Karlsruher Institut für Technologie

Algorithmen II
Vorlesung am 07.11.2013

Minimale Schnitte in Graphen

INSTITUT FÜR THEORETISCHE INFORMATIK · PROF. DR. DOROTHEA WAGNER

Schnitte minimalen Gewichts: MinCut

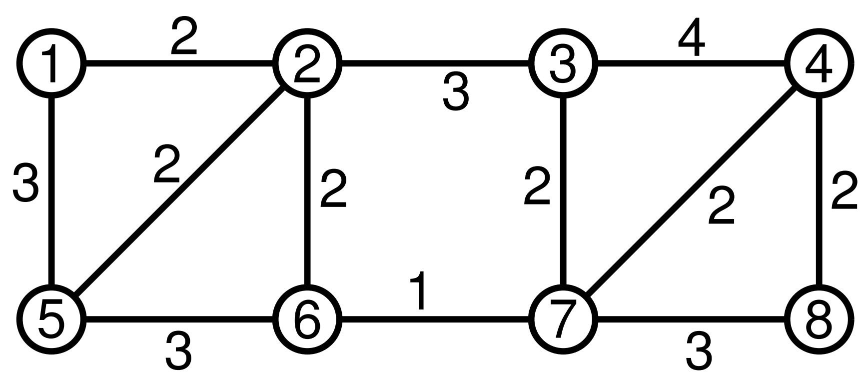
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem – MinCut

Problem: MinCut
Gegeben sei ein Graph $G = (V, E)$ mit Kantengewichtsfunktion $c\colon E \longrightarrow \mathbb{R}_0^+$.
Finde einen *nichttrivialen Schnitt* $(S, V \setminus S)$ *minimalen Gewichts* in G, d.h. finde $S \subseteq V$ mit $\emptyset \neq S \neq V$, sodass

$$c(S, V \setminus S) := \sum_{\substack{\{u,v\} \in E, \\ u \in S, \\ v \in V \setminus S}} c(\{u, v\})$$

minimal wird. $(S, V \setminus S)$ wird *minimaler Schnitt* genannt.



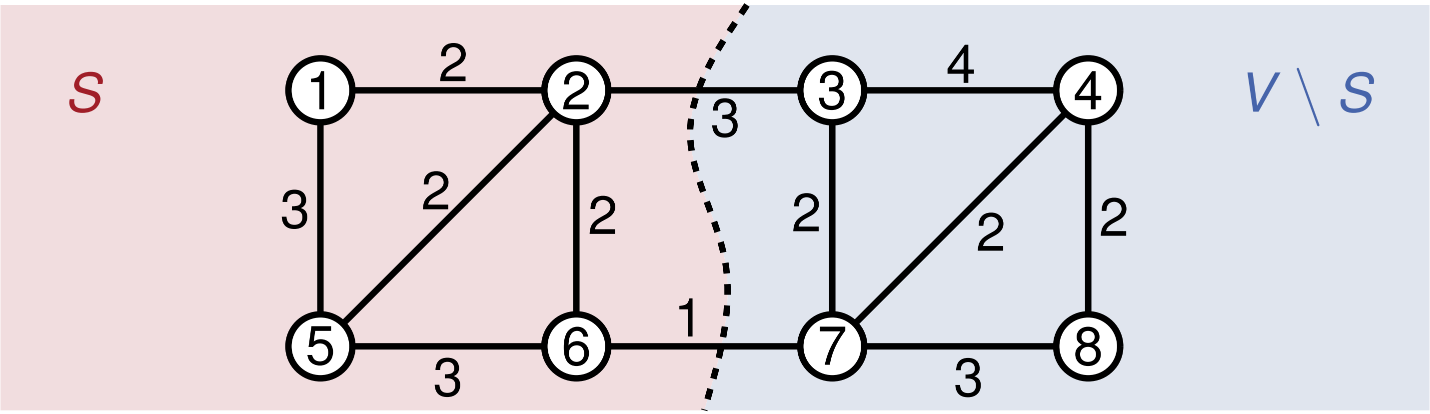
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Problem – MINCUT

Problem: MINCUT
Gegeben sei ein Graph $G = (V, E)$ mit Kantengewichtsfunktion $c\colon E \longrightarrow \mathbb{R}_0^+$.
Finde einen *nichttrivialen Schnitt* $(S, V \setminus S)$ *minimalen Gewichts* in G, d.h. finde $S \subseteq V$ mit $\emptyset \neq S \neq V$, sodass

$$c(S, V \setminus S) := \sum_{\substack{\{u,v\} \in E,\\ u \in S,\\ v \in V \setminus S}} c(\{u, v\})$$

minimal wird. $(S, V \setminus S)$ wird *minimaler Schnitt* genannt.



$c(S, V \setminus S) = 4$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Schnittberechnung mittels Flussalgorithmus

Karlsruher Institut für Technologie

Bemerkung: Dualität zu maximalem Fluss **(Bemerkung 3.1)**

Zu gegebenen $s, t \in V$ kann ein minimaler s-t-Schnitt mit einem Flussalgorithmus (z.B. Ford & Fulkerson, Goldberg & Tarjan) berechnet werden.

- Das Minimum über alle Paare $s, t \in V$ liefert einen global minimalen Schnitt. $\rightarrow \binom{|V|}{2} \in \Theta(|V|^2)$ Flussberechnungen.
- Da im minimalen Schnitt jeder Knoten von irgendeinem anderen getrennt wird, kann man stattdessen $s \in V$ auch festhalten und $t \in V \setminus \{s\}$ wähle. $\rightarrow |V| - 1$ Flussberechnungen.

Heute: Effizientere Berechnung eines minimalen Schnittes ohne Flussalgorithmus.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

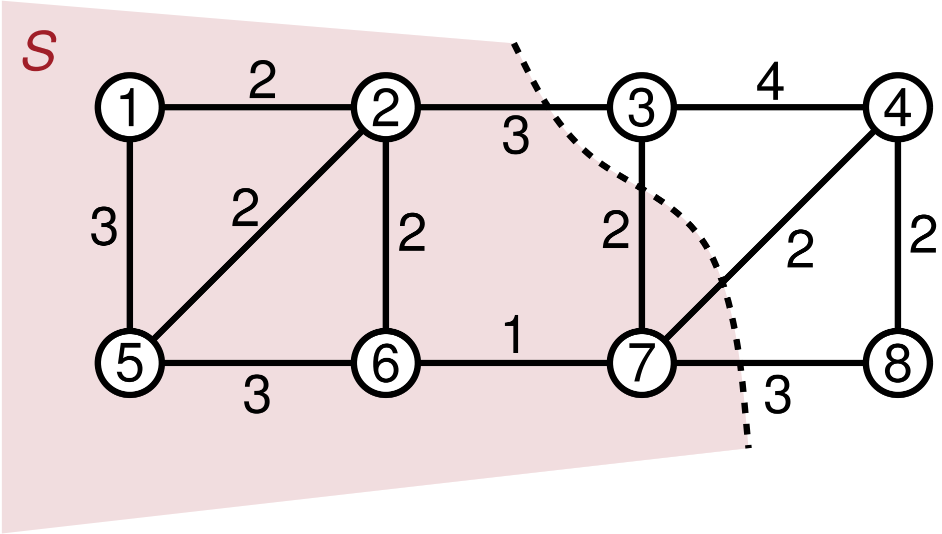
Stark verbundene Knoten

Definition: Am stärksten verbundene Knoten **(Definition 3.2)**

Zu $S \subseteq V$ und $v \in V \setminus S$ sei

$$c(S, v) = \sum_{\substack{\{u,v\} \in E \\ u \in S}} c(\{u, v\}).$$

Den Knoten $v \in V \setminus S$, für den $c(S, v)$ maximal wird, nennen wir auch den *am stärksten mit S verbundenen Knoten*.



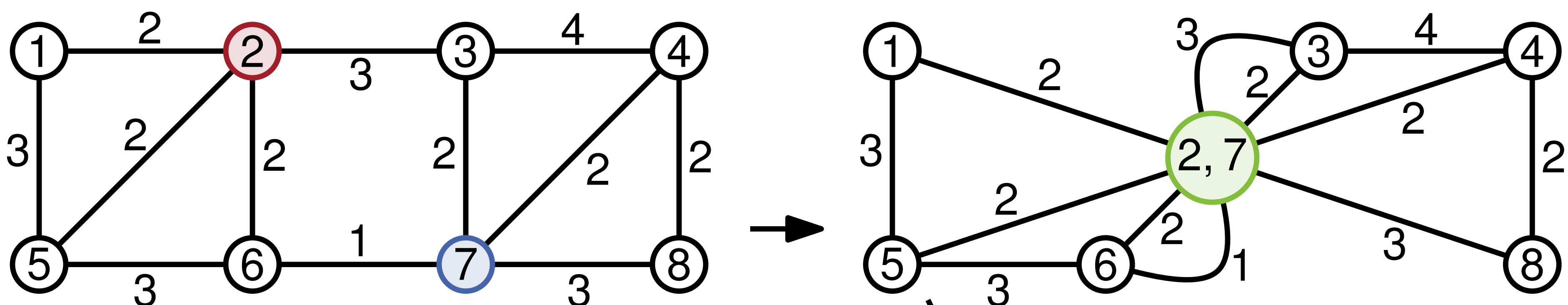
$c(S, 3) = 3 + 2 = 5$
$c(S, 4) = 2$
$c(S, 8) = 3$
$\Rightarrow$ Knoten 3 ist am stärksten mit S verbunden.

Verschmelzen zweier Knoten

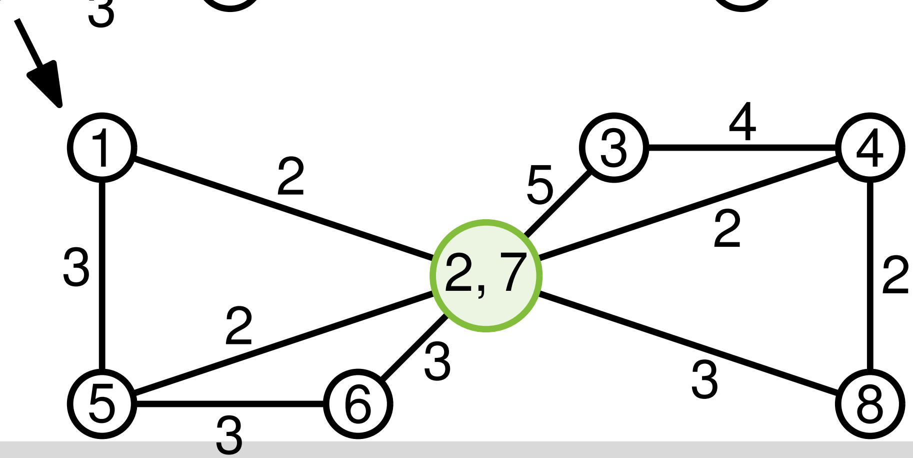
Definition: Verschmelzen zweier Knoten **(Definition 3.3)**

Seien $s, t \in V$. Dann können s und t wie folgt *verschmolzen* werden.

- s und t werden durch einen neuen Knoten $x_{s,t}$ ersetzt.
- Alle Kanten die vorher zu s oder t inzident waren sind jetzt zu $x_{s,t}$ inzident (abgesehen von $\{s, t\}$, falls s und t adjazent waren).
- Mehrfachkanten werden aufgelöst indem Kantengewichte addiert werden.



Verschmelzen der Knoten 2 und 7.



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Überblick

Karlsruher Institut für Technologie

Der Algorithmus von Stoer & Wagner besteht $|V| - 1$ Phasen.

- In jeder Phase i wird ein Schnitt in einem Graphen $G_i = (V_i, E_i)$ berechnet, der *Schnitt der Phase i*.
- G_i entsteht aus G_{i-1} durch Verschmelzen „geeigneter Knoten“, wobei $G_1 = G$.
- Ergebnis des Algorithmus ist der minimale Schnitt aller Schnitte der einzelnen Phasen i (für $1 \leq i \leq |V| - 1$).

Ablauf einer Phase i

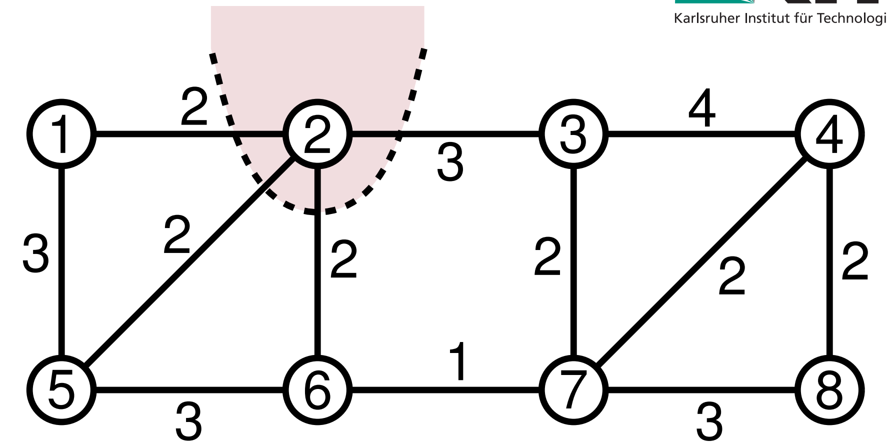
- Starte mit $S_i = \{a\}$, wobei a ein beliebiger Startknoten in G_i ist.
- Füge iterativ den am stärksten zu S_i verbundenen Knoten zu S_i hinzu.
- Seien s und t die als vorletztes bzw. als letztes zu S_i hinzugefügten Knoten.
- Der Schnitt der Phase i ist $(V_i \setminus \{t\}, \{t\})$.
- G_{i+1} entsteht aus G_i durch Verschmelzen von s und t.

Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)



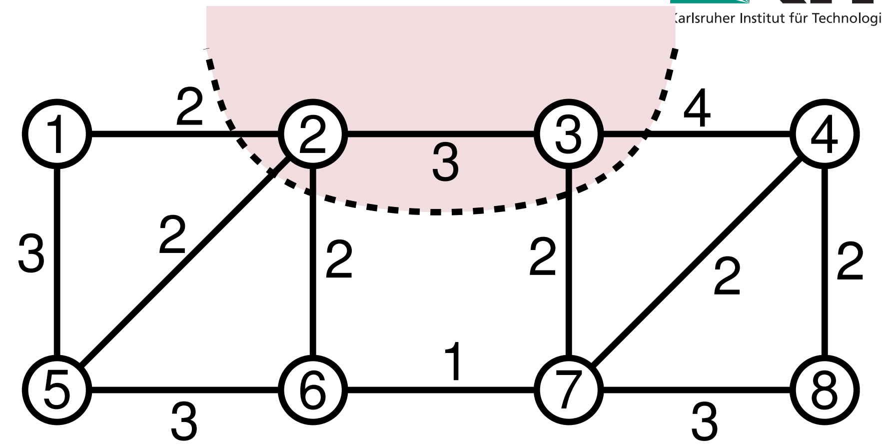
Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)

$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Beispiel

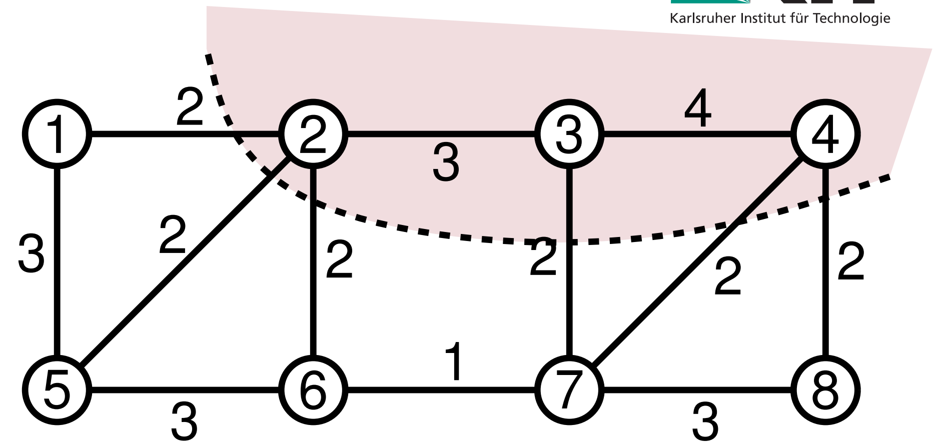
Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)

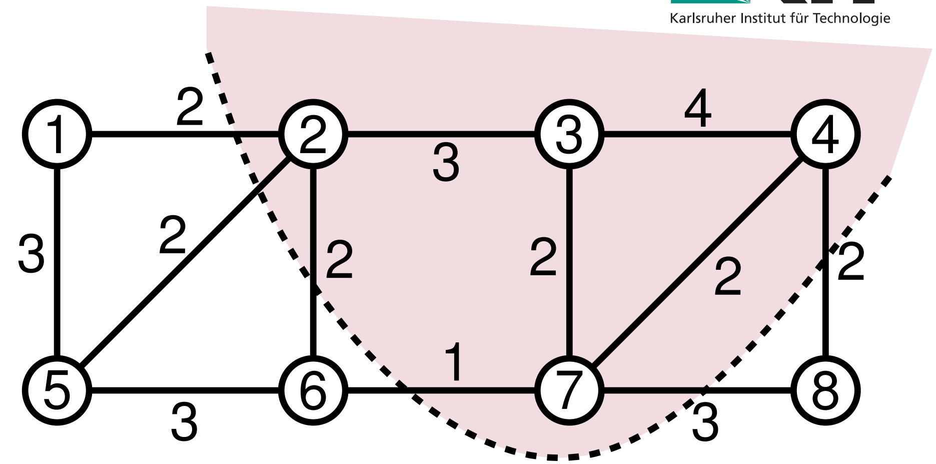
$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)

$S_1 = \{2, 3, 4\}$ (4 am stärksten zu $\{2, 3\}$ verbunden)



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Beispiel



Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)

$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)

$S_1 = \{2, 3, 4\}$ (4 am stärksten zu $\{2, 3\}$ verbunden)

$S_1 = \{2, 3, 4, 7\}$

Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$G_1 = G$

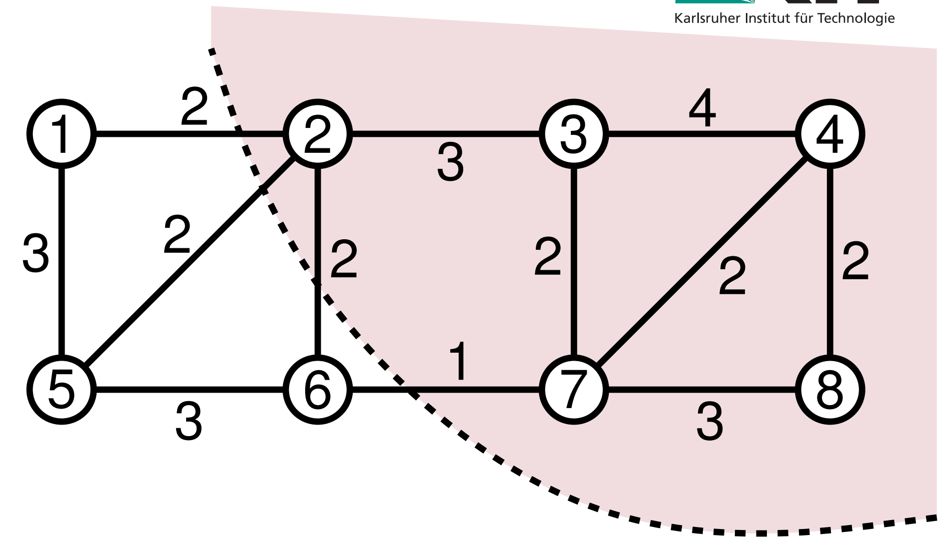
$S_1 = \{2\}$ (beliebig gewählter Startknoten)

$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)

$S_1 = \{2, 3, 4\}$ (4 am stärksten zu $\{2, 3\}$ verbunden)

$S_1 = \{2, 3, 4, 7\}$

$S_1 = \{2, 3, 4, 7, 8\}$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)

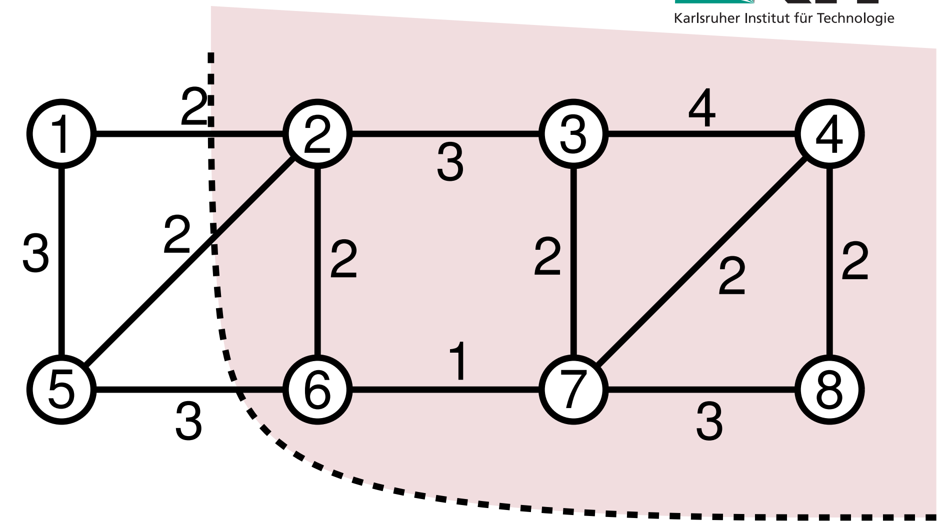
$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)

$S_1 = \{2, 3, 4\}$ (4 am stärksten zu $\{2, 3\}$ verbunden)

$S_1 = \{2, 3, 4, 7\}$

$S_1 = \{2, 3, 4, 7, 8\}$

$S_1 = \{2, 3, 4, 7, 8, 6\}$



Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)

$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)

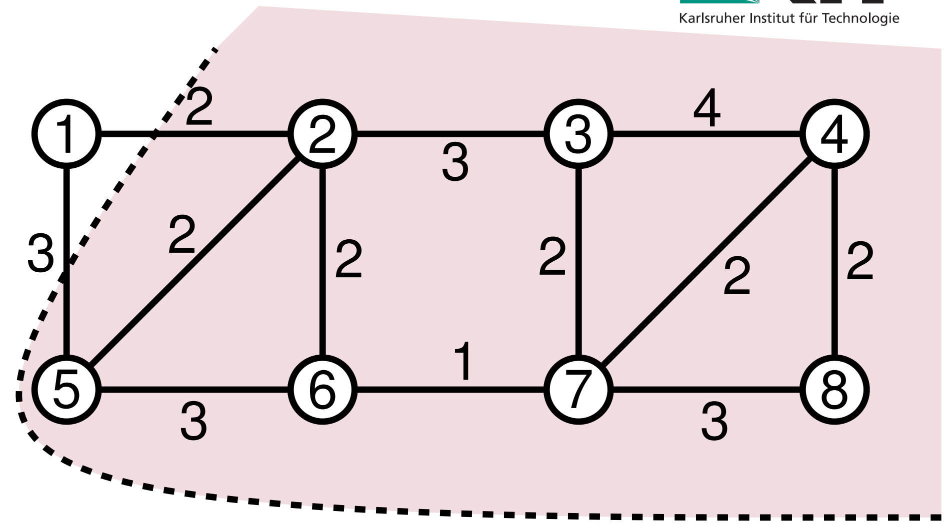
$S_1 = \{2, 3, 4\}$ (4 am stärksten zu $\{2, 3\}$ verbunden)

$S_1 = \{2, 3, 4, 7\}$

$S_1 = \{2, 3, 4, 7, 8\}$

$S_1 = \{2, 3, 4, 7, 8, 6\}$

$S_1 = \{2, 3, 4, 7, 8, 6, 5\}$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)

$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)

$S_1 = \{2, 3, 4\}$ (4 am stärksten zu $\{2, 3\}$ verbunden)

$S_1 = \{2, 3, 4, 7\}$

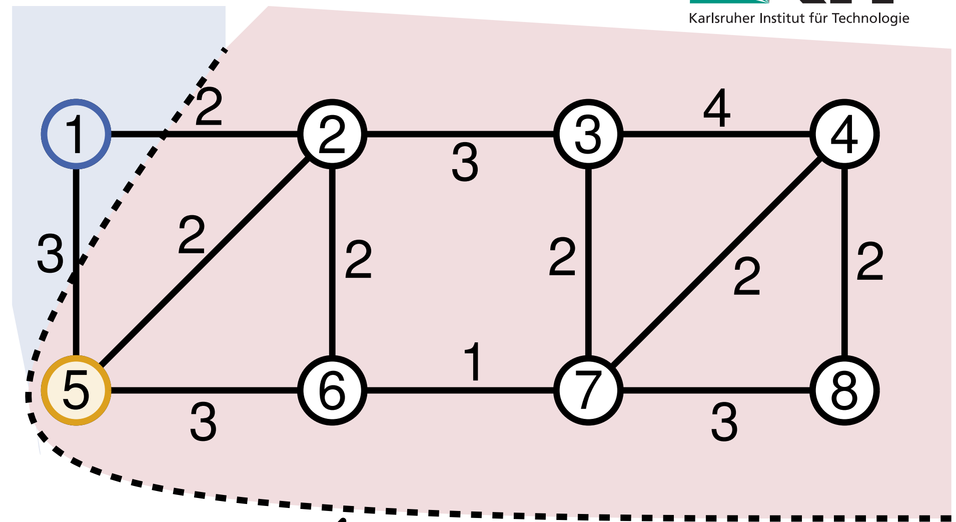
$S_1 = \{2, 3, 4, 7, 8\}$

$S_1 = \{2, 3, 4, 7, 8, 6\}$

$S_1 = \{2, 3, 4, 7, 8, 6, 5\}$

$S_1 = \{2, 3, 4, 7, 8, 6, 5, 1\}$

s t



Schnitt der Phase: $\{V_1 \setminus \{1\}, \{1\}\}$

$\rightarrow$ Gewicht 5

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Beispiel

Phase 1

$G_1 = G$

$S_1 = \{2\}$ (beliebig gewählter Startknoten)

$S_1 = \{2, 3\}$ (3 am stärksten zu $\{2\}$ verbunden)

$S_1 = \{2, 3, 4\}$ (4 am stärksten zu $\{2, 3\}$ verbunden)

$S_1 = \{2, 3, 4, 7\}$

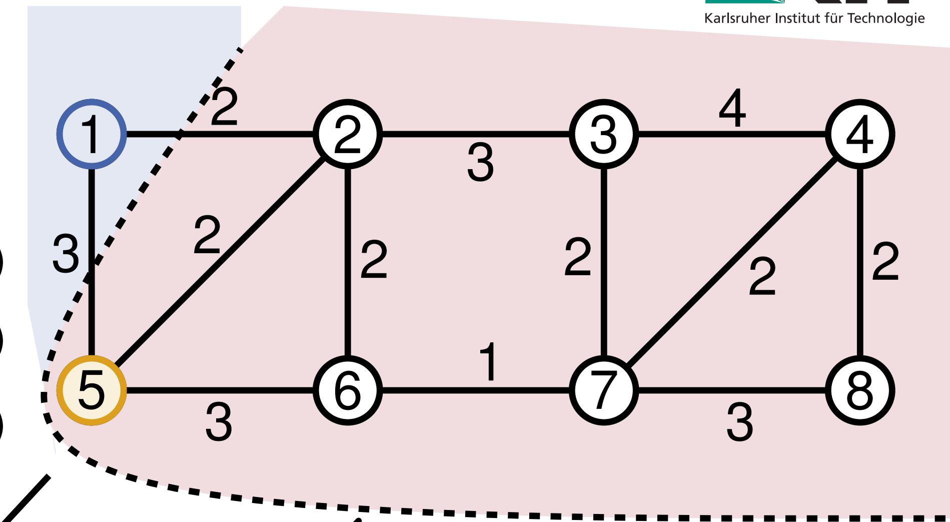
$S_1 = \{2, 3, 4, 7, 8\}$

$S_1 = \{2, 3, 4, 7, 8, 6\}$

$S_1 = \{2, 3, 4, 7, 8, 6, 5\}$

$S_1 = \{2, 3, 4, 7, 8, 6, 5, 1\}$

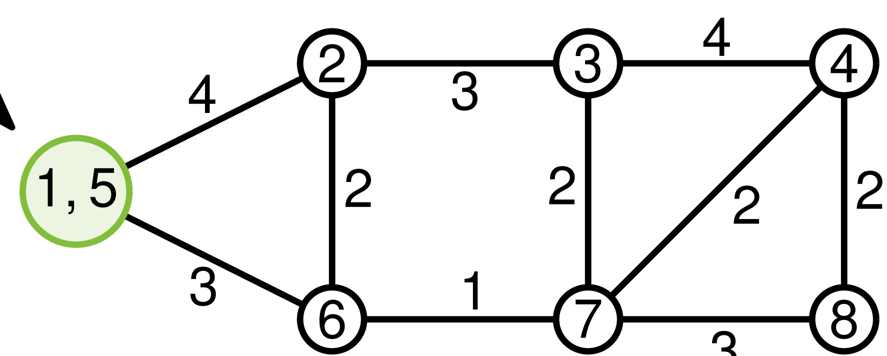
s t



Schnitt der Phase: $\{V_1 \setminus \{1\}, \{1\}\}$

$\rightarrow$ Gewicht 5

Verschmelzen von s und t ergibt G_2

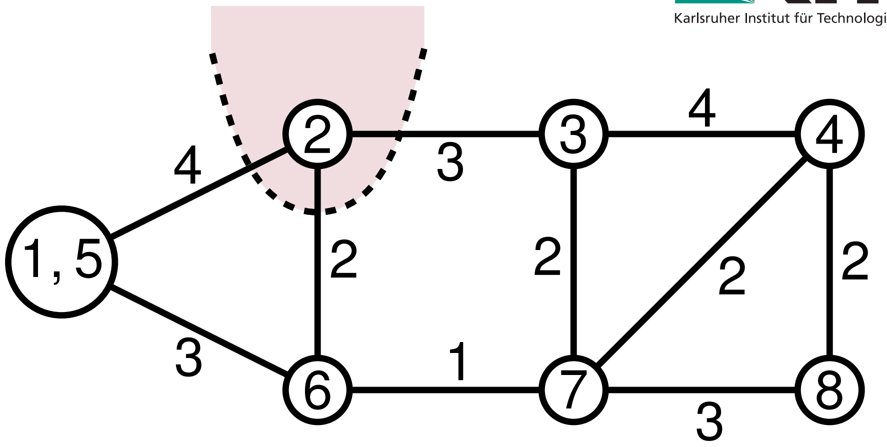


Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

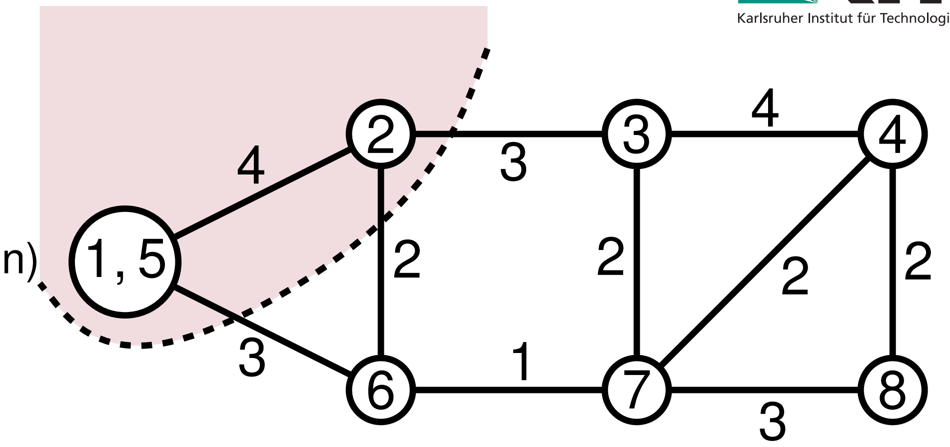
Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1, 5\}\}$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Beispiel

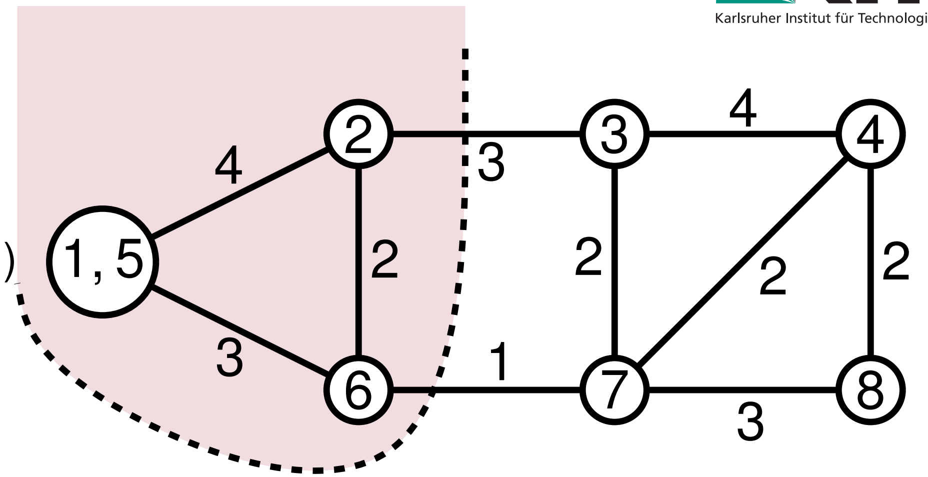
Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1,5\}\}$

$S_2 = \{2, \{1,5\}, 6\}$

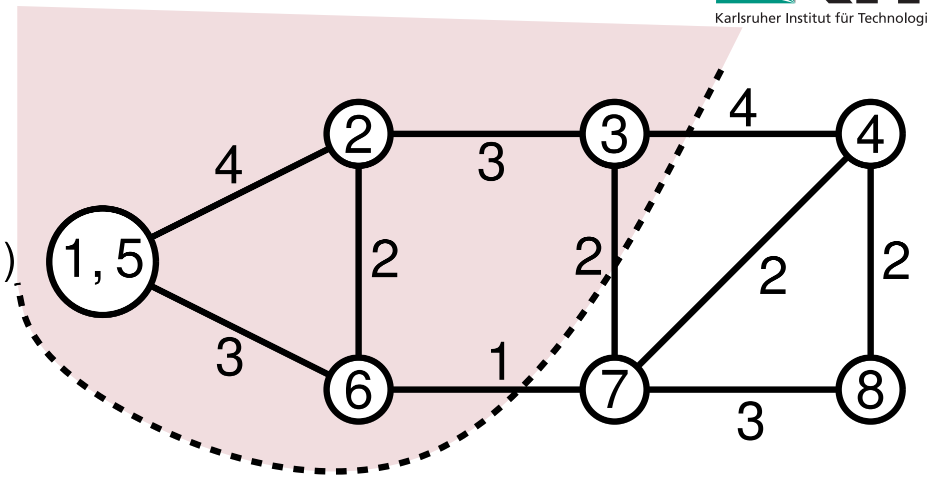


Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)



$S_2 = \{2, \{1, 5\}\}$

$S_2 = \{2, \{1, 5\}, 6\}$

$S_2 = \{2, \{1, 5\}, 6, 3\}$

Algorithmus von Stoer & Wagner – Beispiel

Karlsruher Institut für Technologie

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

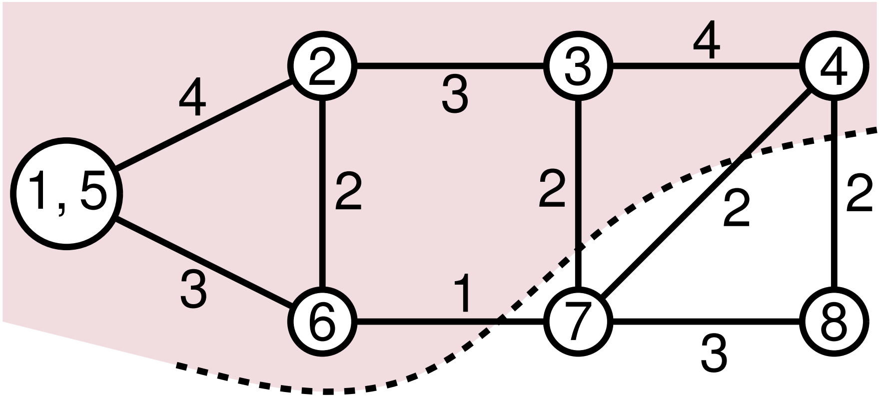
$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1, 5\}\}$

$S_2 = \{2, \{1, 5\}, 6\}$

$S_2 = \{2, \{1, 5\}, 6, 3\}$

$S_2 = \{2, \{1, 5\}, 6, 3, 4\}$



Algorithmus von Stoer & Wagner – Beispiel

Karlsruher Institut für Technologie

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

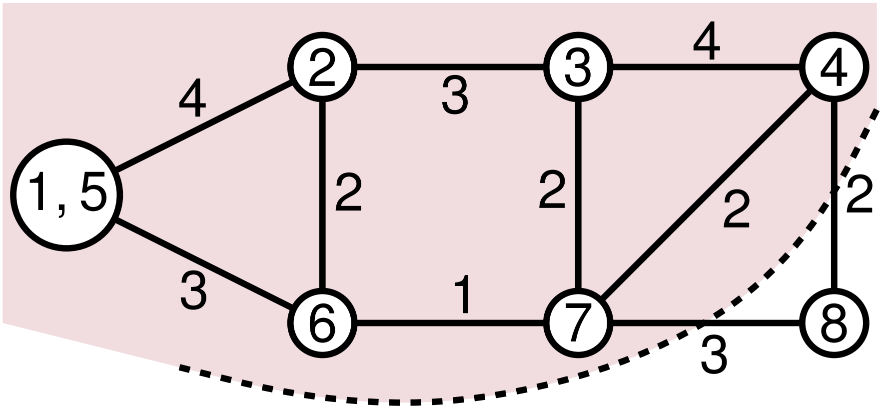
$S_2 = \{2, \{1,5\}\}$

$S_2 = \{2, \{1,5\}, 6\}$

$S_2 = \{2, \{1,5\}, 6, 3\}$

$S_2 = \{2, \{1,5\}, 6, 3, 4\}$

$S_2 = \{2, \{1,5\}, 6, 3, 4, 7\}$



Algorithmus von Stoer & Wagner – Beispiel

Phase 2

$G_2 = G_1$ mit 1 und 5 verschmolzen

$S_2 = \{2\}$ (beliebig gewählter Startknoten)

$S_2 = \{2, \{1,5\}\}$

$S_2 = \{2, \{1,5\}, 6\}$

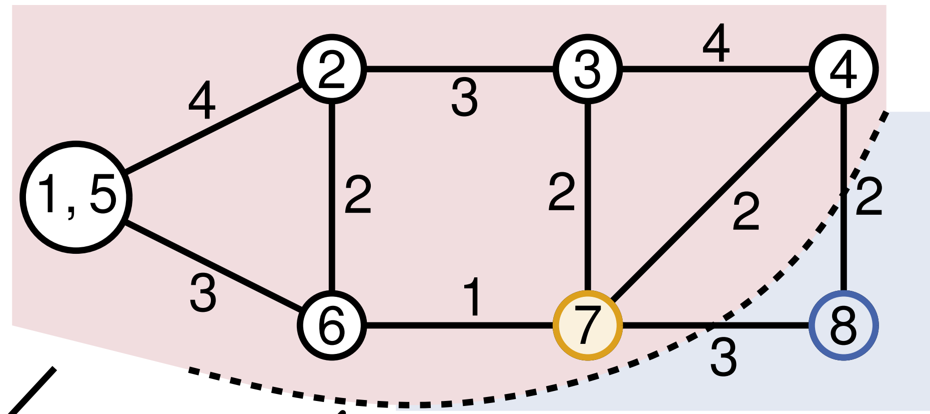
$S_2 = \{2, \{1,5\}, 6, 3\}$

$S_2 = \{2, \{1,5\}, 6, 3, 4\}$

$S_2 = \{2, \{1,5\}, 6, 3, 4, 7\}$

$S_2 = \{2, \{1,5\}, 6, 3, 4, 7, 8\}$

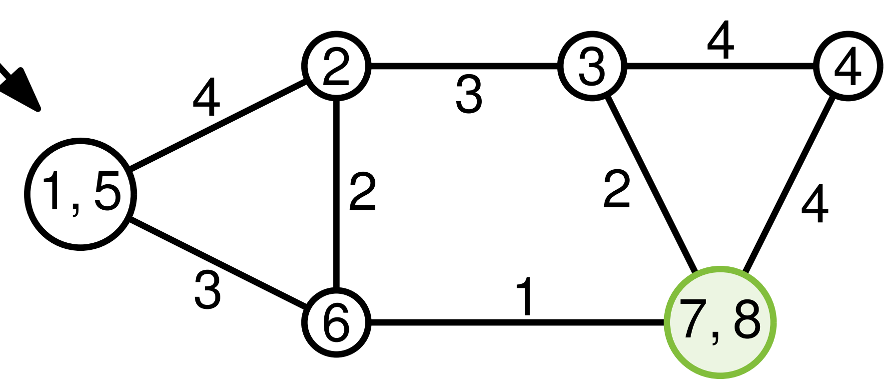
s t



Schnitt der Phase: $\{V_2 \setminus \{8\}, \{8\}\}$

$\rightarrow$ Gewicht 5

Verschmelzen von s und t ergibt G_3

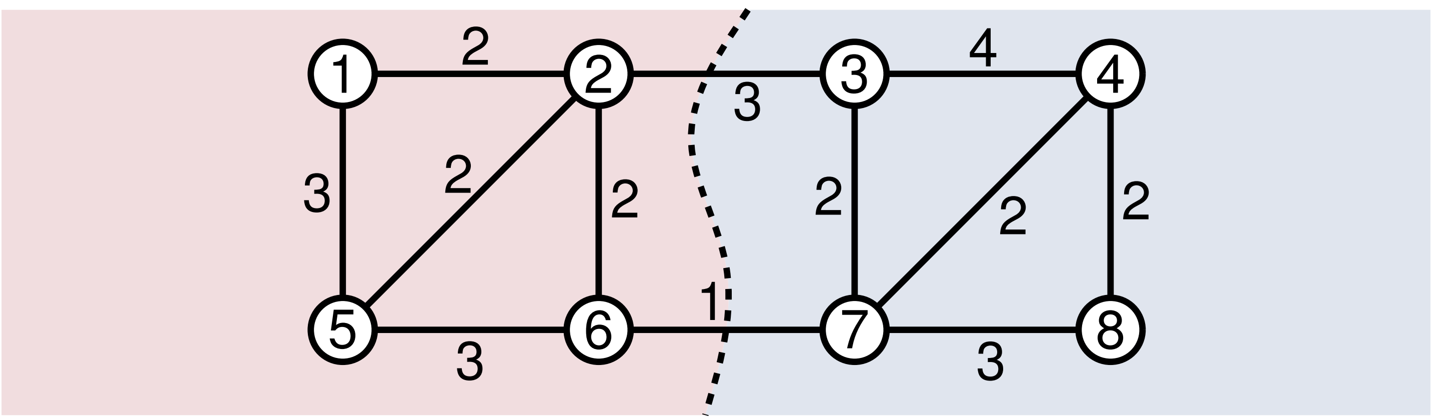


Algorithmus von Stoer & Wagner – Beispiel

Phase 1 Schnitt der Phase: $\{V_1 \setminus \{1\}, \{1\}\} \to$ Gewicht 5
Phase 2 Schnitt der Phase: $\{V_2 \setminus \{8\}, \{8\}\} \to$ Gewicht 5
Phase 3 Schnitt der Phase: $\{V_3 \setminus \{\{7,8\}\}, \{\{7,8\}\}\} \to$ Gewicht 7
Phase 4 Schnitt der Phase: $\{V_4 \setminus \{\{4,7,8\}\}, \{\{4,7,8\}\}\} \to$ Gewicht 7
Phase 5 Schnitt der Phase: $\{V_5 \setminus \{\{3,4,7,8\}\}, \{\{3,4,7,8\}\}\} \to$ Gewicht 4
Phase 6 Schnitt der Phase: $\{V_6 \setminus \{\{1,5\}\}, \{\{1,5\}\}\} \to$ Gewicht 7
Phase 7 Schnitt der Phase: $\{V_7 \setminus \{2\}, \{2\}\} \to$ Gewicht 9

siehe Skript

Der Schnitt aus **Phase 5** ist minimal unter den Schnitten der einzelnen Phasen.
$\Rightarrow$ Der Algorithmus von Stoer & Wagner gibt diesen Schnitt aus.



(Beweis, dass der so bestimmte Schnitt immer ein minimaler Schnitt ist folgt später.)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Laufzeit

MINSCHNITTPHASE(G_i, c, a)

$S \leftarrow \{a\}$

$t \leftarrow a$

$O(1)$

while $S \neq V_i$ **do**

$v \leftarrow$ Knoten aus $V_i \setminus S$ sodass $c(S, v)$ maximal $O(\log|V| + \deg(v))$

$S \leftarrow S \cup \{v\}$

$s \leftarrow t$

$t \leftarrow v$

$O(1)$

Speichere ($V_i \setminus \{t\}, \{t\}$) als SCHNITT-DER-PHASE

Konstruiere aus G_i Graph G_{i+1} durch Verschmelzen von s und t

Benutze einen FIBONACCI-HEAP um $c(S, u)$ für alle $u \in V_i \setminus S$ zu speichern.
Maximum v entfernen: $O(\log|V|)$
Nachbarn von v updaten: $O(\deg(v))$

Algorithmus von Stoer & Wagner – Laufzeit

MINSCHNITTPHASE(G_i, c, a)

$S \leftarrow \{a\}$
$t \leftarrow a$ — $O(1)$

while $S \neq V_i$ **do** — $O(|V| \log |V| + |E|)$
- $v \leftarrow$ Knoten aus $V_i \setminus S$ sodass $c(S, v)$ maximal — $O(\log |V| + \deg(v))$
- $S \leftarrow S \cup \{v\}$
  $s \leftarrow t$
  $t \leftarrow v$ — $O(1)$

Speichere ($V_i \setminus \{t\}, \{t\}$) als SCHNITT-DER-PHASE

Konstruiere aus G_i Graph G_{i+1} durch Verschmelzen von s und t

Benutze einen FIBONACCI-HEAP um $c(S, u)$ für alle $u \in V_i \setminus S$ zu speichern.
Maximum v entfernen: $O(\log |V|)$
Nachbarn von v updaten: $O(\deg(v))$

Jeder Knoten wird nur einmal zu S hinzugefügt.
$$\sum_{v \in V} \deg(v) = 2|E| \in O(|E|)$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Laufzeit

MINSCHNITTPHASE(G_i, c, a) $O(|V| \log |V| + |E|)$

$S \leftarrow \{a\}$
$t \leftarrow a$ $O(1)$

while $S \neq V_i$ **do** $O(|V| \log |V| + |E|)$

$v \leftarrow$ Knoten aus $V_i \setminus S$ sodass $c(S, v)$ maximal $O(\log |V| + \deg(v))$

$S \leftarrow S \cup \{v\}$
$s \leftarrow t$
$t \leftarrow v$ $O(1)$

Speichere $(V_i \setminus \{t\}, \{t\})$ als SCHNITT-DER-PHASE
Konstruiere aus G_i Graph G_{i+1} durch Verschmelzen von s und t $O(|E|)$

Benutze einen FIBONACCI-HEAP um $c(S, u)$ für alle $u \in V_i \setminus S$ zu speichern.
Maximum v entfernen: $O(\log |V|)$
Nachbarn von v updaten: $O(\deg(v))$

Jeder Knoten wird nur einmal zu S hinzugefügt.
$$\sum_{v \in V} \deg(v) = 2|E| \in O(|E|)$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Laufzeit

MINSCHNITTPHASE(G_i, c, a)	$O(\|V\| \log \|V\| + \|E\|)$

MIN-SCHNITT(G, c, a)	$O(\|V\|^2 \log \|V\| + \|V\|\|E\|)$
$G_1 \leftarrow G$	$O(1)$
for $i = 1$ **to** $\|V\| - 1$ **do**	$O(\|V\|^2 \log \|V\| + \|V\|\|E\|)$
MINSCHNITTPHASE(G_i, c, a)	$O(\|V\| \log \|V\| + \|E\|)$
if SCHNITT-DER-PHASE *ist kleiner als* MIN-SCHNITT **then** speichere SCHNITT-DER-PHASE als MIN-SCHNITT	$O(1)$
Gib MIN-SCHNITT aus.	$O(1)$

Lemma: Laufzeit des Algorithmus von Stoer & Wagner
Der Algorithmus von Stoer & Wagner hat eine Laufzeit von $O(|V|^2 \log |V| + |V||E|)$.

Zum Vergleich: Der Flussalgorithmus von Goldberg & Tarjan hat eine Laufzeit von $O(|V||E| \log(|V|^2/|E|))$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Korrektheit

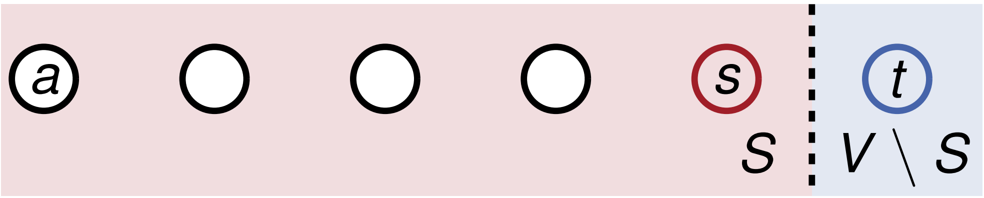
Definition: s-t-Schnitt
Ein Schnitt $(S, V \setminus S)$ heißt *s-t-Schnitt*, falls $s \in S$ und $t \in V \setminus S$ für $s, t \in V$, $s \neq t$. Ein s-t-Schnitt *trennt* Knoten u und v, wenn $u \in S$ und $v \in V \setminus S$.

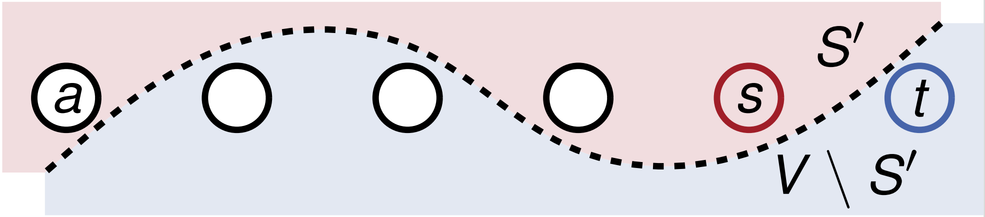
Lemma: Schnitt-der-Phase ist minimaler s-t-Schnitt **(Lemma 3.5)**
Sei $(S, V \setminus S)$ der Schnitt-der-Phase in einem Graphen $G = (V, E)$ mit Kostenfunktion $c\colon E \longrightarrow \mathbb{R}_0^+$ und Startknoten $a \in V$. Seien s und t der vorletzte bzw. letzte betrachtete Knoten. Dann ist $(S, V \setminus S)$ minimal unter allen s-t-Schnitten.

Definition: s-t-Schnitt
Ein Schnitt $(S, V \setminus S)$ heißt *s-t-Schnitt*, falls $s \in S$ und $t \in V \setminus S$ für $s, t \in V$, $s \neq t$. Ein s-t-Schnitt *trennt* Knoten u und v, wenn $u \in S$ und $v \in V \setminus S$.

Lemma: Schnitt-der-Phase ist minimaler s-t-Schnitt **(Lemma 3.5)**
Sei $(S, V \setminus S)$ der Schnitt-der-Phase in einem Graphen $G = (V, E)$ mit Kostenfunktion $c\colon E \longrightarrow \mathbb{R}_0^+$ und Startknoten $a \in V$. Seien s und t der vorletzte bzw. letzte betrachtete Knoten. Dann ist $(S, V \setminus S)$ minimal unter allen s-t-Schnitten.

Beweis: Zeige: Für jeden s-t-Schnitt $(S', V \setminus S')$ gilt: $c(S, V \setminus S) \leq c(S', V \setminus S')$

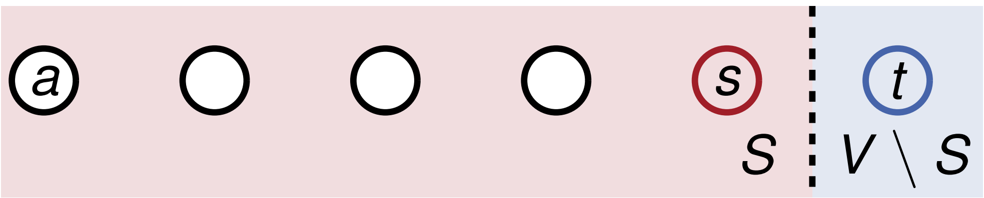


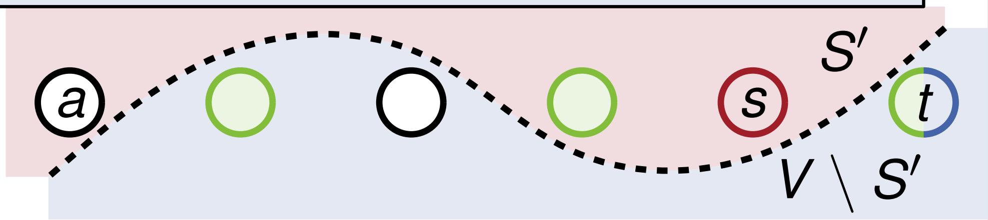


Algorithmus von Stoer & Wagner – Korrektheit

Beweis: Zeige: Für jeden s-t-Schnitt $(S', V \setminus S')$ gilt: $c(S, V \setminus S) \leq c(S', V \setminus S')$

Definition: aktive Knoten
MINSCHNITTPHASE betrachtet die Knoten aus V gemäß einer linearen Ordnung, die mit a beginnt und mit s und t endet. Ein Knoten $v \in V$ heißt *aktiv* (bzgl. S'), wenn $\{S', V \setminus S'\}$ den Knoten v von seinem Vorgänger trennt.



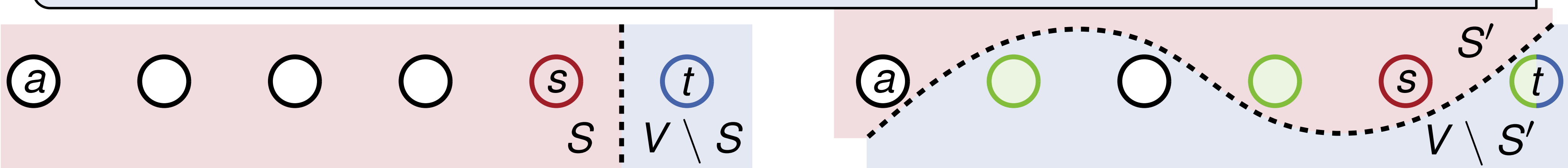


Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

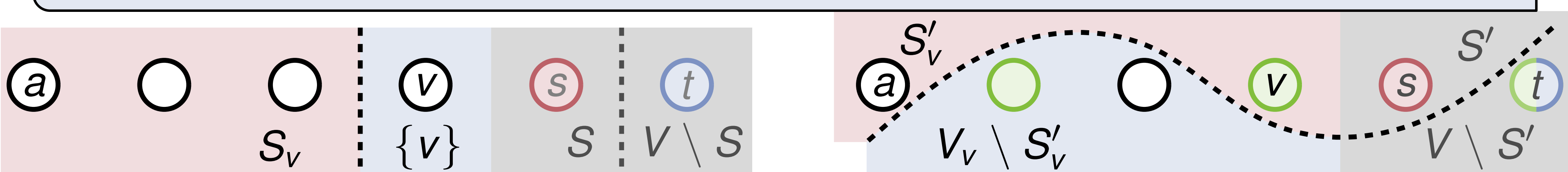
Algorithmus von Stoer & Wagner – Korrektheit

Beweis: Zeige: Für jeden s-t-Schnitt $(S', V \setminus S')$ gilt: $c(S, V \setminus S) \leq c(S', V \setminus S')$

Definition: aktive Knoten
MINSCHNITTPHASE betrachtet die Knoten aus V gemäß einer linearen Ordnung, die mit a beginnt und mit s und t endet. Ein Knoten $v \in V$ heißt *aktiv* (bzgl. S'), wenn $\{S', V \setminus S'\}$ den Knoten v von seinem Vorgänger trennt.



Definition: Für $v \in V \setminus \{a\}$ sei S_v Menge der Knoten vor v.
Sei weiter $V_v = S_v \cup \{v\}$ sowie $S'_v = S' \cap V_v$.

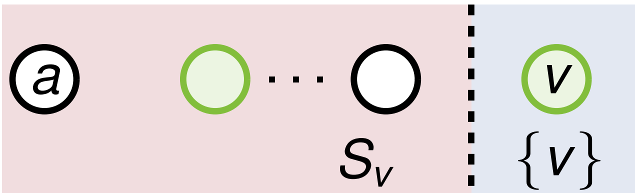


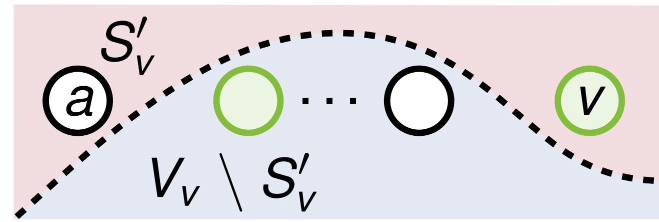
Betrachte Einschränkung von G auf V_v für aktiven Knoten v. Zeige:
$c(S_V, \{v\}) \leq c(S'_v, V_v \setminus S'_v)$ (zeigt genau das gewünschte für $v = t$)

Algorithmus von Stoer & Wagner – Korrektheit

Zeige induktiv über aktive Knoten (entsprechend Einfügereihenfolge):

$$c(S_V, \{v\}) \quad \leq \quad c(S'_v, V_v \setminus S'_v)$$

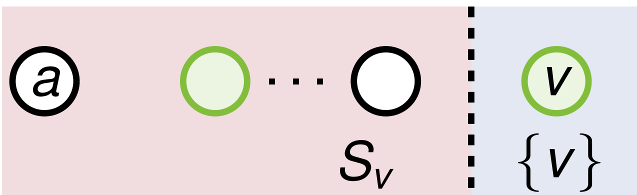


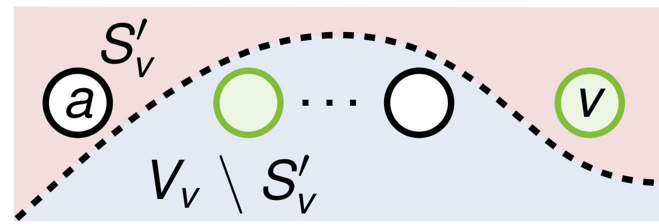


Algorithmus von Stoer & Wagner – Korrektheit

Zeige induktiv über aktive Knoten (entsprechend Einfügereihenfolge):

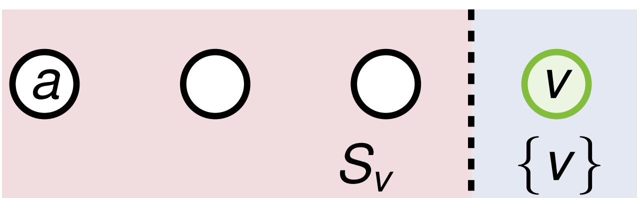
$$c(S_V, \{v\}) \quad \leq \quad c(S'_v, V_v \setminus S'_v)$$

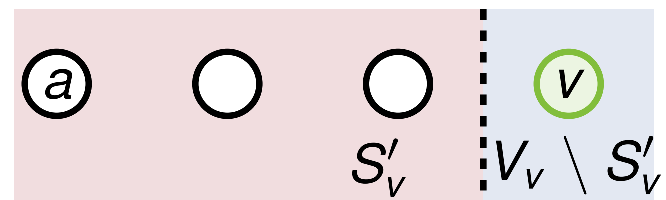




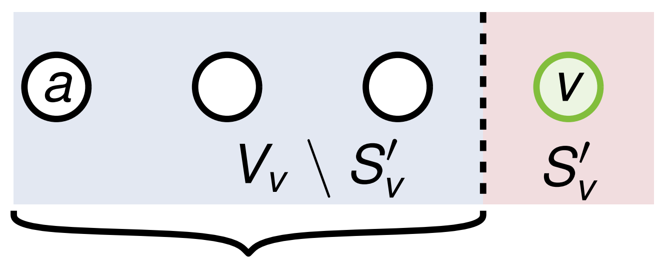
Induktionsanfang: Sei v erster aktiver Knoten.

$$c(S_V, \{v\}) \quad = \quad c(S'_v, V_v \setminus S'_v)$$





oder



also auch $\leq$

keine aktiven Knoten

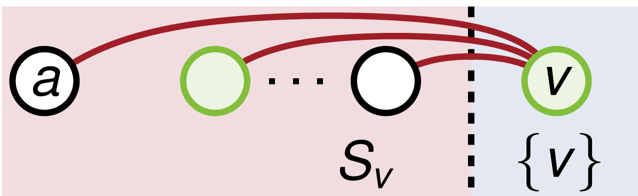
($\Rightarrow$ kein Knoten wird vom Vorgänger getrennt)

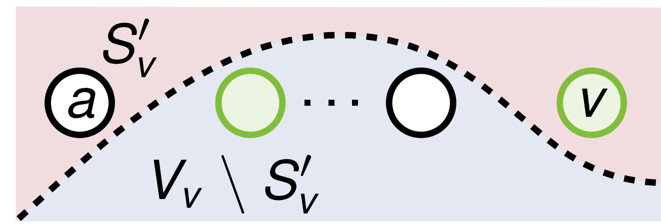
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Algorithmus von Stoer & Wagner – Korrektheit

Zeige induktiv über aktive Knoten (entsprechend Einfügereihenfolge):

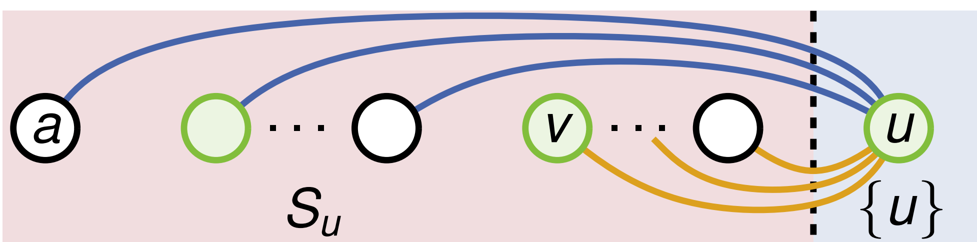
$$c(S_v, \{v\}) \quad \leq \quad c(S'_v, V_v \setminus S'_v)$$





Induktionsschritt: Behauptung gilt für v; sei u nächster aktiver Knoten.

Schätze zunächst $c(S_u, \{u\})$ ab:



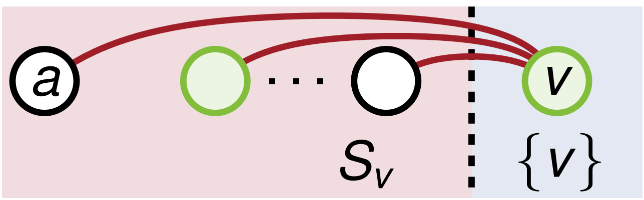
$$\begin{aligned} c(S_u, \{u\}) &= c(S_v, \{u\}) + c(S_u \setminus S_v, \{u\}) \\ &\leq c(S_v, \{v\}) + c(S_u \setminus S_v, \{u\}) \end{aligned}$$

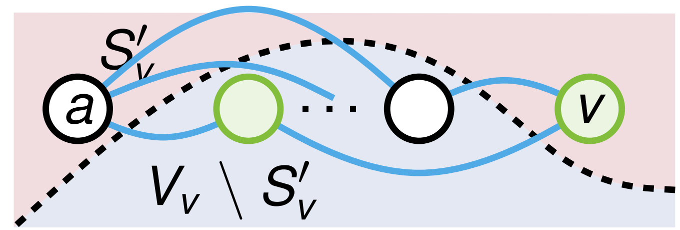
v ist mindestens so stark mit S_v verbunden wie u
(sonst wäre die Reihenfolge anders)

Algorithmus von Stoer & Wagner – Korrektheit

Zeige induktiv über aktive Knoten (entsprechend Einfügereihenfolge):

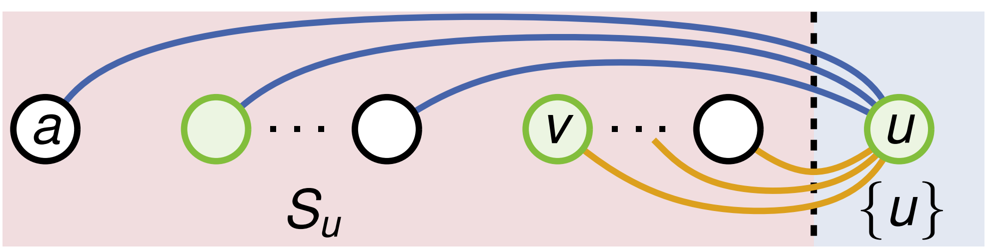
$c(S_v, \{v\}) \quad \leq \quad c(S'_v, V_v \setminus S'_v)$





Induktionsschritt: Behauptung gilt für v; sei u nächster aktiver Knoten.

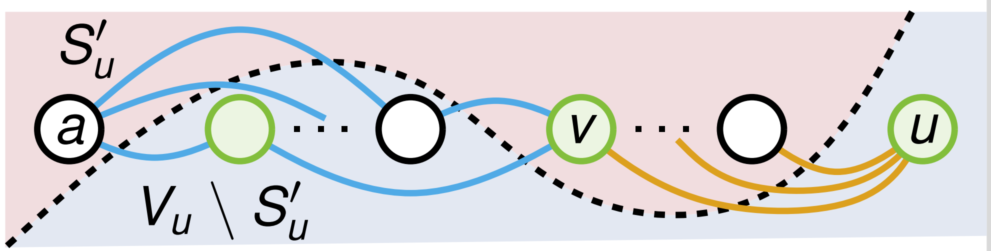
Schätze zunächst $c(S_u, \{u\})$ ab:



$$c(S_u, \{u\}) = c(S_v, \{u\}) + c(S_u \setminus S_v, \{u\})$$
$$\leq c(S_v, \{v\}) + c(S_u \setminus S_v, \{u\})$$

v ist mindestens so stark mit S_v verbunden wie u
(sonst wäre die Reihenfolge anders)

Schätze dann $c(S'_u, V_u \setminus S'_u)$ ab:

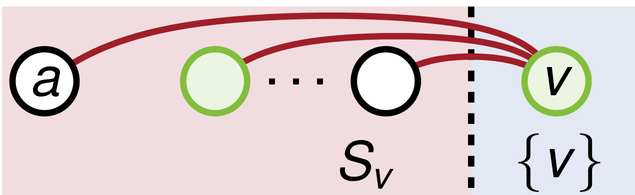


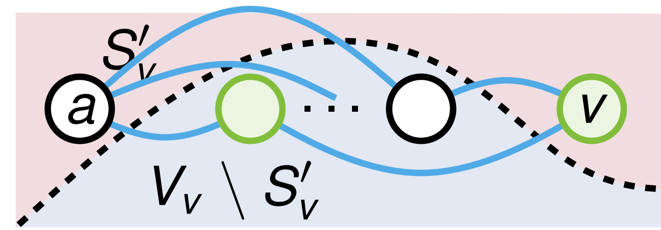
$$c(S'_v, V_v \setminus S'_v) + c(S_u \setminus S_v, \{u\}) \leq c(S'_u, V_u \setminus S'_u)$$

Algorithmus von Stoer & Wagner – Korrektheit

Zeige induktiv über aktive Knoten (entsprechend Einfügereihenfolge):

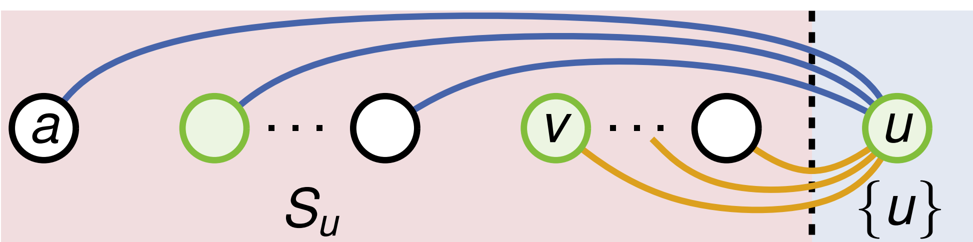
$$c(S_v, \{v\}) \quad \leq \quad c(S'_v, V_v \setminus S'_v)$$



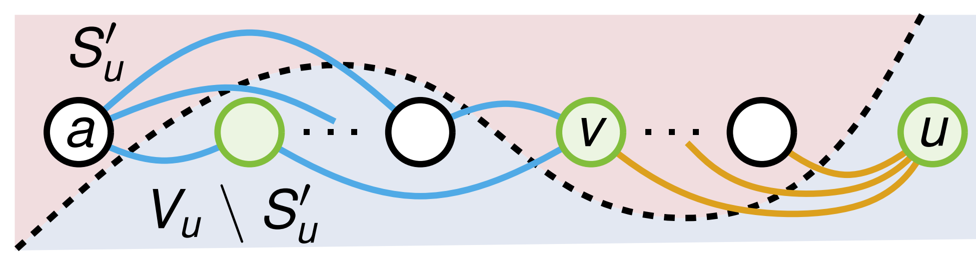


Induktionsschritt: Behauptung gilt für v; sei u nächster aktiver Knoten.

Schätze zunächst $c(S_u, \{u\})$ ab:



Schätze dann $c(S'_u, V_u \setminus S'_u)$ ab:



$$\begin{aligned} c(S_u, \{u\}) &= c(S_v, \{u\}) + c(S_u \setminus S_v, \{u\}) \\ &\leq c(S_v, \{v\}) + c(S_u \setminus S_v, \{u\}) \leq c(S'_v, V_v \setminus S'_v) + c(S_u \setminus S_v, \{u\}) \leq c(S'_u, V_u \setminus S'_u) \end{aligned}$$

nach Induktionsvoraussetzung

v ist mindestens so stark mit S_v verbunden wie u
(sonst wäre die Reihenfolge anders)

Algorithmus von Stoer & Wagner – Korrektheit (2)

Satz: Korrektheit des Algorithmus von Stoer & Wagner **(Satz 3.6)**

Der minimale Schnitt von allen Ergebnissen der $|V| - 1$ Ausführungen von MIN-SCHNITTPHASE ist ein minimaler, nichttrivialer Schnitt in $G = (V, E)$ mit $|V| \geq 2$.

Algorithmus von Stoer & Wagner – Korrektheit (2)

Satz: Korrektheit des Algorithmus von Stoer & Wagner **(Satz 3.6)**
Der minimale Schnitt von allen Ergebnissen der $|V| - 1$ Ausführungen von MIN-SCHNITTPHASE ist ein minimaler, nichttrivialer Schnitt in $G = (V, E)$ mit $|V| \geq 2$.

Beweis: Induktion über $|V|$.

Induktionsanfang: $|V| = 2$ ist trivial.

Induktionsschritt: $|V| \geq 3$

Betrachte Phase 1 mit vorletztem bzw. letztem Knoten s und t.
Fall 1: G hat einen nichttrivialen minimalen Schnitt, der s von t trennt.
$\Rightarrow$ Schnitt der ersten Phase ist ein nichttrivialer minimaler Schnitt.
Fall 2: G hat keinen nichttrivialen minimalen Schnitt, der s von t trennt.
$\Rightarrow$ In jedem nichttrivialen minimalen Schnitt liegen s und t auf der gleichen Seite.
$\Rightarrow$ Verschmilzt man s und t, so induziert ein minimaler Schnitt im resultierenden Graph G' einen in minimalen Schnitt in G.
$\Rightarrow$ Laut Induktionsvoraussetzung liefert der Algorithmus einen minimalen Schnitt für G'.

$\square$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner