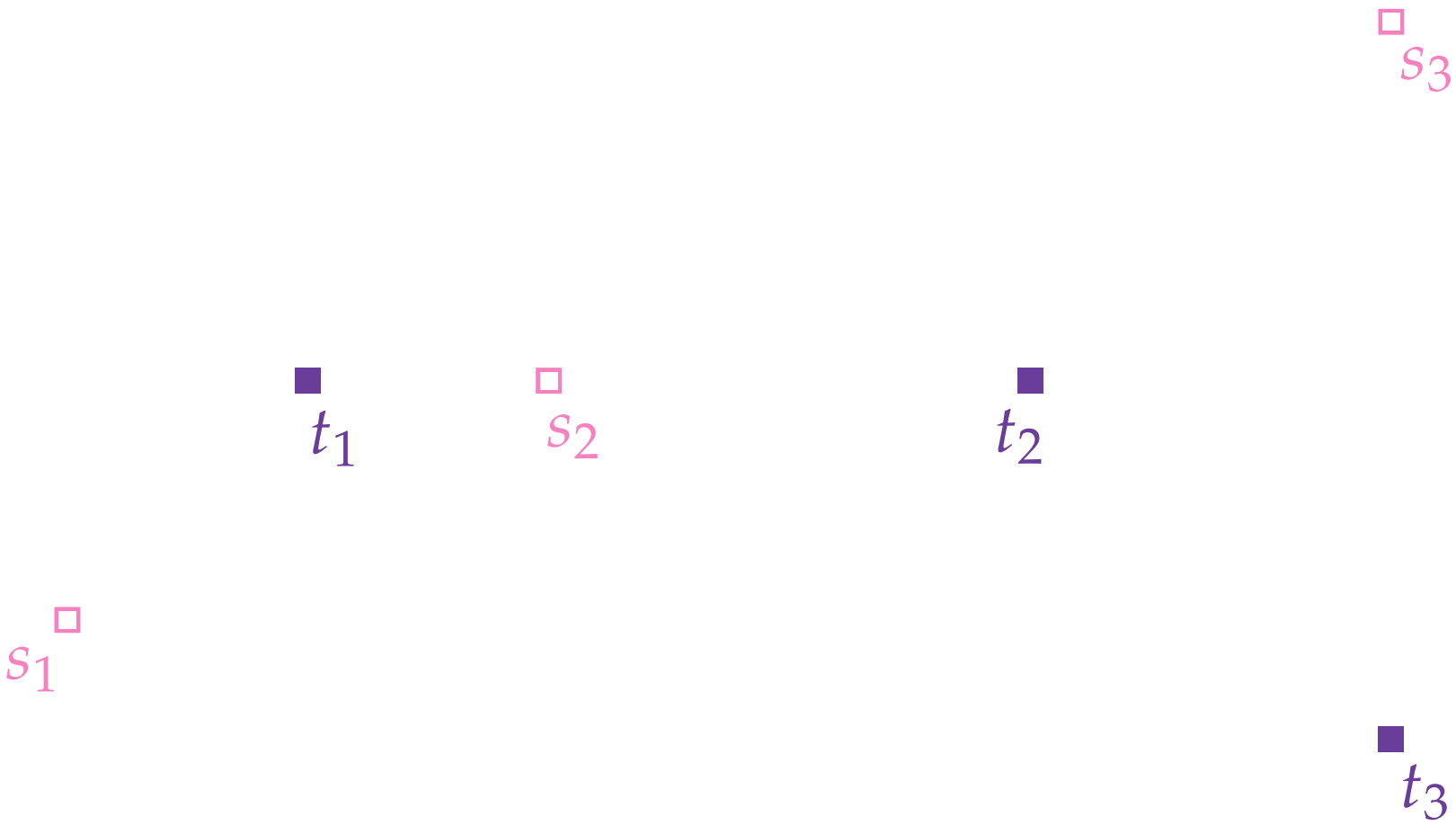
  if $F' \setminus \{e_j\}$ ist zulässige Lösung **then**

    $F' \leftarrow F' \setminus \{e_j\}$

return F'

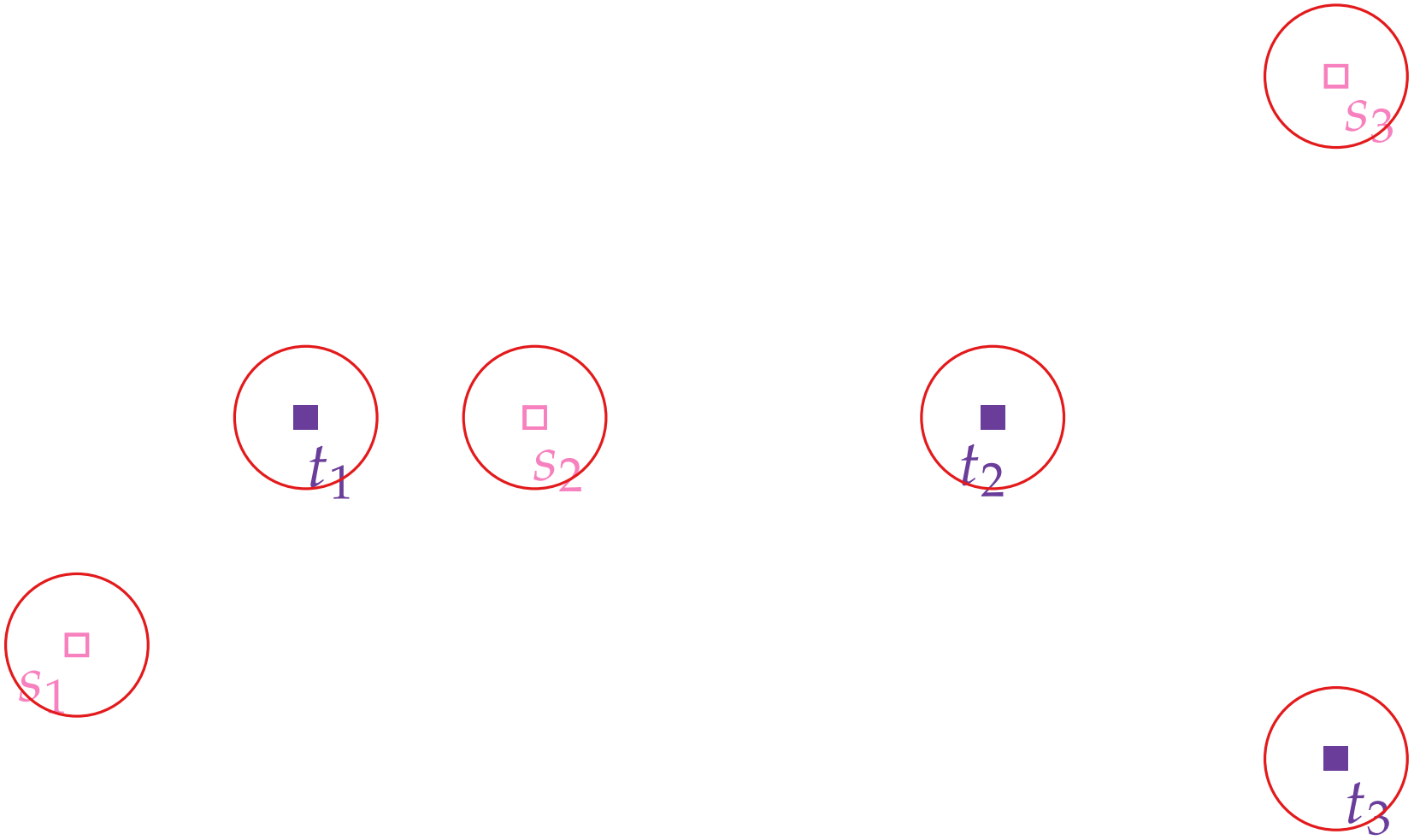
Visualisierung

$G = K_6$ mit euklidischen Kosten



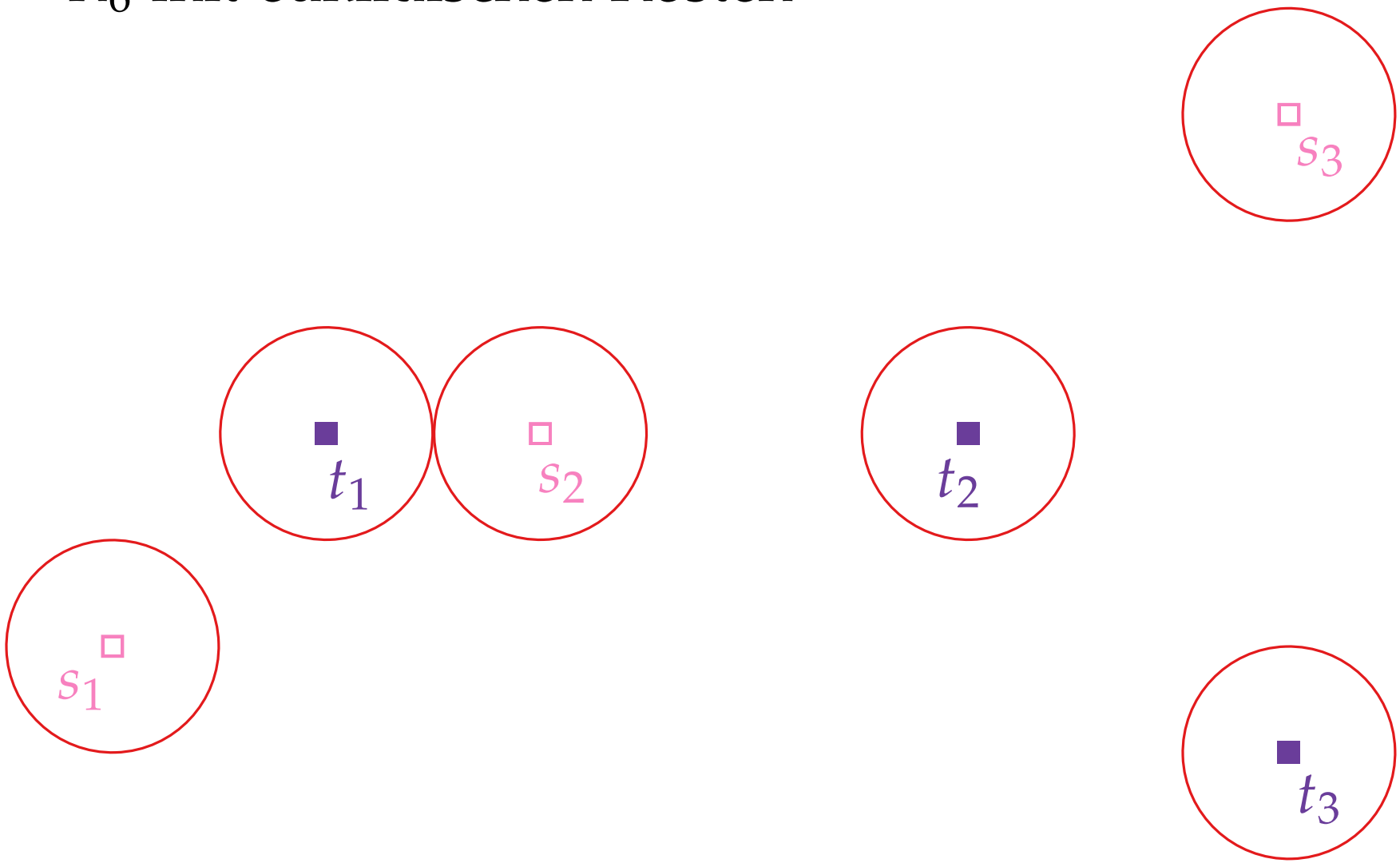
Visualisierung

$G = K_6$ mit euklidischen Kosten



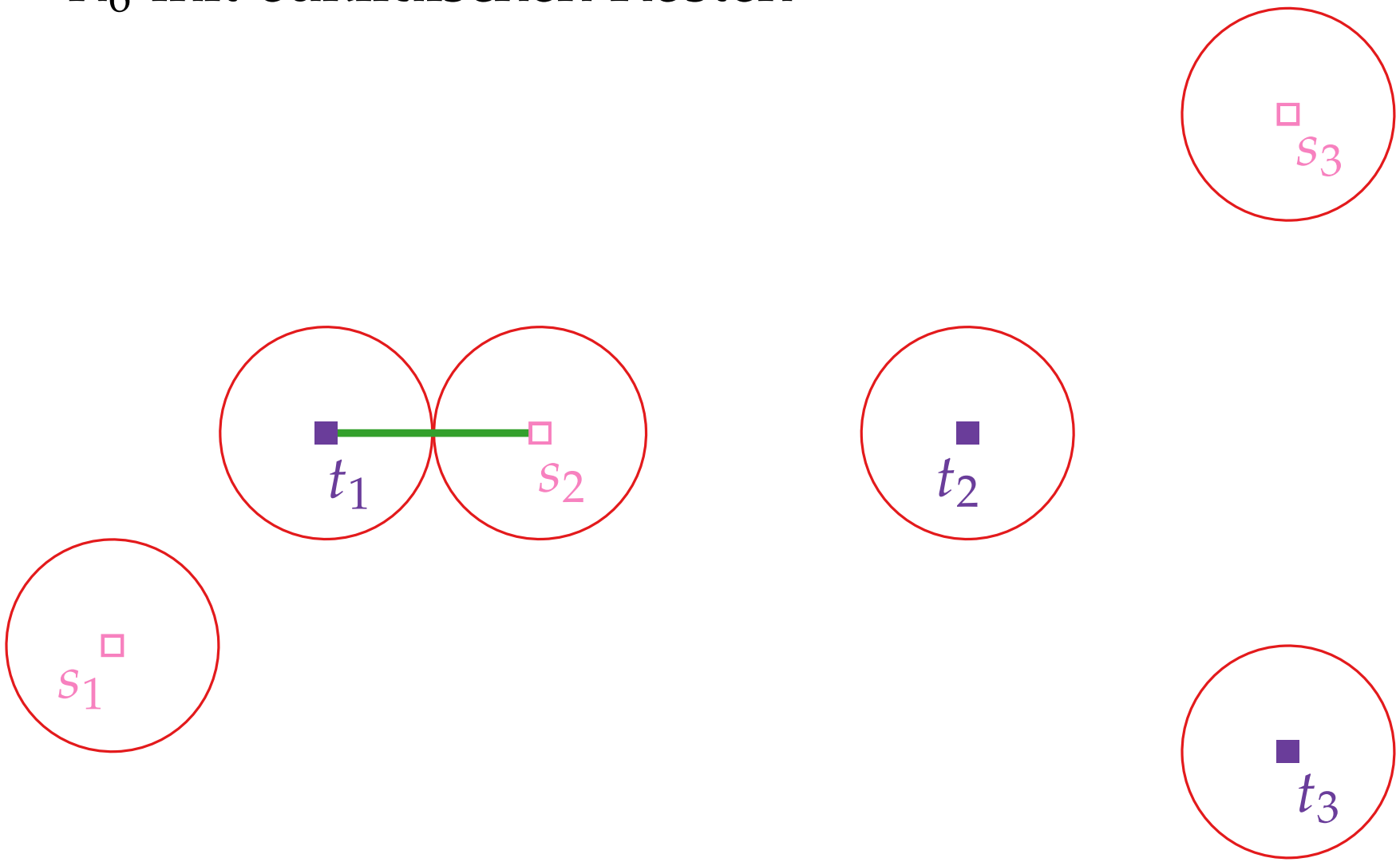
Visualisierung

$G = K_6$ mit euklidischen Kosten



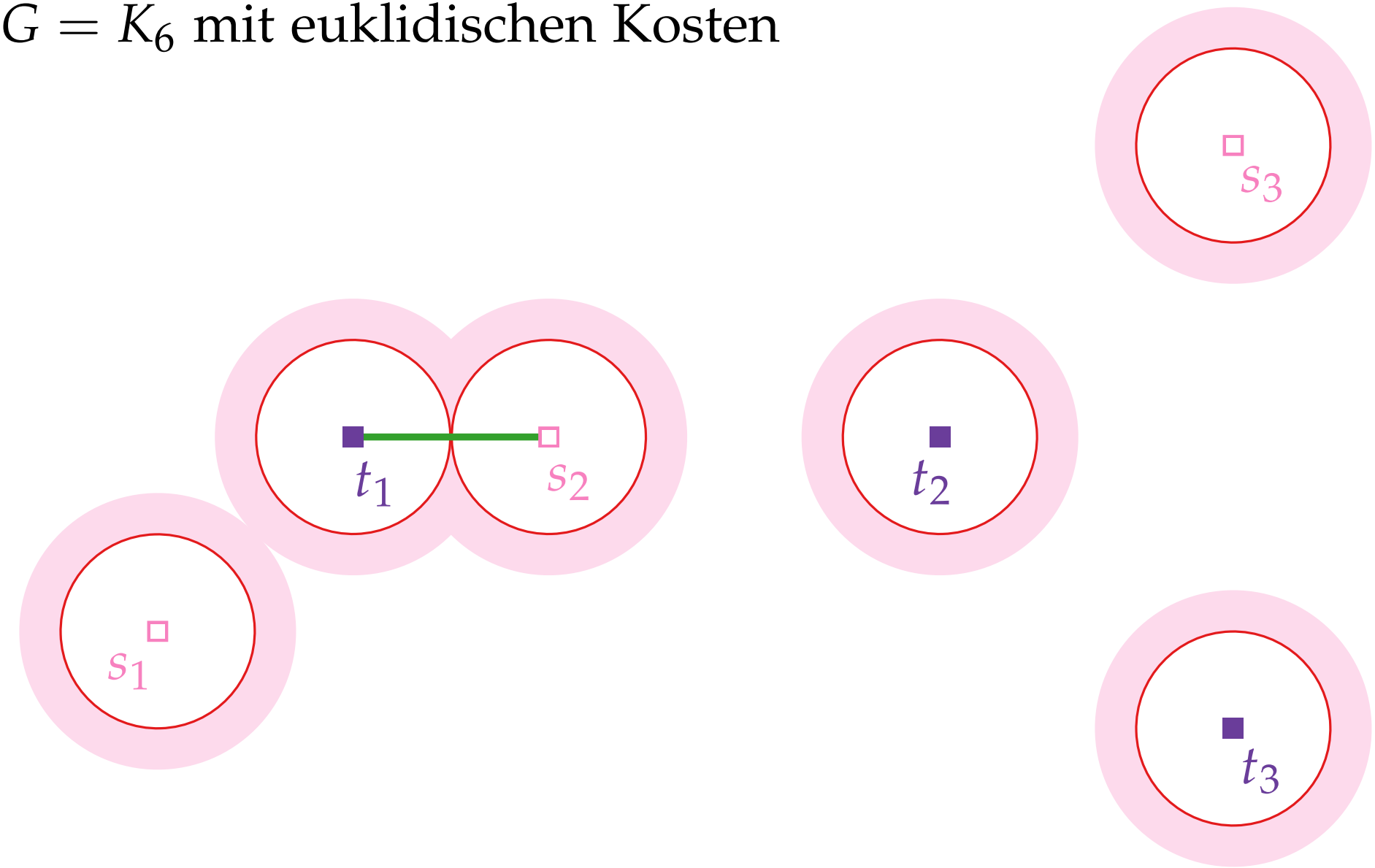
Visualisierung

$G = K_6$ mit euklidischen Kosten



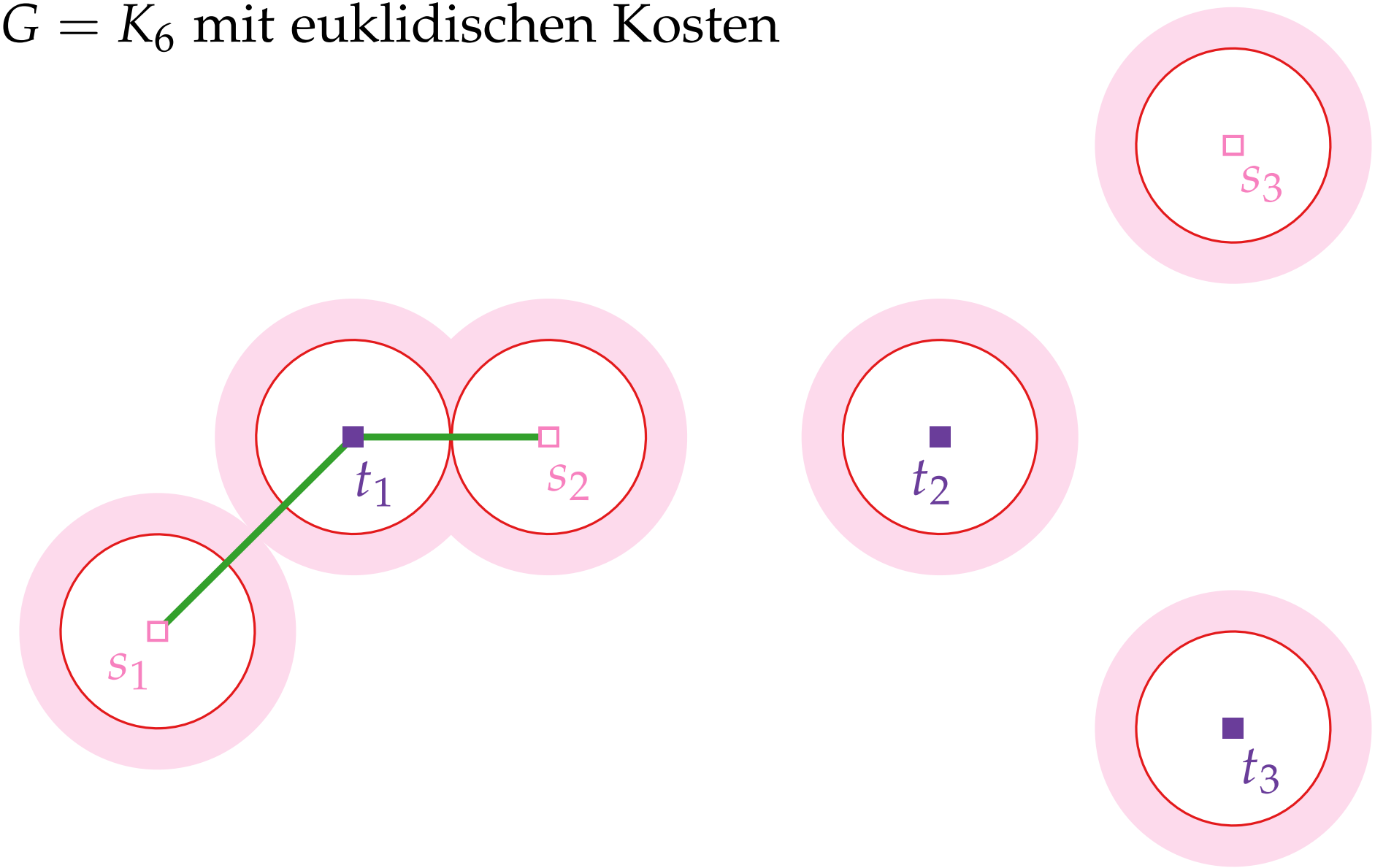
Visualisierung

$G = K_6$ mit euklidischen Kosten



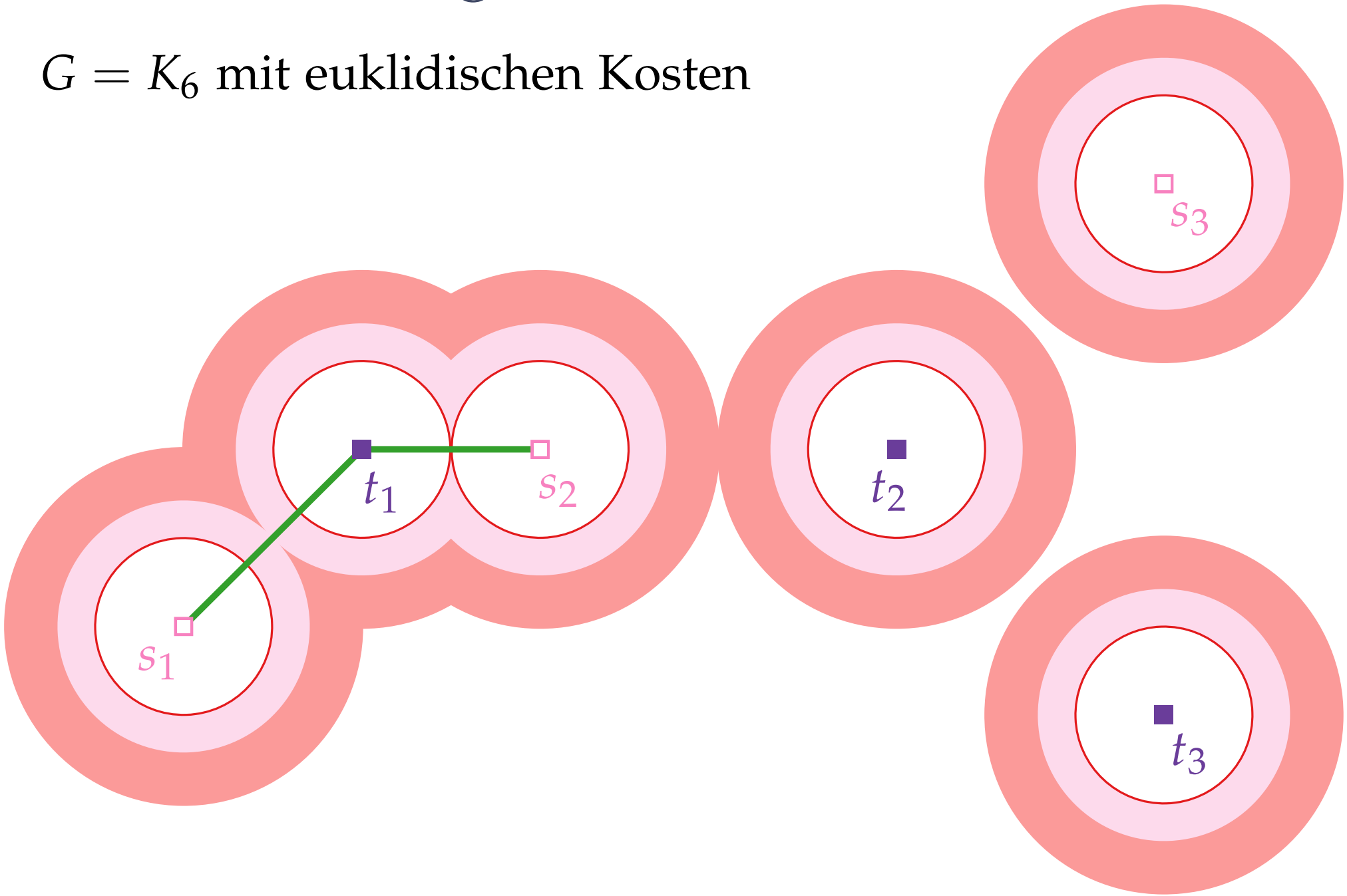
Visualisierung

$G = K_6$ mit euklidischen Kosten



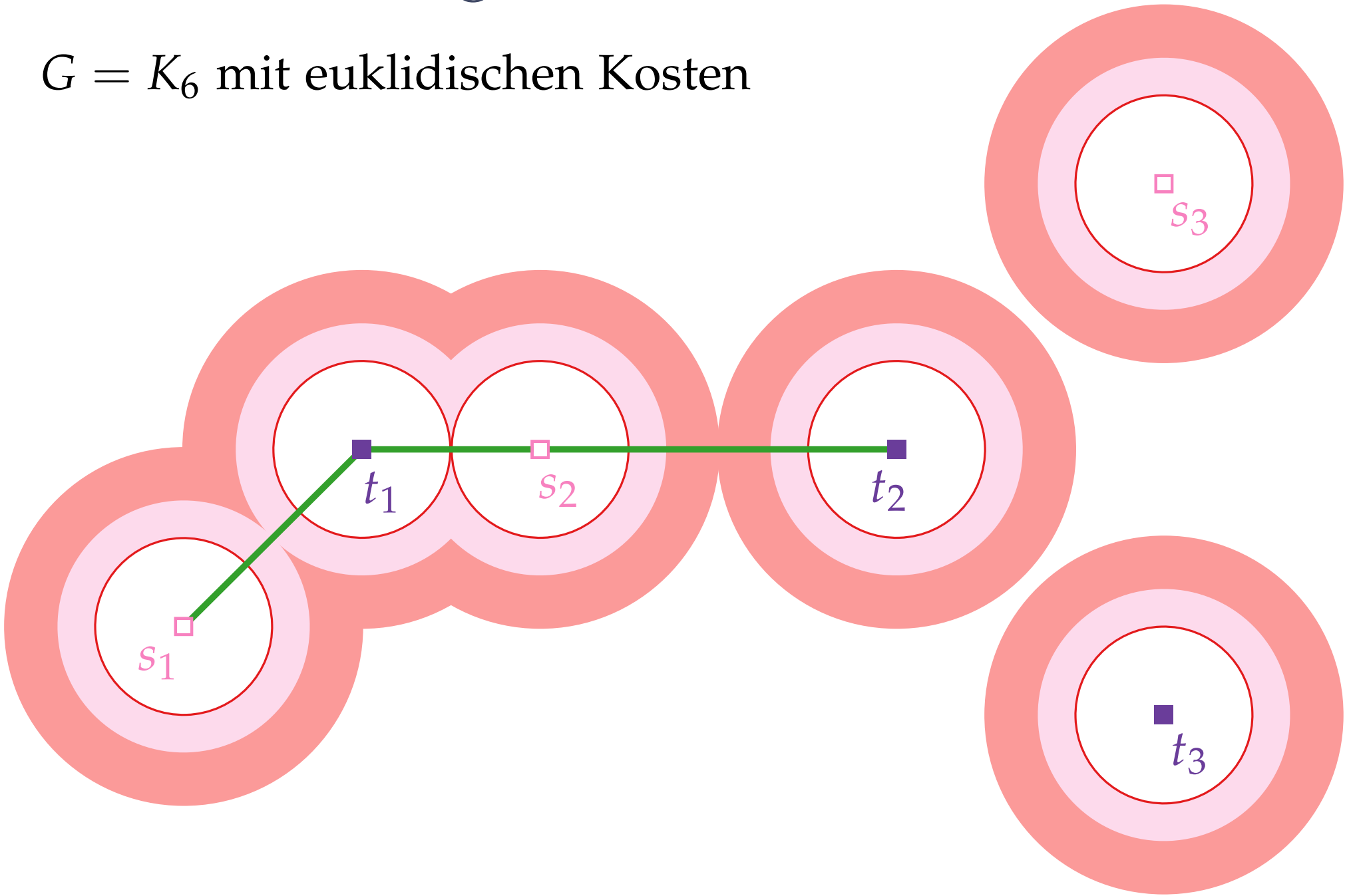
Visualisierung

$G = K_6$ mit euklidischen Kosten



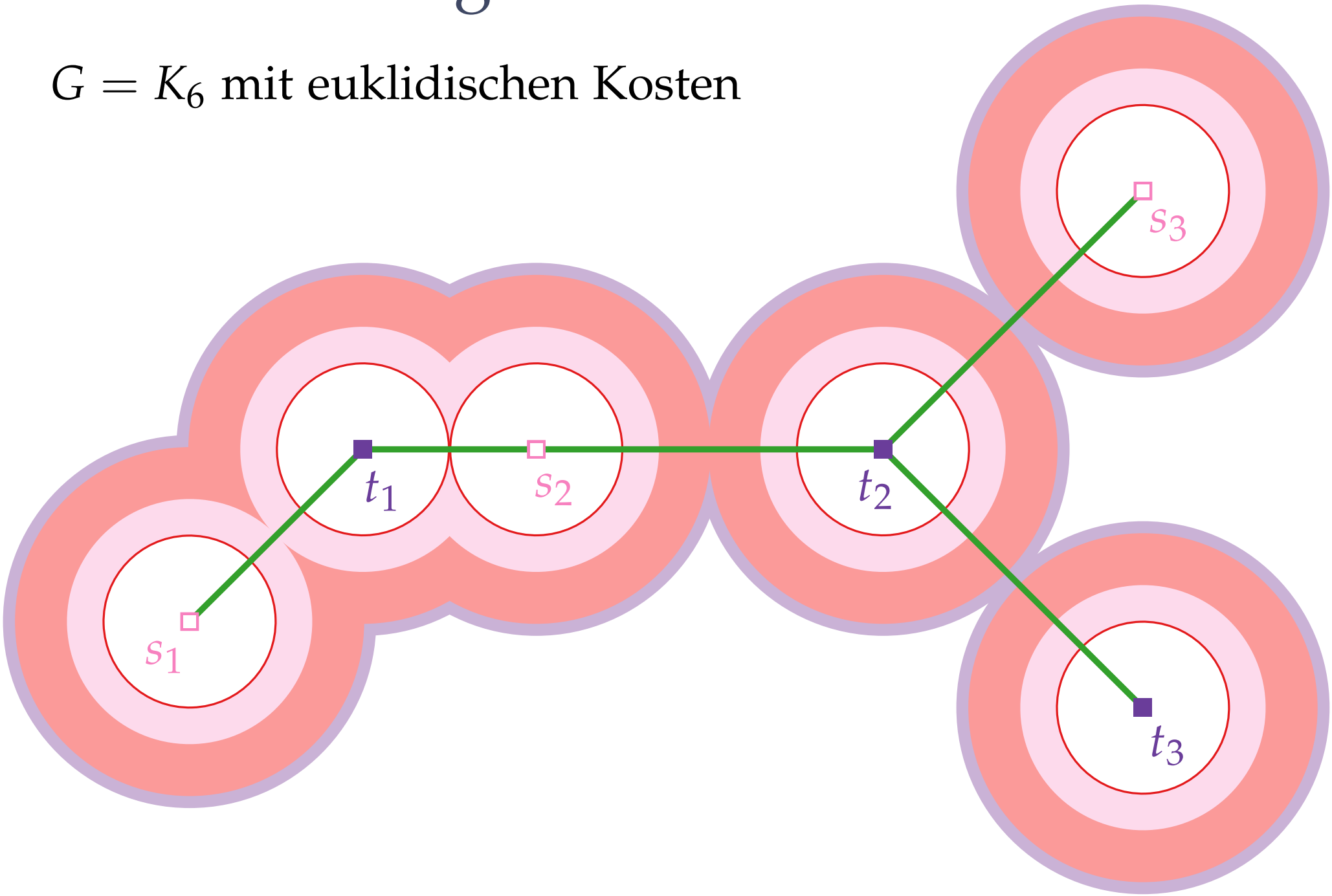
Visualisierung

$G = K_6$ mit euklidischen Kosten



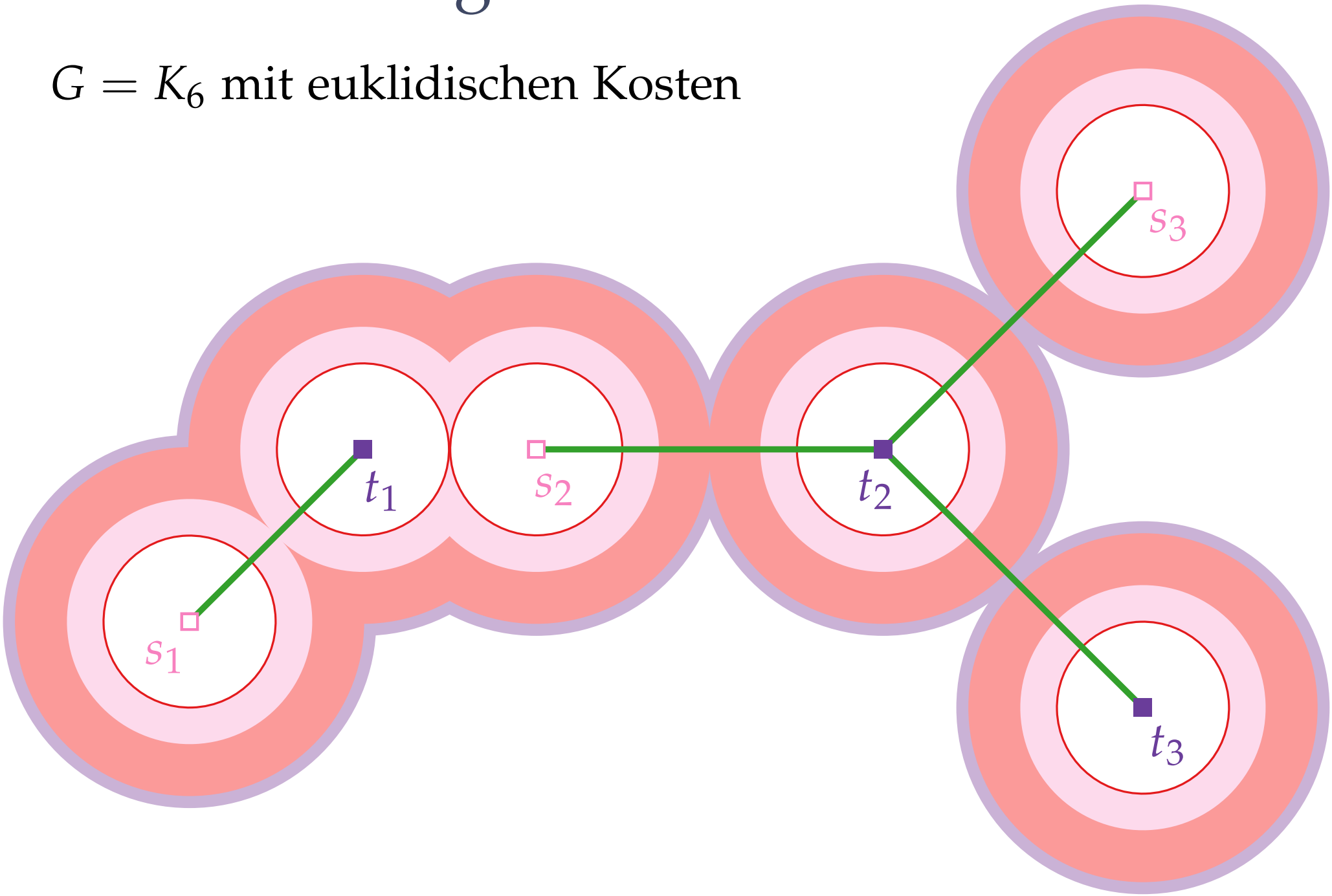
Visualisierung

$G = K_6$ mit euklidischen Kosten



Visualisierung

$G = K_6$ mit euklidischen Kosten



Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

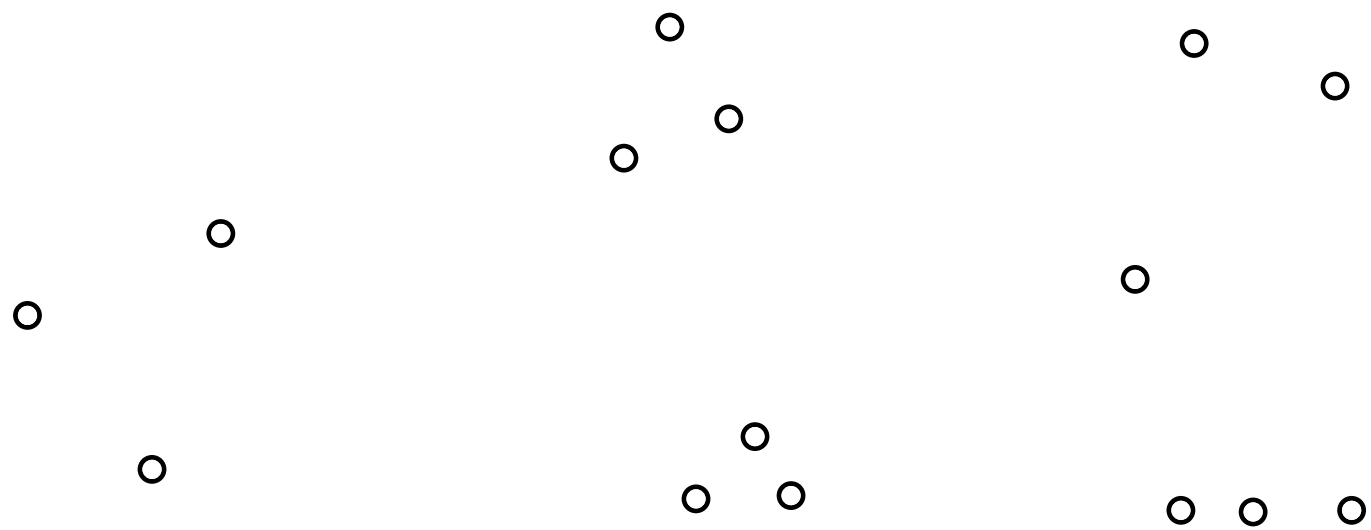
Beweisidee. Zunächst Intuition…

Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

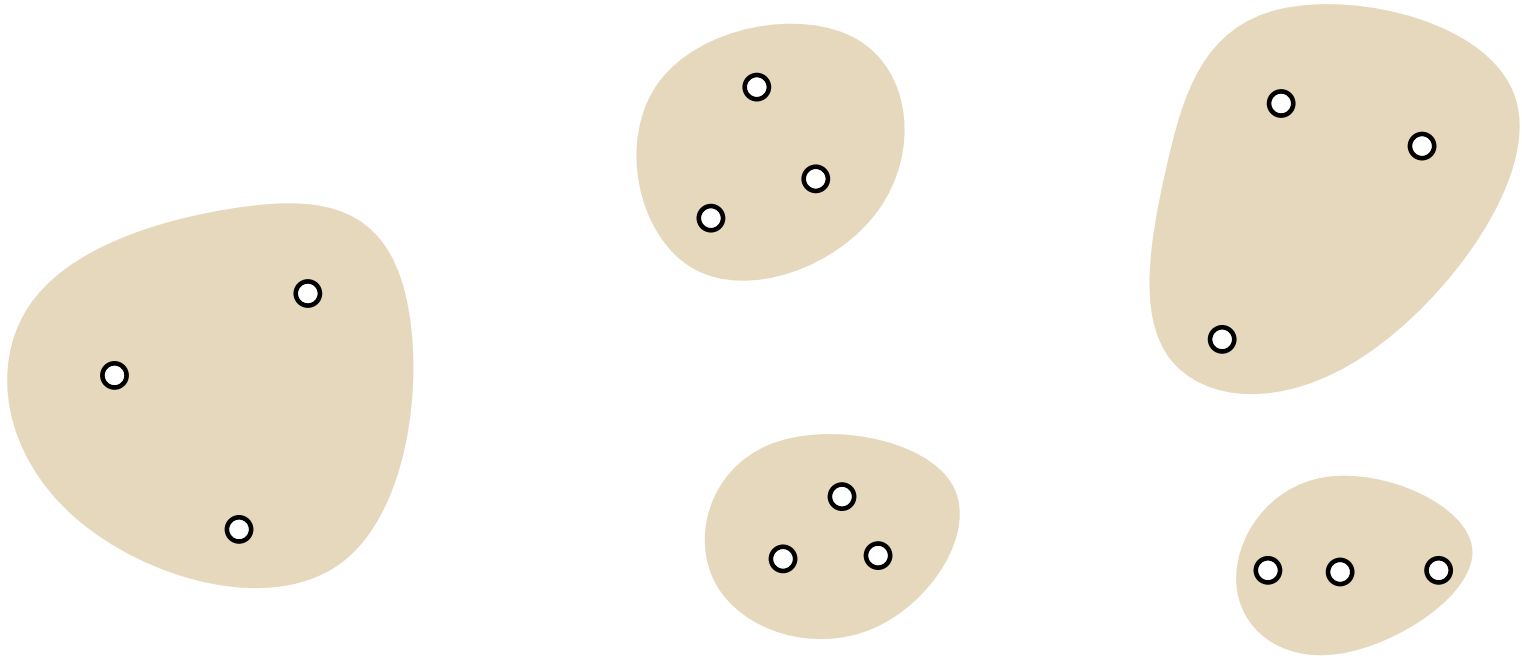


Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

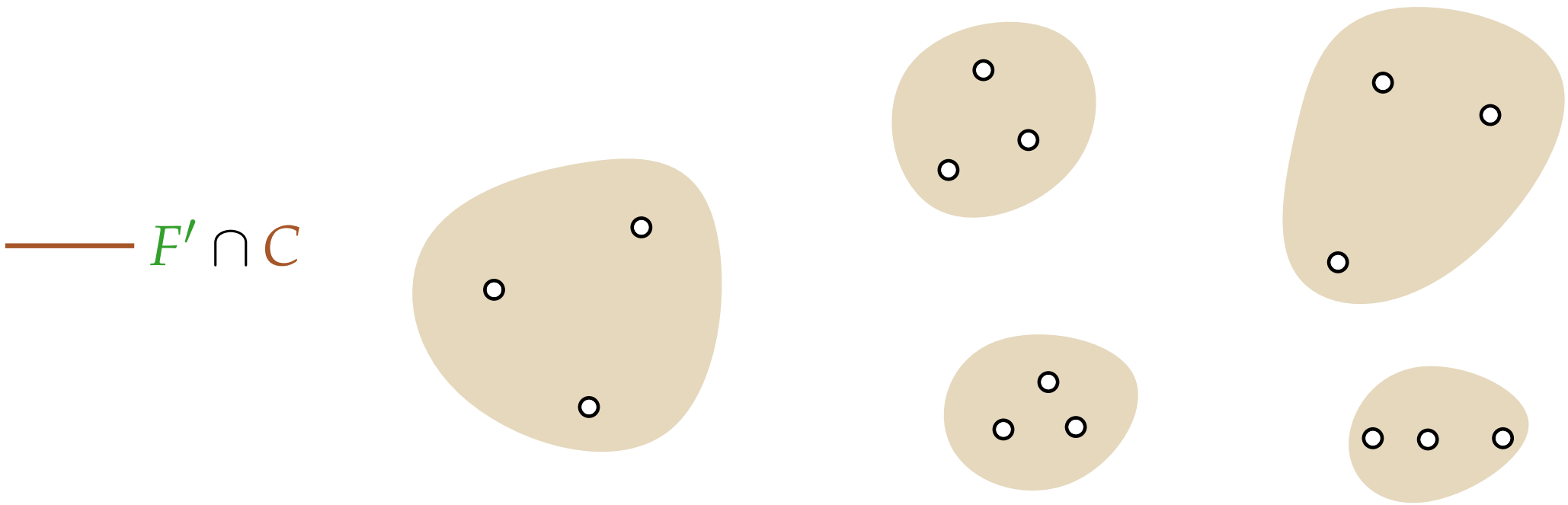


Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

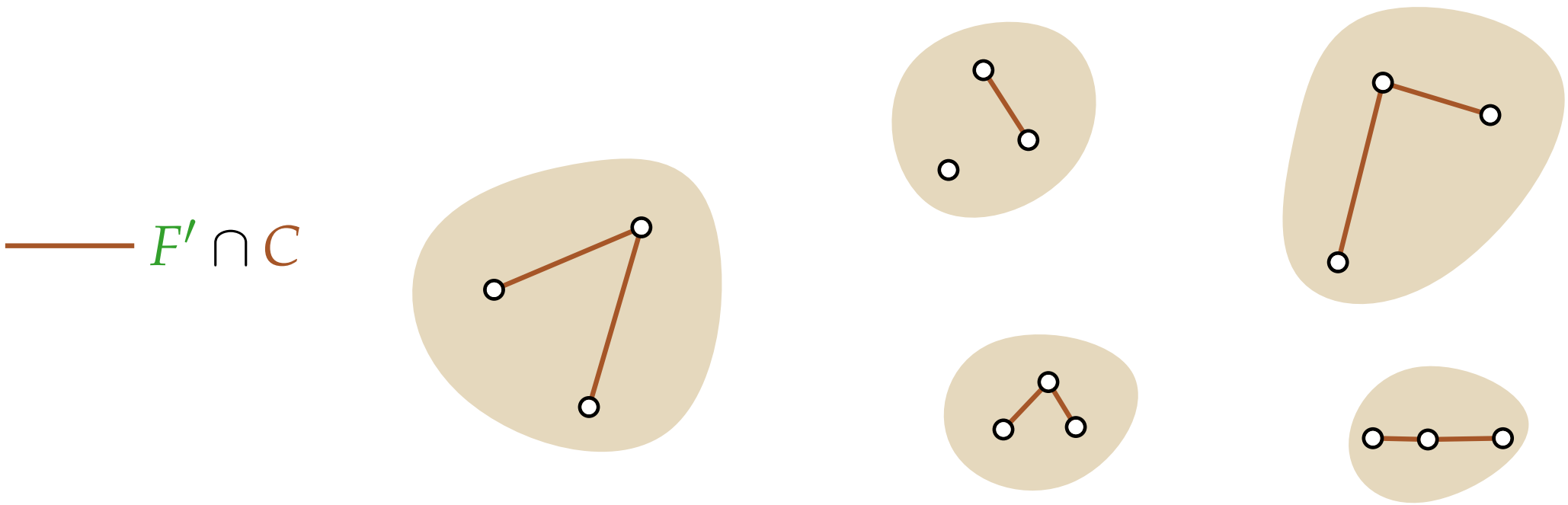


Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

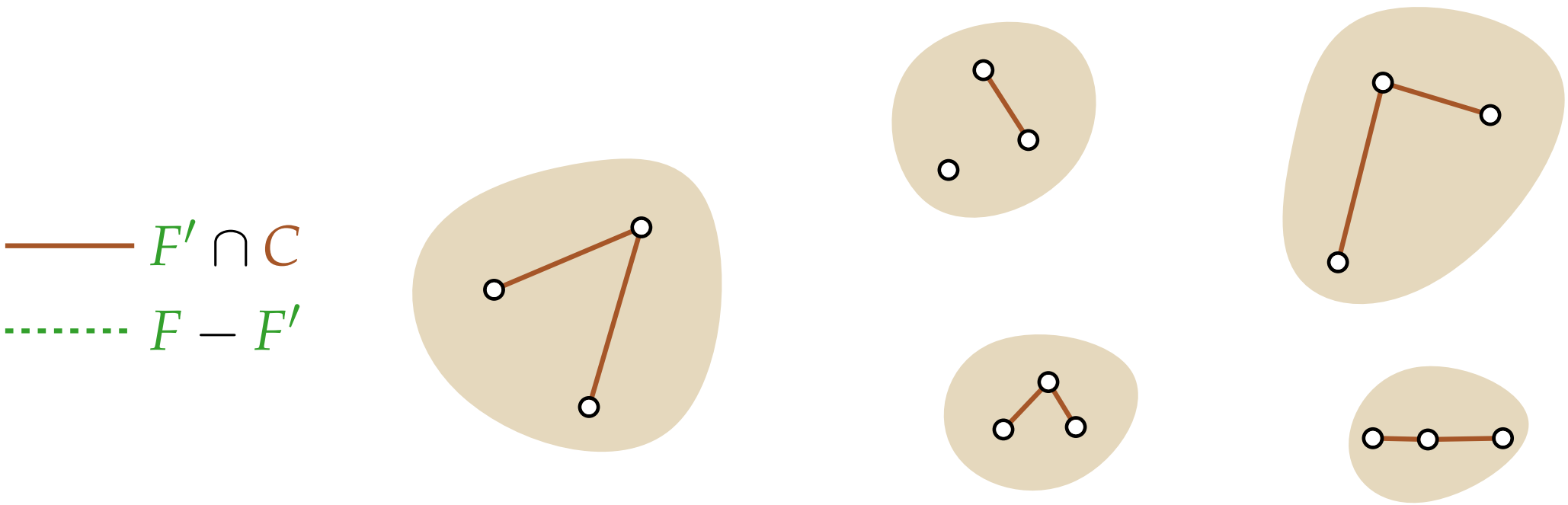


Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

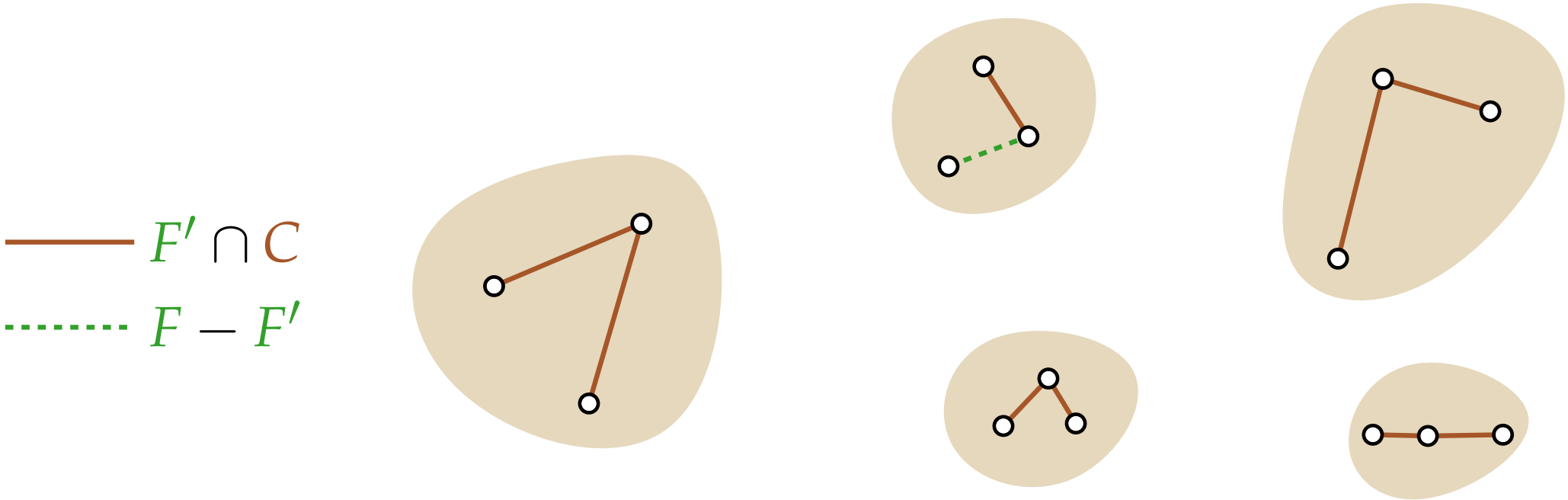


Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

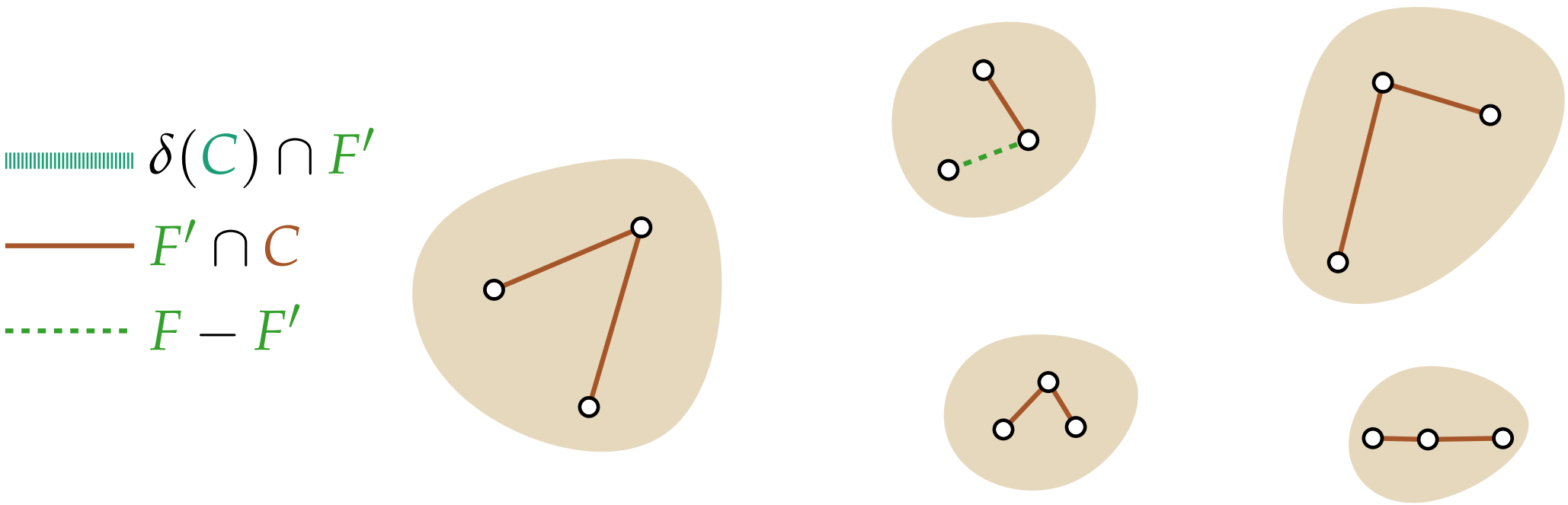


Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

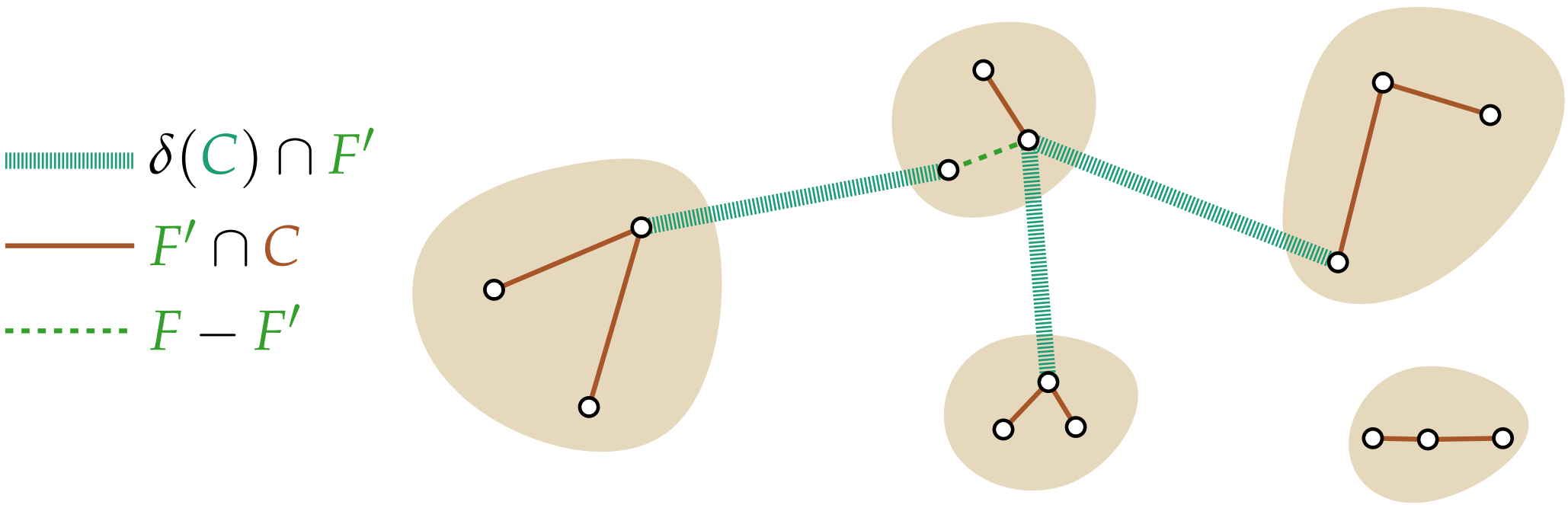


Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…



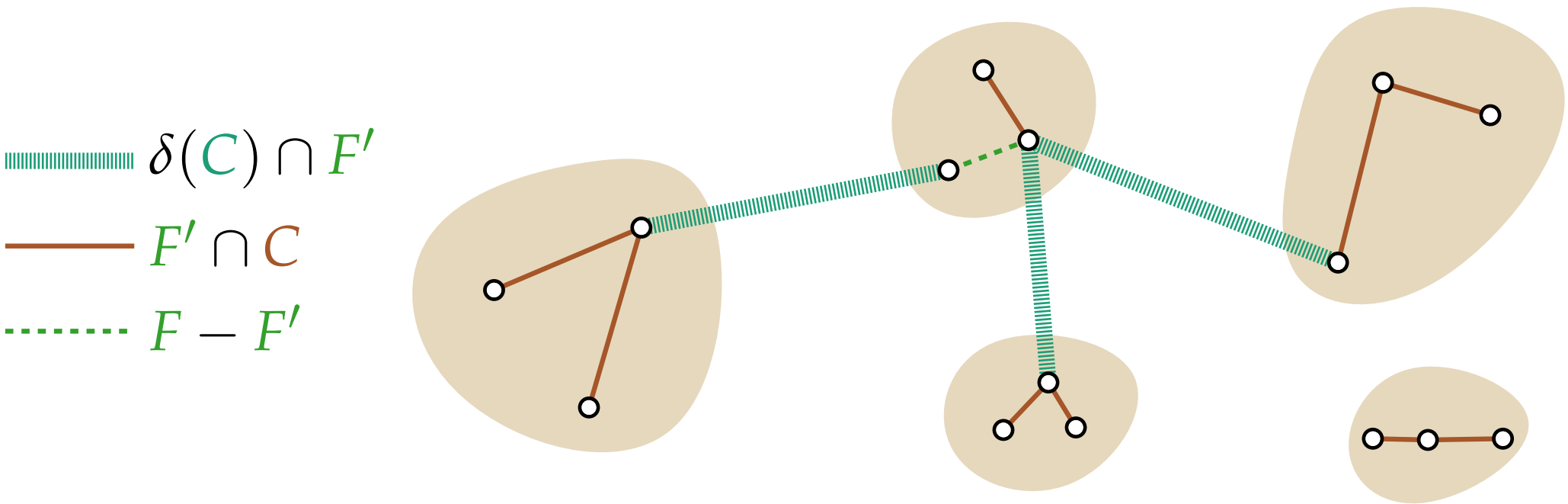
Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

Durchschnittsgrad der ZK C bezüglich F ist



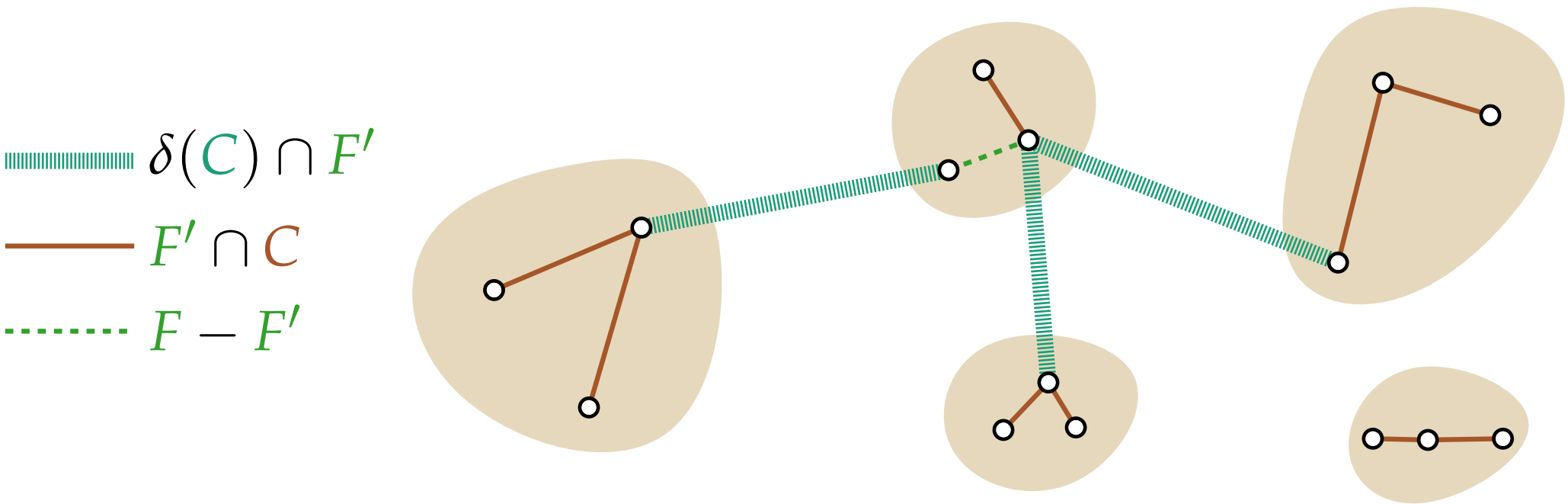
Struktur-Lemma

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition…

Durchschnittsgrad der ZK C bezüglich F ist ≤ 2!



Struktur-Lemma

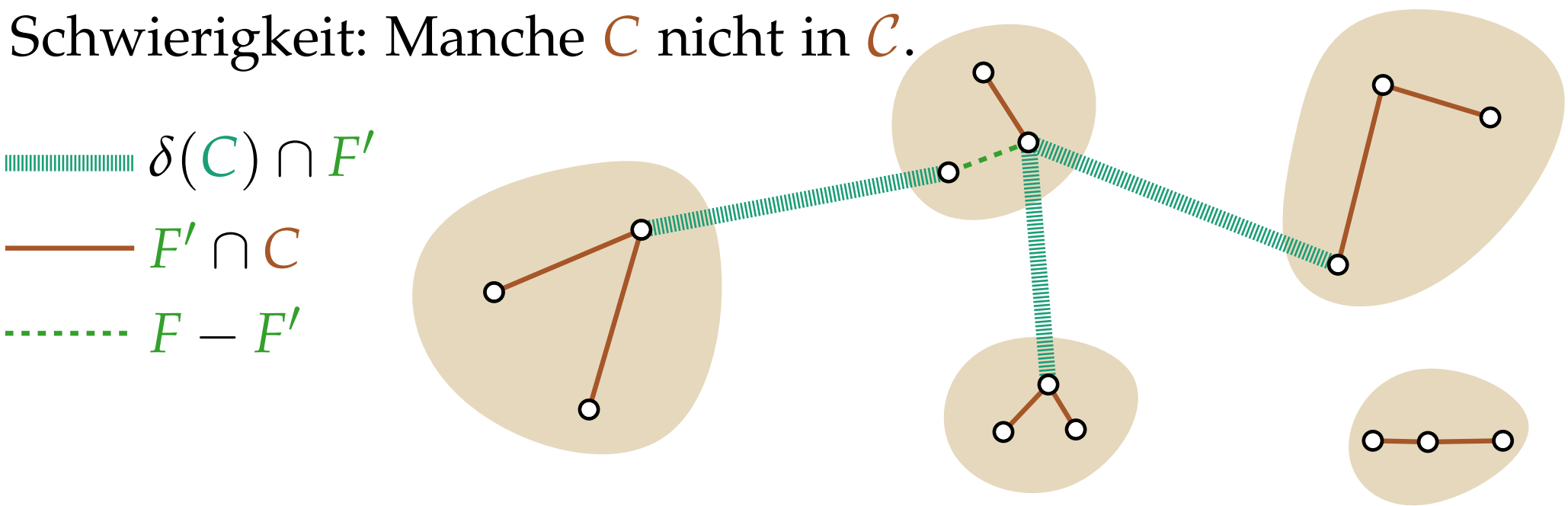
Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweisidee. Zunächst Intuition...

Durchschnittsgrad der ZK C bezüglich F ist ≤ 2!

Schwierigkeit: Manche C nicht in $\mathcal{C}$.



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$.

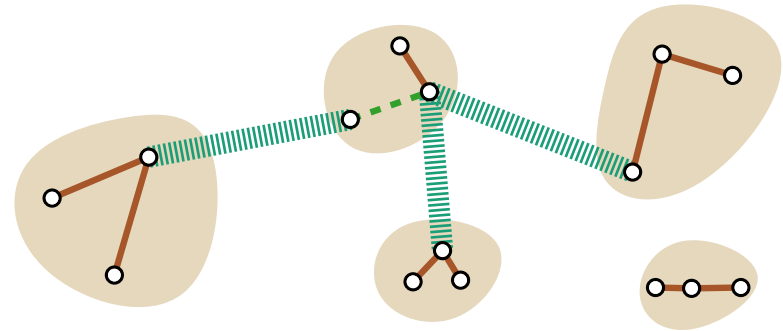
Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$.



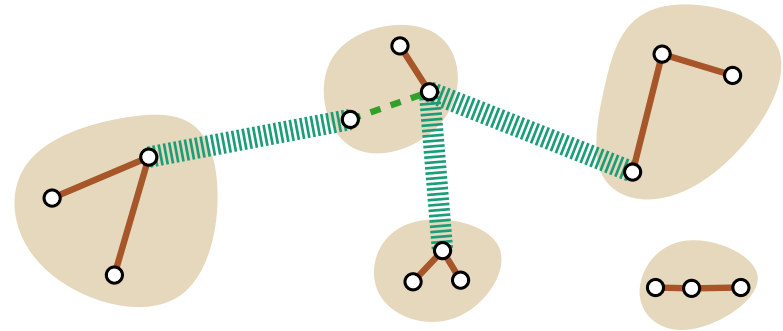
Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.



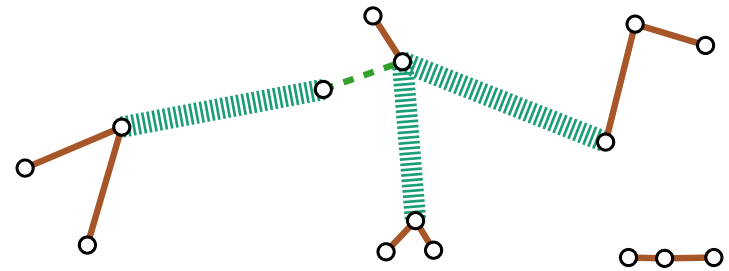
Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.



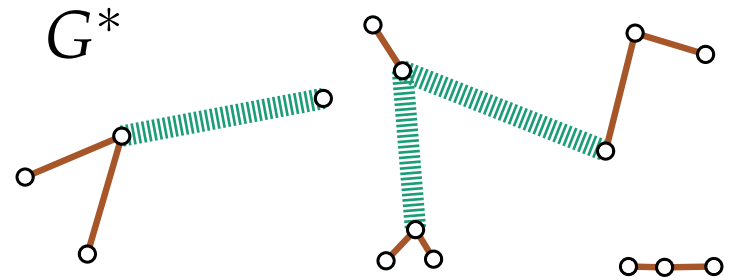
Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.



Beweis des Struktur-Lemmas

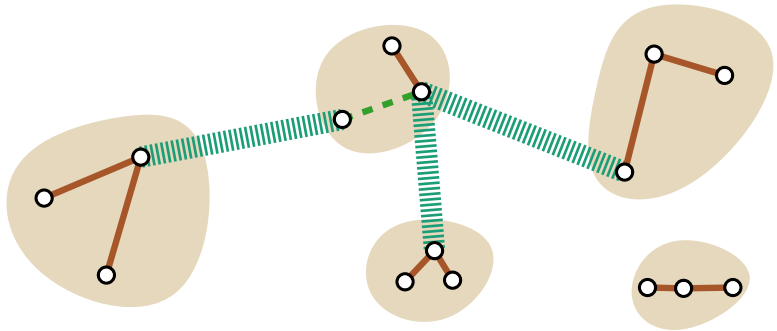
Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.



Beweis des Struktur-Lemmas

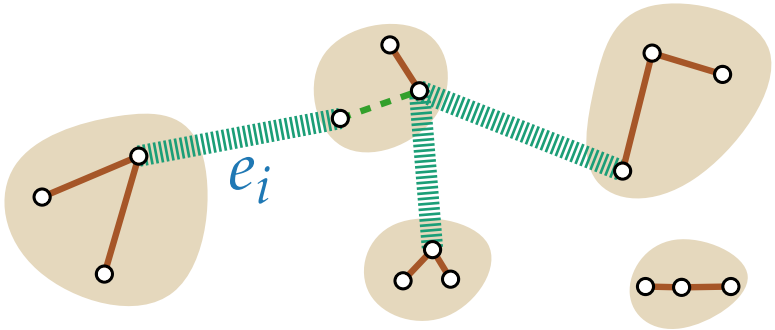
Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

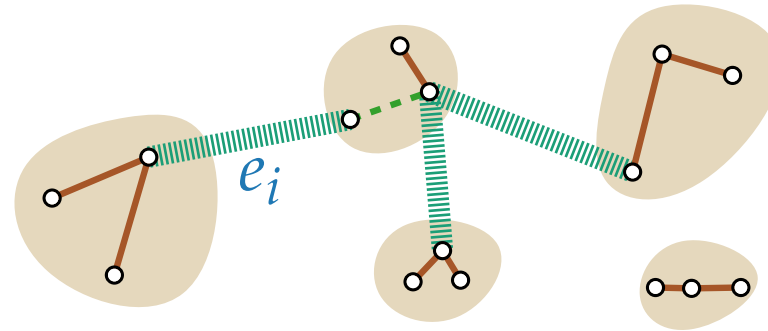
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

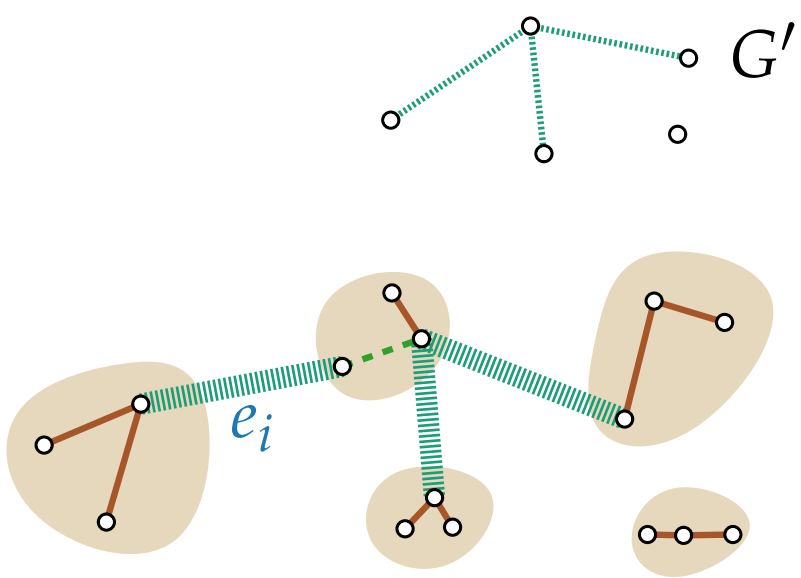
$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}|\,.$$

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

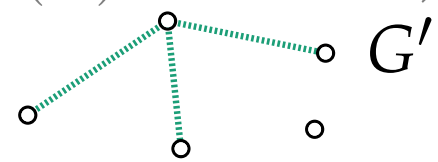
Beweis.

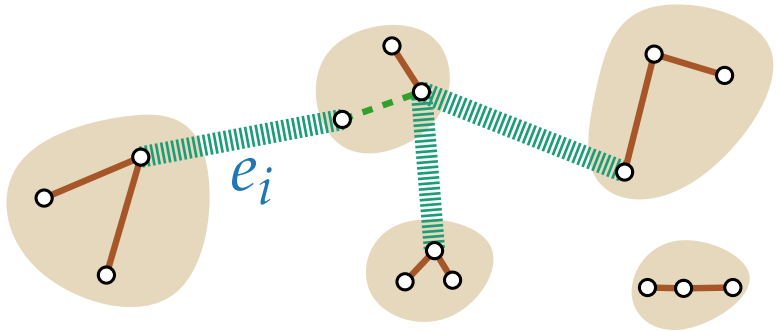
Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)





Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

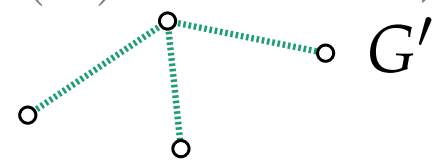
Beweis.

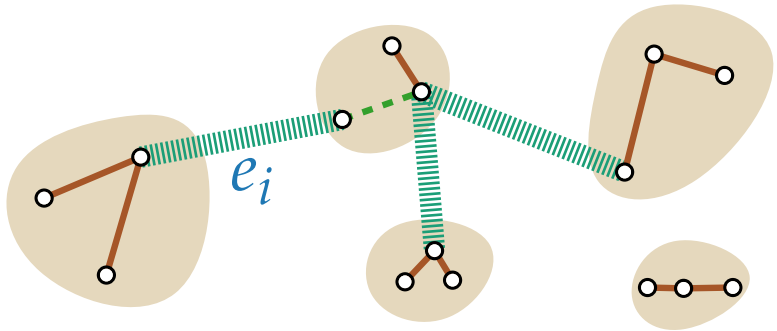
Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)





Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

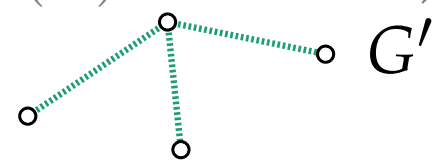
Sei $G^* = (V, F')$. G^* ist ein Wald.

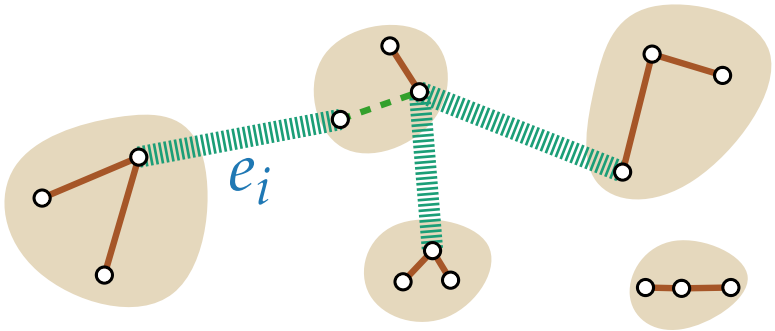
Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)

Beh. G' ist ein Wald.





Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

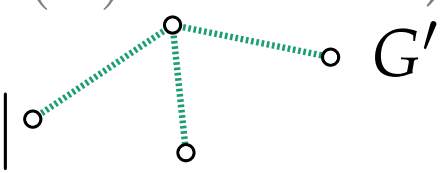
Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

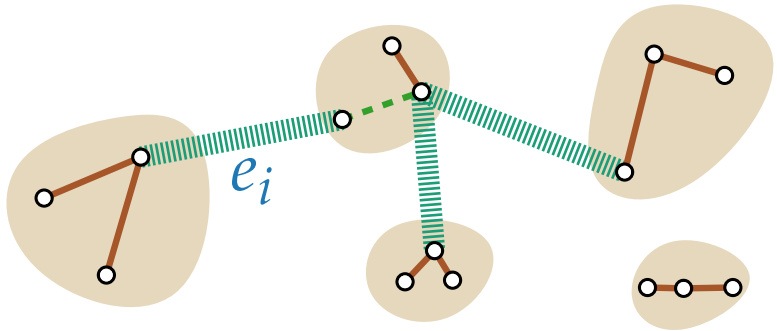
Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)

Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$





Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

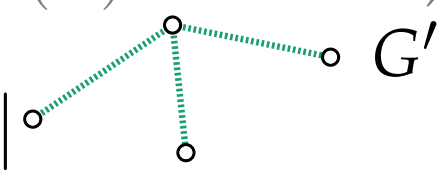
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

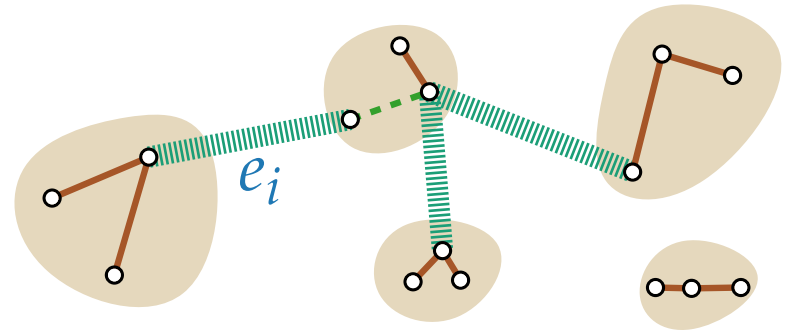
(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

$$= 2|E(G')|$$



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

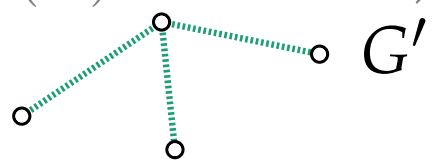
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

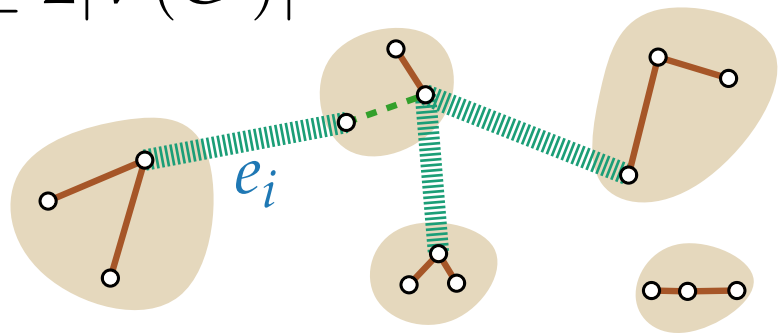
(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

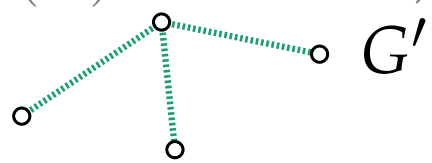
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

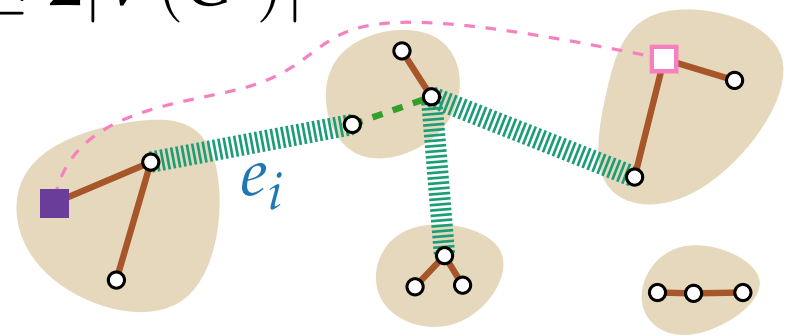
(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

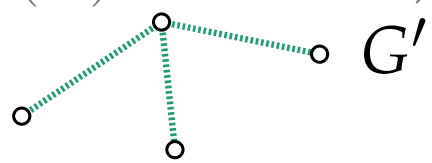
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

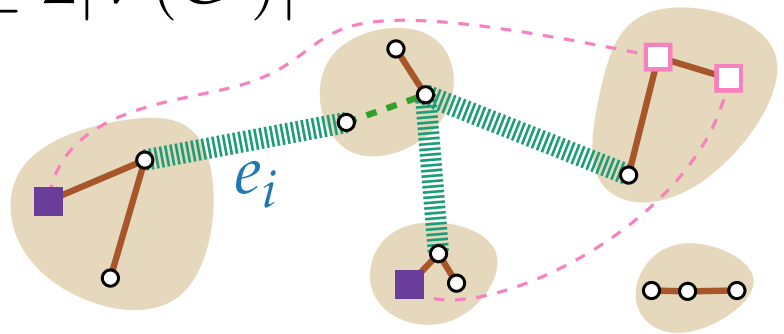
(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

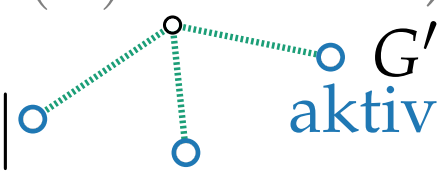
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

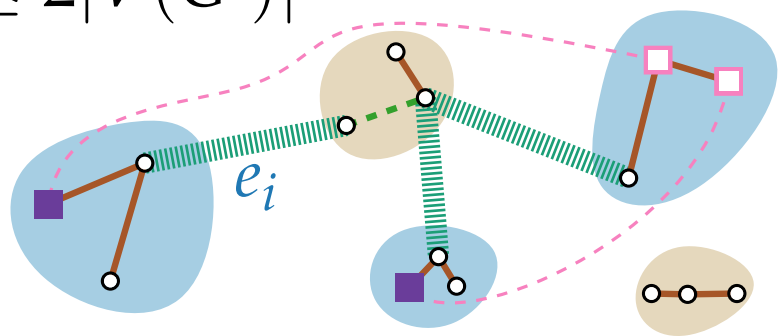
(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

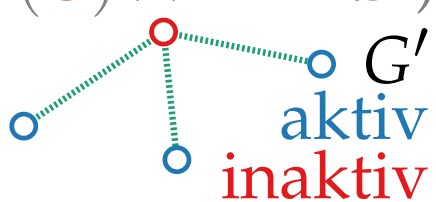
Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

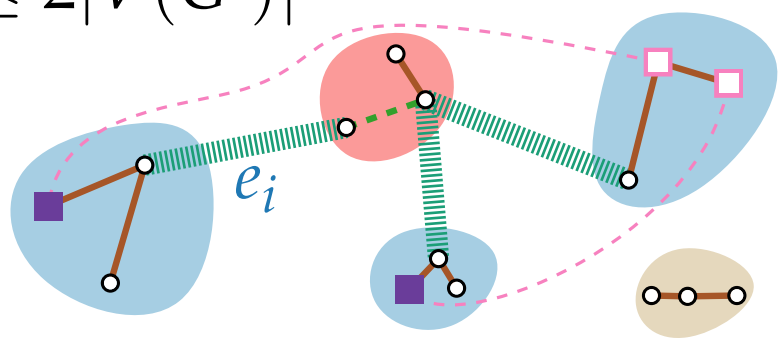
(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)

Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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





Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

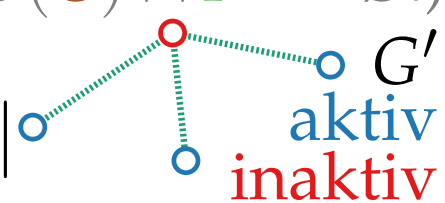
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)

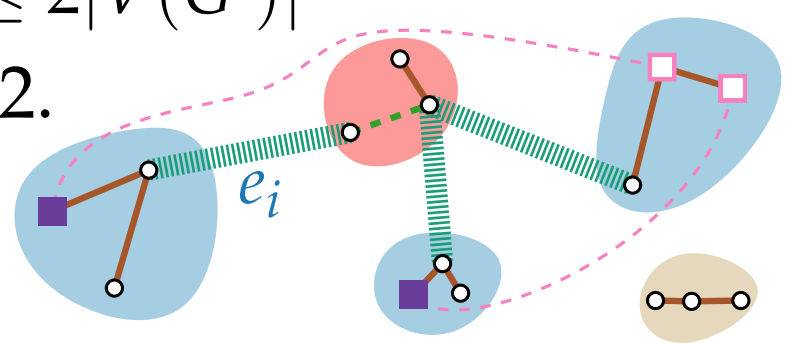


Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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

Beh. Inaktive Knoten haben Grad ≥ 2.



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

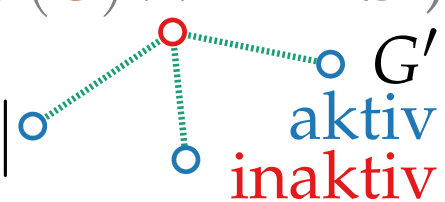
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



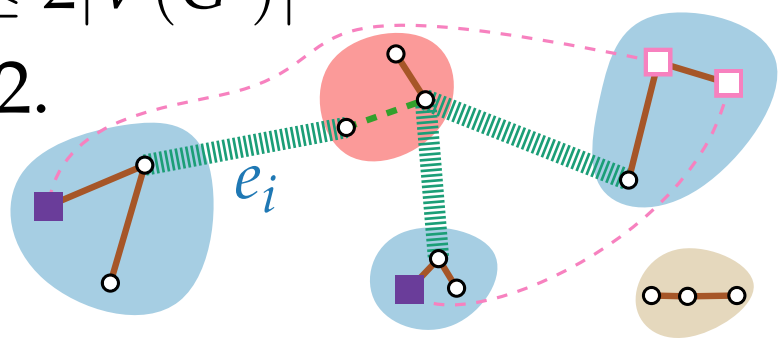
Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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

Beh. Inaktive Knoten haben Grad ≥ 2.

Dann, $\sum_{v \text{ aktiv}} |\deg_{G'}(v)| \leq$



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

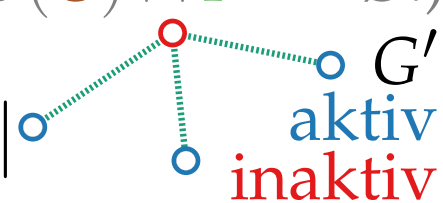
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



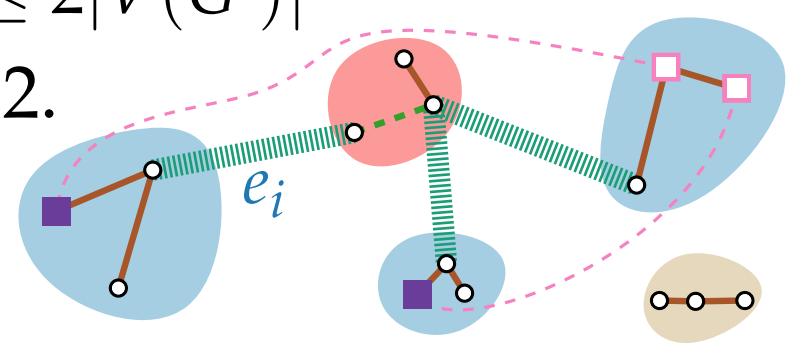
Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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

Beh. Inaktive Knoten haben Grad ≥ 2.

Dann, $\sum_{v \text{ aktiv}} |\deg_{G'}(v)| \leq$
$2 \cdot |V(G')| - 2 \cdot \#(\text{inaktiv})$



Beweis des Struktur-Lemmas

Lemma 1. In jeder Iteration des Algorithmus gilt

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

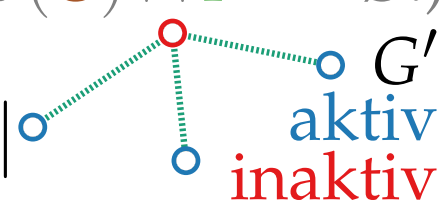
Beweis.

Sei $G^* = (V, F')$. G^* ist ein Wald.

Betrachte die i. Iteration, in der e_i zu F hinzugefügt wird.

Kontrahiere jede ZK C zu einem Knoten $\rightsquigarrow G'$.

(Ignoriere alle ZK C mit $\delta(C) \cap F' = \emptyset$.)



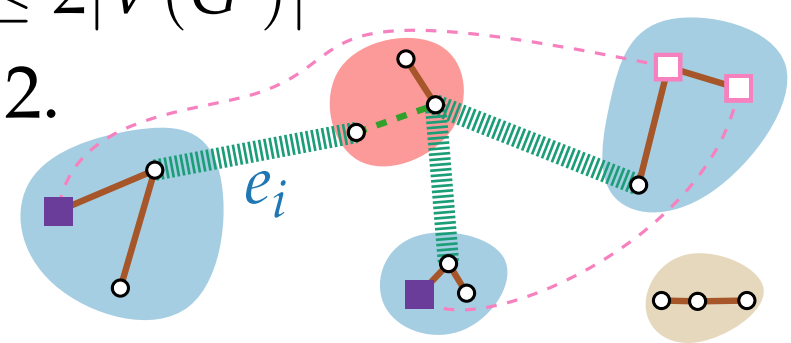
Beh. G' ist ein Wald.

Beachte: $\sum_{C \text{ ZK}} |\delta(C) \cap F'| = \sum_{v \in V(G')} |\deg_{G'}(v)|$

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

Beh. Inaktive Knoten haben Grad ≥ 2.

Dann, $\sum_{v \text{ aktiv}} |\deg_{G'}(v)| \leq$
$2 \cdot |V(G')| - 2 \cdot \#(\text{inaktiv}) = 2|\mathcal{C}|$. $\square$



Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

Wie zuvor

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

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

Wie zuvor

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

Wir zeigen per Induktion über die Anzahl der Iterationen des Algorithmus, dass

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

Wie zuvor

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

Wir zeigen per Induktion über die Anzahl der Iterationen des Algorithmus, dass

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

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

Wie zuvor

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

Wir zeigen per Induktion über die Anzahl der Iterationen des Algorithmus, dass

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

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

Wie zuvor

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

Wir zeigen per Induktion über die Anzahl der Iterationen des Algorithmus, dass

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

Daraus folgt die Behauptung des Satzes.

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Angenommen $(*)$ gilt zu Beginn einer Iteration.

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Angenommen $(*)$ gilt zu Beginn einer Iteration.

In dieser Iteration erhöhen wir y_C für alle $C \in \mathcal{C}$ um den gleichen Betrag, sagen wir um $\varepsilon \geq 0$.

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Angenommen $(*)$ gilt zu Beginn einer Iteration.

In dieser Iteration erhöhen wir y_C für alle $C \in \mathcal{C}$ um den gleichen Betrag, sagen wir um $\varepsilon \geq 0$.

Das erhöht die linke Seite von $(*)$ um

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Angenommen $(*)$ gilt zu Beginn einer Iteration.

In dieser Iteration erhöhen wir y_C für alle $C \in \mathcal{C}$ um den gleichen Betrag, sagen wir um $\varepsilon \geq 0$.

Das erhöht die linke Seite von $(*)$ um $\varepsilon \sum_{C \in \mathcal{C}} |F' \cap \delta(C)|$

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Angenommen $(*)$ gilt zu Beginn einer Iteration.

In dieser Iteration erhöhen wir y_C für alle $C \in \mathcal{C}$ um den gleichen Betrag, sagen wir um $\varepsilon \geq 0$.

Das erhöht die linke Seite von $(*)$ um $\varepsilon \sum_{C \in \mathcal{C}} |F' \cap \delta(C)|$ und die rechte Seite um

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Angenommen $(*)$ gilt zu Beginn einer Iteration.

In dieser Iteration erhöhen wir y_C für alle $C \in \mathcal{C}$ um den gleichen Betrag, sagen wir um $\varepsilon \geq 0$.

Das erhöht die linke Seite von $(*)$ um $\varepsilon \sum_{C \in \mathcal{C}} |F' \cap \delta(C)|$ und die rechte Seite um $2\varepsilon|\mathcal{C}|$.

Analyse

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Beweis.

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

Beziehung gilt trivialerweise zu Begin, da $y_S = 0$ für alle S.

Angenommen $(*)$ gilt zu Beginn einer Iteration.

In dieser Iteration erhöhen wir y_C für alle $C \in \mathcal{C}$ um den gleichen Betrag, sagen wir um $\varepsilon \geq 0$.

Das erhöht die linke Seite von $(*)$ um $\varepsilon \sum_{C \in \mathcal{C}} |F' \cap \delta(C)|$ und die rechte Seite um $2\varepsilon|\mathcal{C}|$.

Nach Lemma 1 gilt $(*)$ somit auch nach der Iteration. $\square$

Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Zusammenfassung

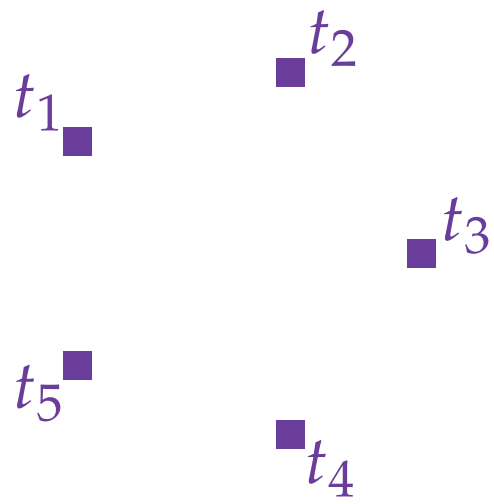
Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Analyse scharf?

Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

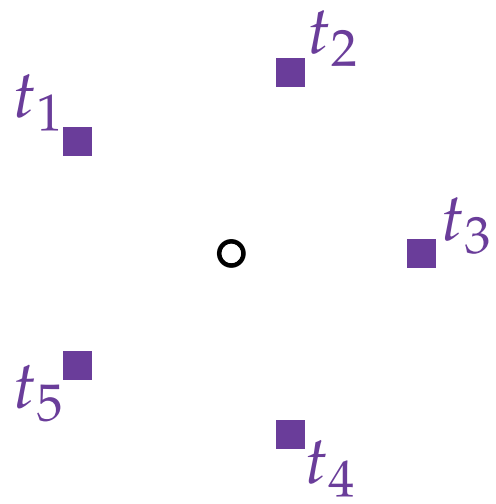
Analyse scharf?



Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

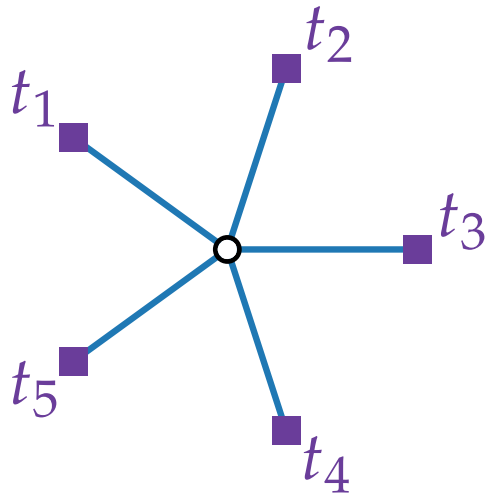
Analyse scharf?



Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

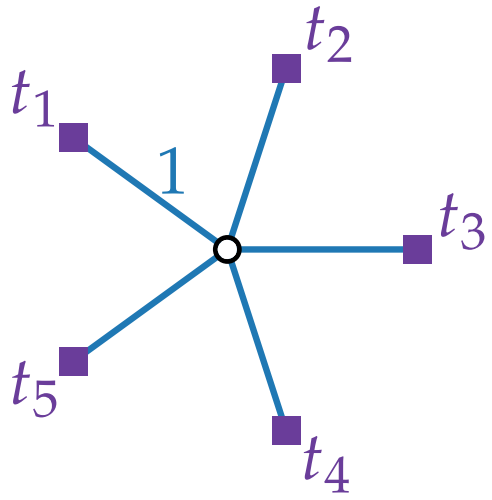
Analyse scharf?



Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

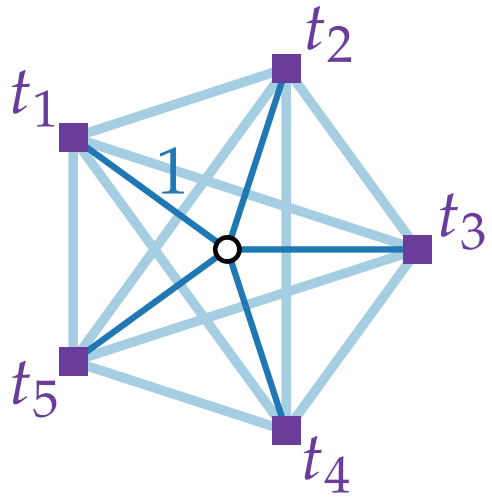
Analyse scharf?



Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

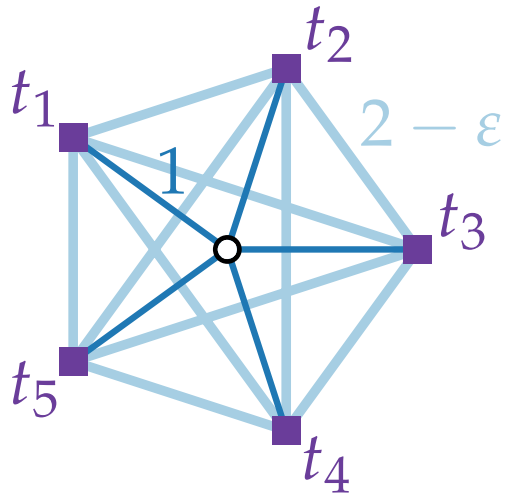
Analyse scharf?



Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

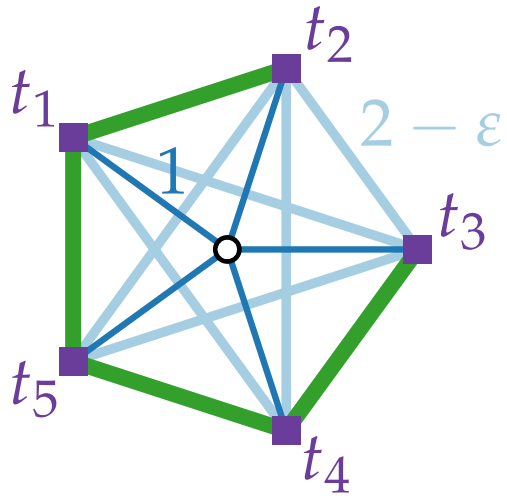
Analyse scharf?



Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Analyse scharf?

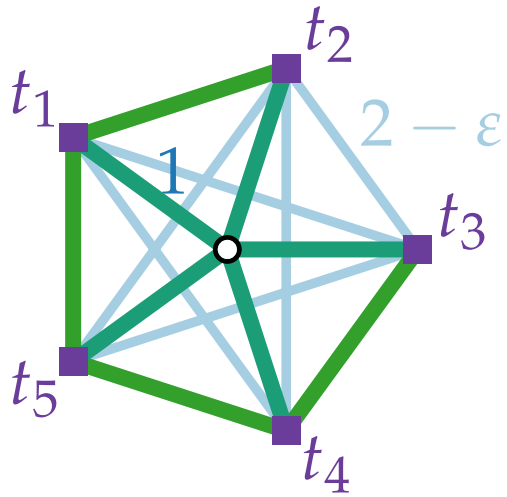


$$\text{ALG} = (2-\varepsilon)(n-1)$$

Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Analyse scharf?

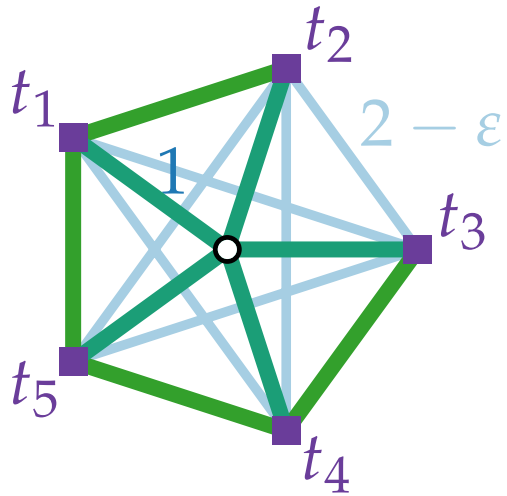


$\text{ALG} = (2-\varepsilon)(n-1)$
$\text{OPT} = n$

Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Analyse scharf?



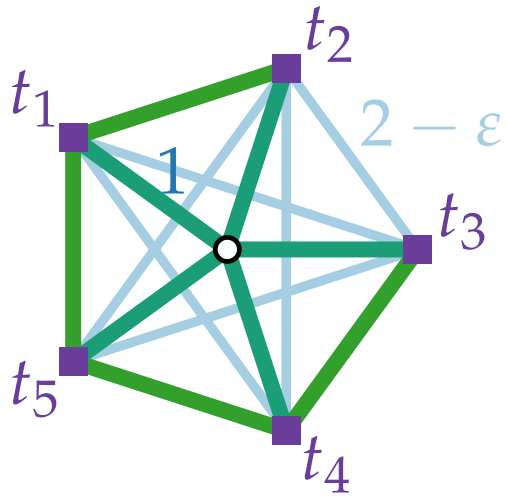
$\text{ALG} = (2-\varepsilon)(n-1)$
$\text{OPT} = n$

besser?

Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Analyse scharf?



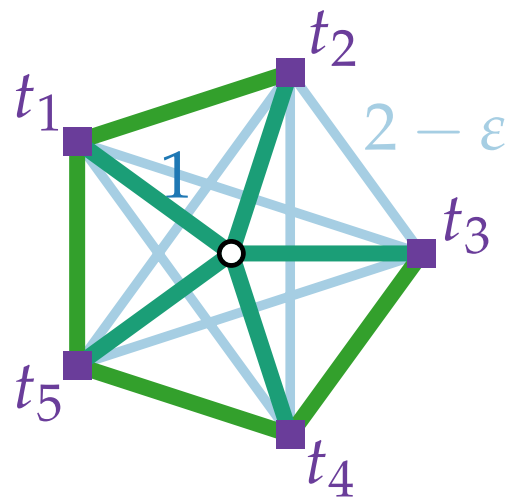
$\text{ALG} = (2-\varepsilon)(n-1)$
$\text{OPT} = n$

besser?
Es ist kein besserer Approximationsfaktor bekannt.

Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Analyse scharf?



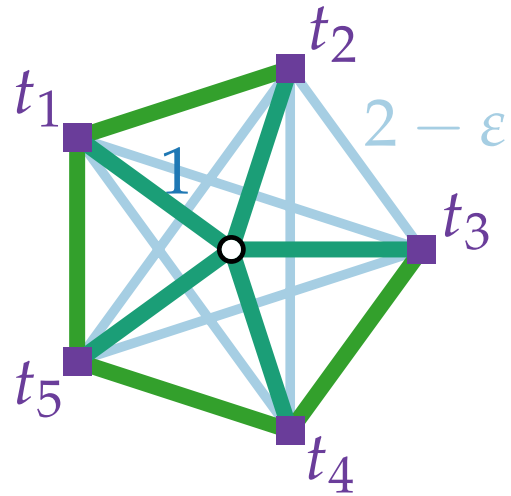
$\text{ALG} = (2-\varepsilon)(n-1)$
$\text{OPT} = n$

besser?
Es ist kein besserer Approximationsfaktor bekannt.
Das Integrality Gap beträgt $2 - 1/n$.

Zusammenfassung

Satz 1. Der synchronisierte Primal-Dual-Algorithm liefert eine 2-Approximation für das SteinerWald Problem.

Analyse scharf?



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

besser?
Es ist kein besserer Approximationsfaktor bekannt.
Das Integrality Gap beträgt $2 - 1/n$.
SteinerWald lässt sich nicht mit Faktor $1,0074$ approximieren (außer P=NP) [Thimm '03]