Effiziente Algorithmen (SS2015)

Kapitel 2
Weitere Flüsse

Walter Unger

Lehrstuhl für Informatik 1

7:05 Uhr, den 6. November 2018

Inhalt I

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|$, $m = |E|$)
- $s, t \in V$ mit $s \neq t$

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : c'(e) \leqslant f(e) \leqslant c(e)$

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|$, $m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : c'(e) \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Das Flussproblem mit Mindestfluss

Definition (Flussproblem mit Mindestfluss)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

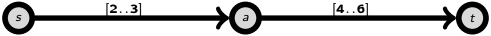
Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : c'(e) \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Ziel: Bestimme, ob es so einen Fluss gibt. Falls ja, dann maximiere $w(f) = \sum_{(s,v)\in E} f((s, v))$.

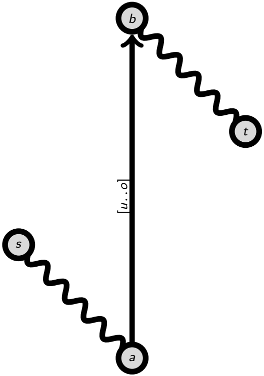
Lösbarkeit

Es muss nicht immer eine Lösung geben. Hier ein einfaches Beispiel:

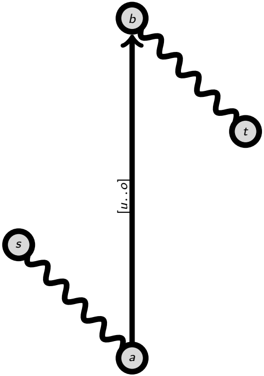


Idee

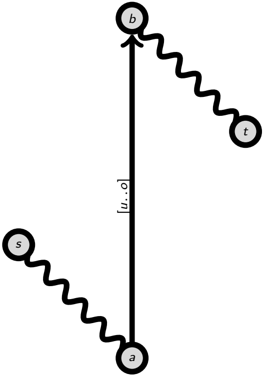
Idee



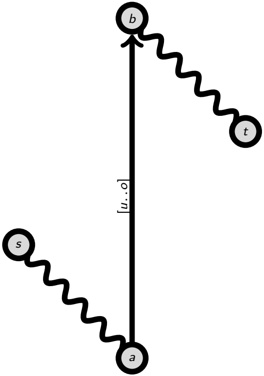
Idee

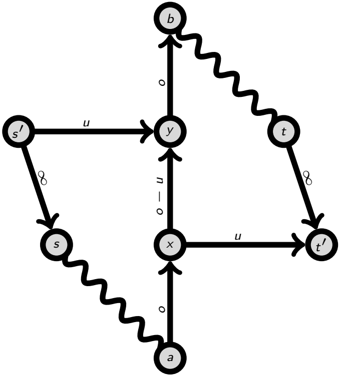


Idee



Idee





Verfahren

Verfahren

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.
- Ersetze jede Kante (v, w) durch einen Weg der Länge 3:

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.
- Ersetze jede Kante (v, w) durch einen Weg der Länge 3:
  - für jede Kante (v, w) erzeuge zwei neue Knoten x, y und setze:

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.
- Ersetze jede Kante (v, w) durch einen Weg der Länge 3:
  - für jede Kante (v, w) erzeuge zwei neue Knoten x, y und setze:
  - $c''(v, x) = c(v, w)$ und $c''(y, w) = c(v, w)$.

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.
- Ersetze jede Kante (v, w) durch einen Weg der Länge 3:
  - für jede Kante (v, w) erzeuge zwei neue Knoten x, y und setze:
  - $c''(v, x) = c(v, w)$ und $c''(y, w) = c(v, w)$.
  - $c''(x, y) = c(v, w) - c'(v, w)$.

Verfahren

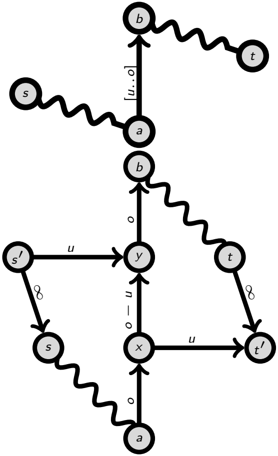
- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.
- Ersetze jede Kante (v, w) durch einen Weg der Länge 3:
  - für jede Kante (v, w) erzeuge zwei neue Knoten x, y und setze:
  - $c''(v, x) = c(v, w)$ und $c''(y, w) = c(v, w)$.
  - $c''(x, y) = c(v, w) - c'(v, w)$.
  - $c''(s', y) = c'(v, w)$ und $c''(x, t') = c'(v, w)$.

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.
- Ersetze jede Kante (v, w) durch einen Weg der Länge 3:
  - für jede Kante (v, w) erzeuge zwei neue Knoten x, y und setze:
  - $c''(v, x) = c(v, w)$ und $c''(y, w) = c(v, w)$.
  - $c''(x, y) = c(v, w) - c'(v, w)$.
  - $c''(s', y) = c'(v, w)$ und $c''(x, t') = c'(v, w)$.
- Setze $c''(t, t') = c''(s', s) = \sum_{e \in E} c(e)$.

Verfahren

- Erzeuge aus $G = (V, E, s, t, c, c')$ einen neuen Graphen $G' = (V', E', s', t', c'')$.
- Füge neue Quelle s' und neue Senke t' hinzu.
- Ersetze jede Kante (v, w) durch einen Weg der Länge 3:
  - für jede Kante (v, w) erzeuge zwei neue Knoten x, y und setze:
  - $c''(v, x) = c(v, w)$ und $c''(y, w) = c(v, w)$.
  - $c''(x, y) = c(v, w) - c'(v, w)$.
  - $c''(s', y) = c'(v, w)$ und $c''(x, t') = c'(v, w)$.
- Setze $c''(t, t') = c''(s', s) = \sum_{e \in E} c(e)$.



Aussage

Aussage

Aussage

Lemma

Es gibt in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt genau dann, wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert.

Aussage

Lemma

Es gibt in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt genau dann, wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert.

Aussage

Lemma

Es gibt in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt genau dann, wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert.

Beachte: x und y sind neu eingefügte Knoten, also nicht s oder t.
Beweis:

Aussage

Lemma

Es gibt in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt genau dann, wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert.

Beachte: x und y sind neu eingefügte Knoten, also nicht s oder t.
Beweis:

- Zeige: $\Longrightarrow$

Aussage

Lemma

Es gibt in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt genau dann, wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert.

Beachte: x und y sind neu eingefügte Knoten, also nicht s oder t.
Beweis:

- Zeige: $\Longrightarrow$
- Zeige: $\Longleftarrow$

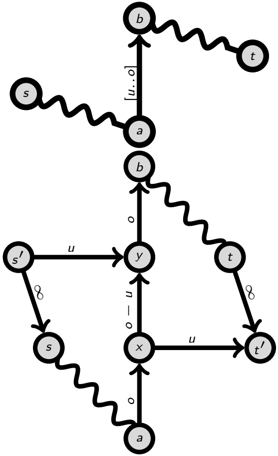
Aussage

Lemma

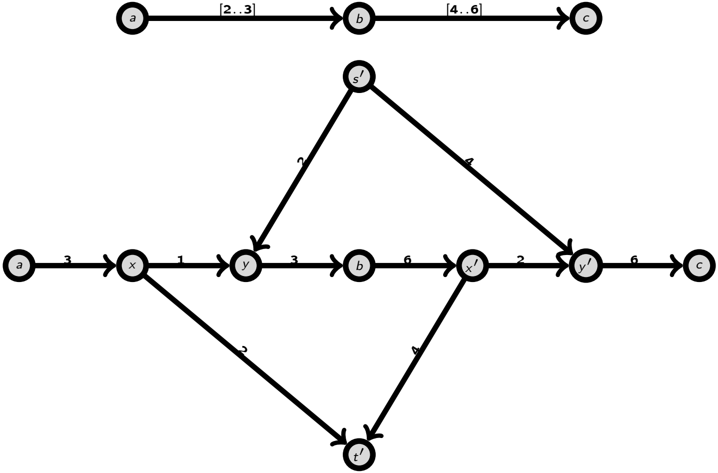
Es gibt in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt genau dann, wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert.

Beachte: x und y sind neu eingefügte Knoten, also nicht s oder t.
Beweis:

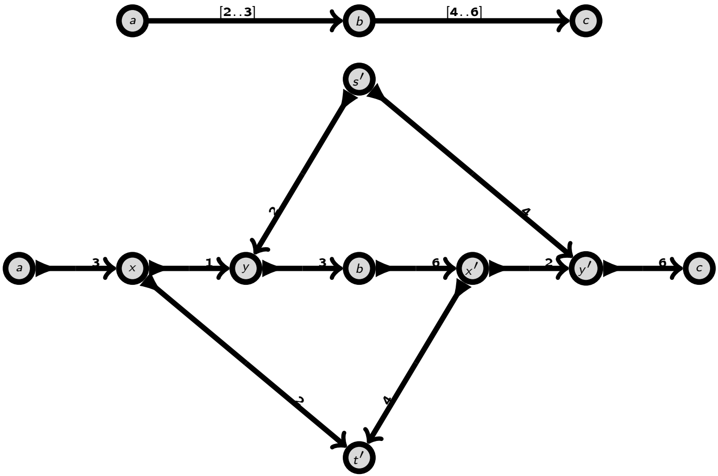
- Zeige: $\Longrightarrow$
- Zeige: $\Longleftarrow$



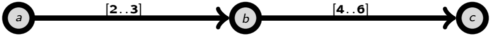
Beispiel (warum (s', y) und (x, t'))

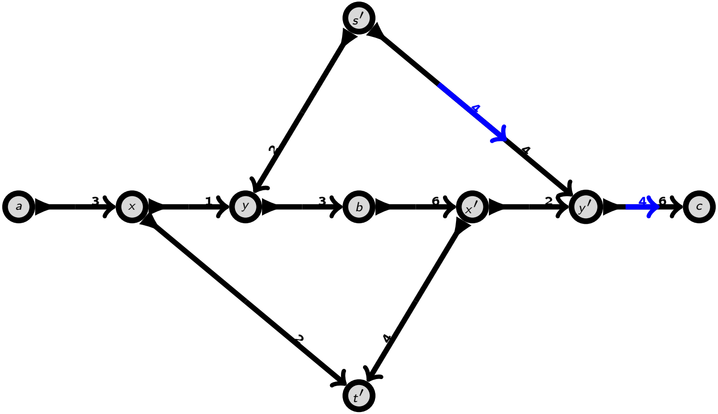


Beispiel (warum (s', y) und (x, t'))

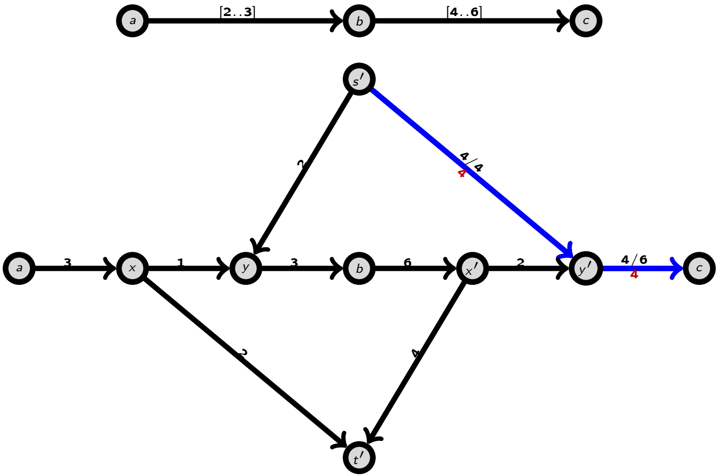


Beispiel (warum (s', y) und (x, t'))

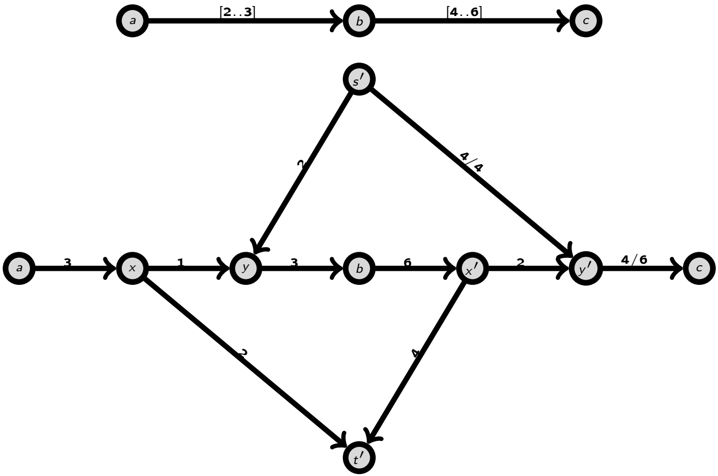




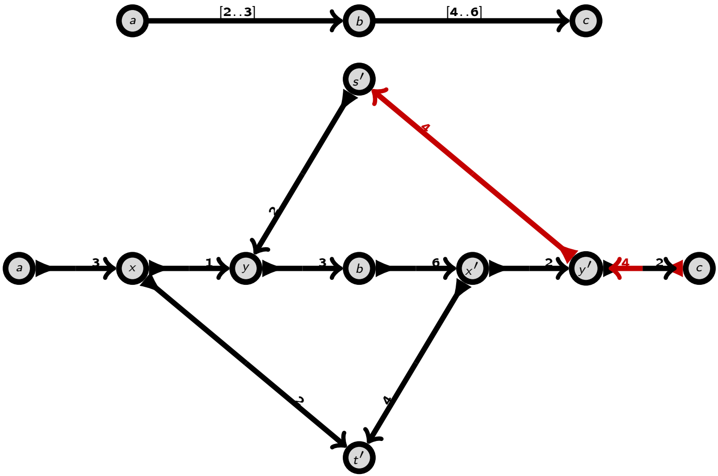
Beispiel (warum (s', y) und (x, t'))



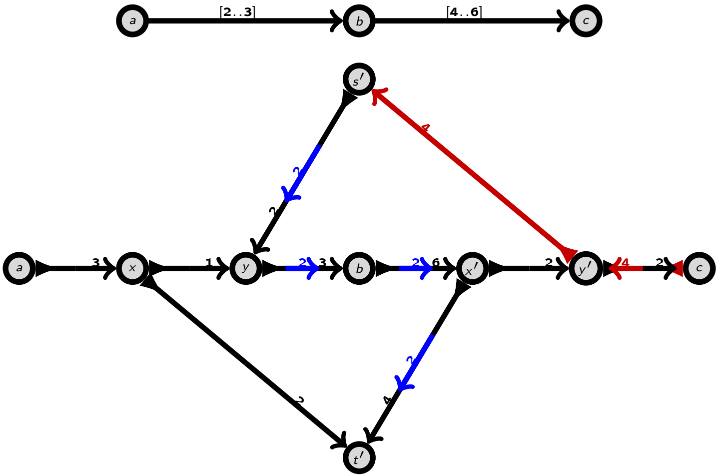
Beispiel (warum (s', y) und (x, t'))



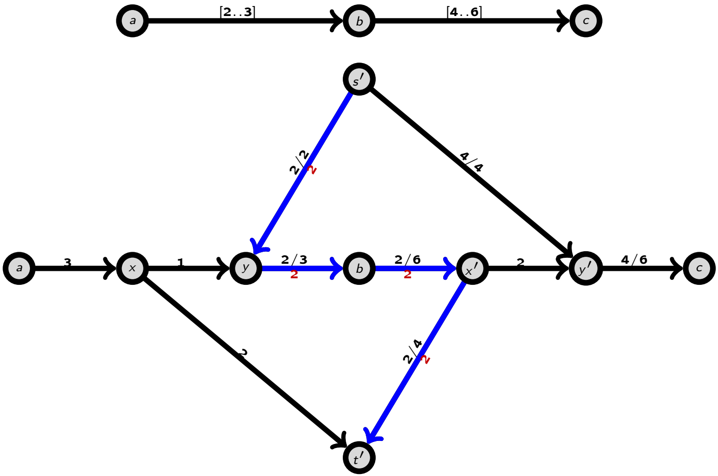
Beispiel (warum (s', y) und (x, t'))



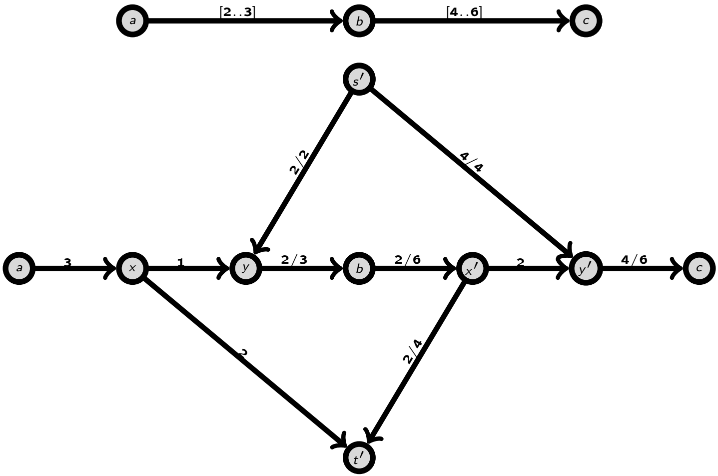
Beispiel (warum (s', y) und (x, t'))



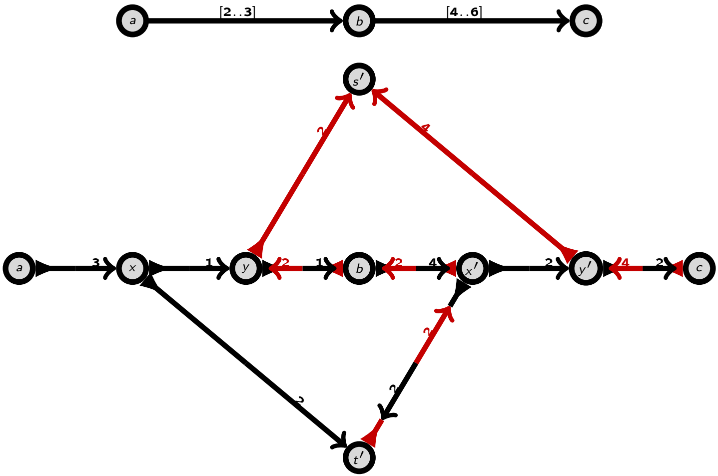
Beispiel (warum (s', y) und (x, t'))



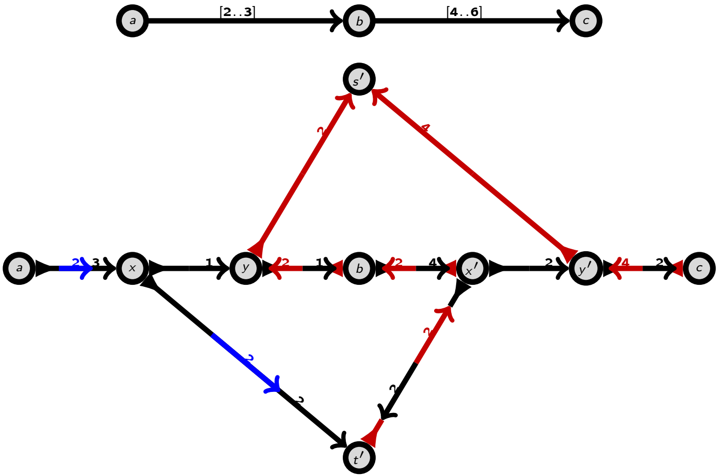
Beispiel (warum (s', y) und (x, t'))



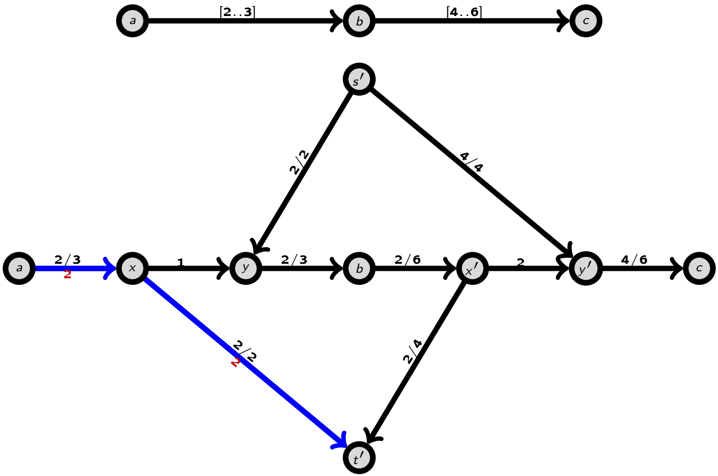
Beispiel (warum (s', y) und (x, t'))



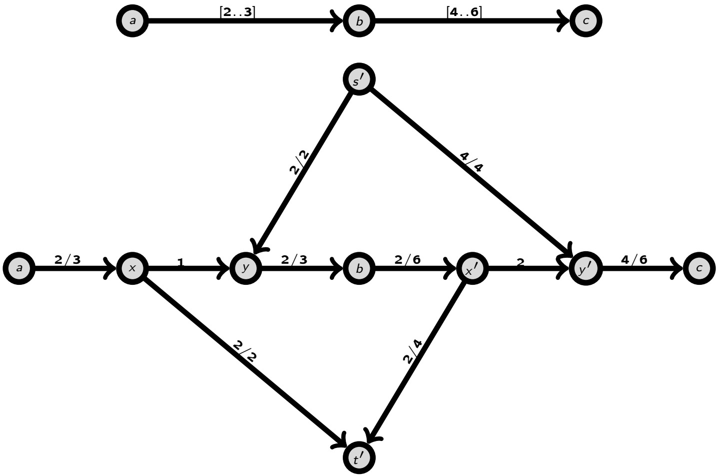
Beispiel (warum (s', y) und (x, t'))



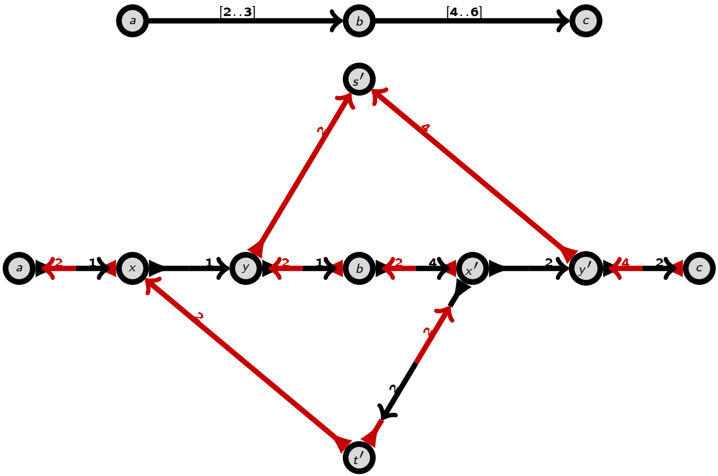
Beispiel (warum (s', y) und (x, t'))



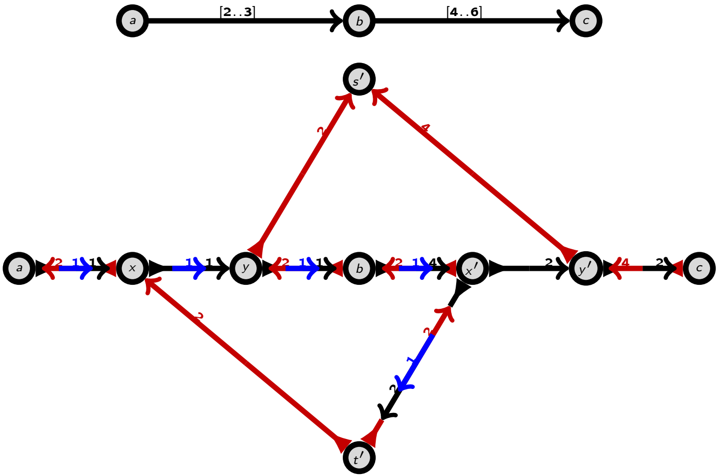
Beispiel (warum (s', y) und (x, t'))



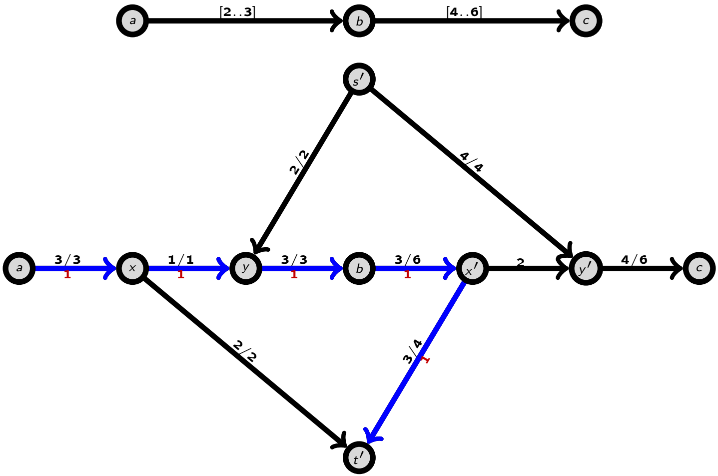
Beispiel (warum (s', y) und (x, t'))



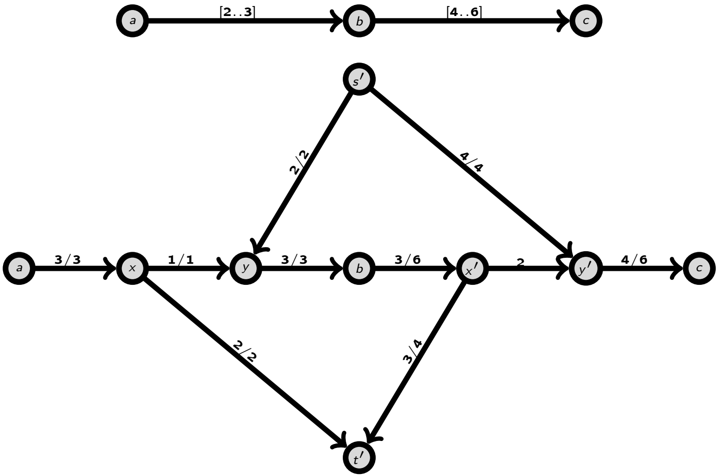
Beispiel (warum (s', y) und (x, t'))



Beispiel (warum (s', y) und (x, t'))

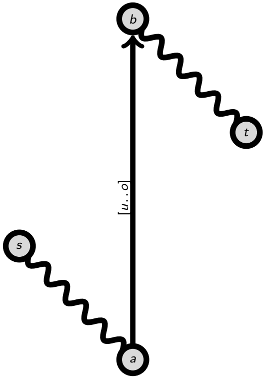


Beispiel (warum (s', y) und (x, t'))



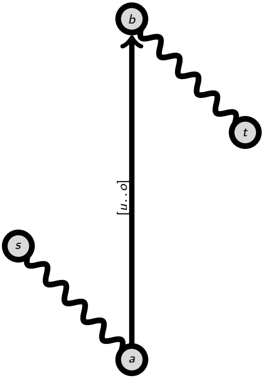
Zeige: $\Longrightarrow$

Zeige: $\Longrightarrow$

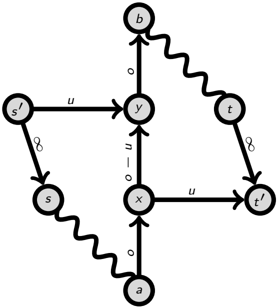


$u \leqslant f(a,b) \leqslant o$

Zeige: $\Longrightarrow$

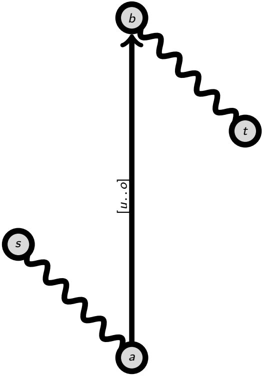


$u \leqslant f(a,b) \leqslant o$

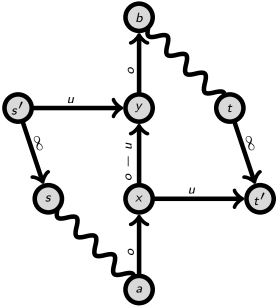


$f(s',y) = f(x,t') = u$

Zeige: $\Longrightarrow$

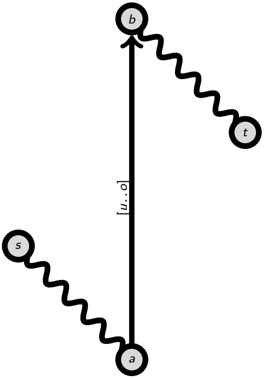


$u \leqslant f(a,b) \leqslant o$

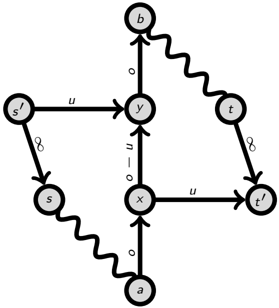


$f(s',y) = f(x,t') = u$
$f(a,x) = f(y,b) = f(a,b)$

Zeige: $\Longrightarrow$

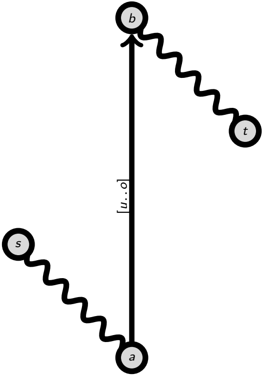


$u \leqslant f(a,b) \leqslant o$

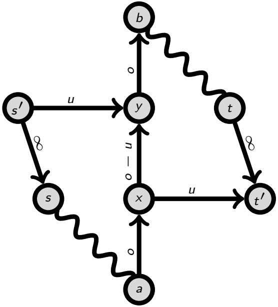


$f(s',y) = f(x,t') = u$
$f(a,x) = f(y,b) = f(a,b)$

Zeige: $\Longrightarrow$



$u \leqslant f(a,b) \leqslant o$



$f(s',y) = f(x,t') = u$
$f(a,x) = f(y,b) = f(a,b)$
$f(x,y) = f(a,b) - u$

Zeige: $\Longleftarrow$

Zeige: $\Longleftarrow$

Zeige: $\Longleftarrow$

- Zeige: Wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert, dann gibt es in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt.

Zeige: $\Longleftarrow$

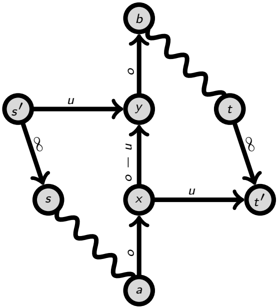
- Zeige: Wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert, dann gibt es in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt.
- Dann gilt: $f(a, x) = f(y, b)$ für jede ursprüngliche Kante (a, b).

Zeige: $\Longleftarrow$

- Zeige: Wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert, dann gibt es in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt.
- Dann gilt: $f(a, x) = f(y, b)$ für jede ursprüngliche Kante (a, b).
- Dann definiert $f(a, b) = f(a, x)$ einen korrekten Fluss auf G.

Zeige: $\Longleftarrow$

- Zeige: Wenn es in G' einen maximalen Fluss gibt, der alle Kanten der Form (s', y) und (x, t') saturiert, dann gibt es in G einen korrekten Fluss, der die Mindestflussbedingung erfüllt.
- Dann gilt: $f(a,x) = f(y,b)$ für jede ursprüngliche Kante (a,b).
- Dann definiert $f(a,b) = f(a,x)$ einen korrekten Fluss auf G.



Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:
  - $f = f''$, denn untere Schranken verringern den Fluss auf G nicht mehr.

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:
  - $f = f''$, denn untere Schranken verringern den Fluss auf G nicht mehr.
  - $f' = f + \sum_{e \in E(G)} c'(e)$

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:
  - $f = f''$, denn untere Schranken verringern den Fluss auf G nicht mehr.
  - $f' = f + \sum_{e \in E(G)} c'(e)$
- Damit haben wir folgendes Verfahren:

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:
  - $f = f''$, denn untere Schranken verringern den Fluss auf G nicht mehr.
  - $f' = f + \sum_{e \in E(G)} c'(e)$
- Damit haben wir folgendes Verfahren:
  1. Bestimme aus G: G', G'', f, f', f''.

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:
  - $f = f''$, denn untere Schranken verringern den Fluss auf G nicht mehr.
  - $f' = f + \sum_{e \in E(G)} c'(e)$
- Damit haben wir folgendes Verfahren:
  1. Bestimme aus G: G', G'', f, f', f''.
  2. Bevorzuge auf G' die Kanten der Form (s', y) und (x, t').

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:
  - $f = f''$, denn untere Schranken verringern den Fluss auf G nicht mehr.
  - $f' = f + \sum_{e \in E(G)} c'(e)$
- Damit haben wir folgendes Verfahren:
  1. Bestimme aus G: G', G'', f, f', f''.
  2. Bevorzuge auf G' die Kanten der Form (s', y) und (x, t').
  3. Falls $f' < f + \sum_{e \in E(G)} c'(e)$ gilt, so gibt es keine Lösung.

Algorithmus

- Sei f (resp. f') der Fluss auf G (resp. G').
- Sei weiter f'' der Fluss auf G ohne die untere Schranke c'. Dann gilt, wenn es eine Lösung für G' gibt:
  - $f = f''$, denn untere Schranken verringern den Fluss auf G nicht mehr.
  - $f' = f + \sum_{e \in E(G)} c'(e)$
- Damit haben wir folgendes Verfahren:
  1. Bestimme aus G: G', G'', f, f', f''.
  2. Bevorzuge auf G' die Kanten der Form (s', y) und (x, t').
  3. Falls $f' < f + \sum_{e \in E(G)} c'(e)$ gilt, so gibt es keine Lösung.
  4. Ansonsten bestimme f aus f', d.h. $f(a, b) = f(a, x)$.

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : c'(e) \leqslant f(e) \leqslant c(e)$ oder $f(e) = 0$.

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : c'(e) \leqslant f(e) \leqslant c(e)$ oder $f(e) = 0$.
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Das Flussproblem mit Alternativen

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c, c')$ mit:
- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $c' : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:
- $\forall e : c'(e) \leqslant f(e) \leqslant c(e)$ oder $f(e) = 0$.
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v) \in E} f((a, v)) = \sum_{(v,a) \in E} f((v, a))$

Ziel: Bestimme, ob es so einen nicht trivialen Fluss gibt. Falls ja, dann maximiere $w(f) = \sum_{(s,v) \in E} f((s, v))$.

Reduktion

Theorem

Zu einem gegeben Flussproblem $G = (V, E, s, t, c, c')$ ist es NP-vollständig zu bestimmen, ob es so einen nicht trivialen Fluss gibt.

Beweis: Übung, b.z.w. Reduktion auf Exact-3-SAT.

Exact-3-SAT

Definition

Eine Boolesche Formel $\mathcal{F}$ ist in Exact-3-KNF:

$$\mathcal{F}(x_1, x_2, \ldots, x_r) \quad = \quad \bigwedge_{i=1}^{k} c_i$$

Exact-3-SAT

Definition

Eine Boolesche Formel $\mathcal{F}$ ist in Exact-3-KNF:

$$\mathcal{F}(x_1, x_2, \ldots, x_r) \quad = \quad \bigwedge_{i=1}^{k} c_i$$

Exact-3-SAT

Definition

Eine Boolesche Formel $\mathcal{F}$ ist in Exact-3-KNF:

$$\mathcal{F}(x_1, x_2, \ldots, x_r) \quad = \quad \bigwedge_{i=1}^{k} c_i$$

$$(Klauseln) \quad c_i \quad = \quad (l_i^1 \vee l_i^2 \vee l_i^3) \qquad \forall\, 1 \leqslant i \leqslant k$$

Exact-3-SAT

Definition

Eine Boolesche Formel $\mathcal{F}$ ist in Exact-3-KNF:

$$\mathcal{F}(x_1, x_2, \ldots, x_r) \quad = \quad \bigwedge_{i=1}^{k} c_i$$

$$(Klauseln) \quad c_i \quad = \quad (l_i^1 \vee l_i^2 \vee l_i^3) \qquad \forall\, 1 \leqslant i \leqslant k$$

$$(Literale) \quad l_i^j \quad = \quad \left\{ \begin{array}{rl} \neg x_l & \text{oder} \\ x_l & \text{für ein } l : 1 \leqslant l \leqslant r \end{array} \right\} \quad \begin{array}{l} \forall\, 1 \leqslant i \leqslant k \text{ und} \\ \forall\, 1 \leqslant j \leqslant 3 \end{array}$$

Exact-3-SAT

Definition

Eine Boolesche Formel $\mathcal{F}$ ist in Exact-3-KNF:

$$\mathcal{F}(x_1, x_2, \ldots, x_r) = \bigwedge_{i=1}^{k} c_i$$

$$(\textit{Klauseln}) \quad c_i = (l_i^1 \vee l_i^2 \vee l_i^3) \qquad \forall\, 1 \leqslant i \leqslant k$$

$$(\textit{Literale}) \quad l_i^j = \left\{ \begin{array}{rl} \neg x_l & \text{oder} \\ x_l & \text{für ein } l : 1 \leqslant l \leqslant r \end{array} \right\} \quad \begin{array}{l} \forall\, 1 \leqslant i \leqslant k \text{ und} \\ \forall\, 1 \leqslant j \leqslant 3 \end{array}$$

Eine Belegung ist eine Funktion $W : \{x_1, x_2, \ldots, x_r\} \mapsto \{0, 1\}$.

Exact-3-SAT

Definition

Eine Boolesche Formel $\mathcal{F}$ ist in Exact-3-KNF:

$$\mathcal{F}(x_1, x_2, ..., x_r) = \bigwedge_{i=1}^{k} c_i$$

$$(Klauseln) \quad c_i = (l_i^1 \vee l_i^2 \vee l_i^3) \qquad \forall\, 1 \leqslant i \leqslant k$$

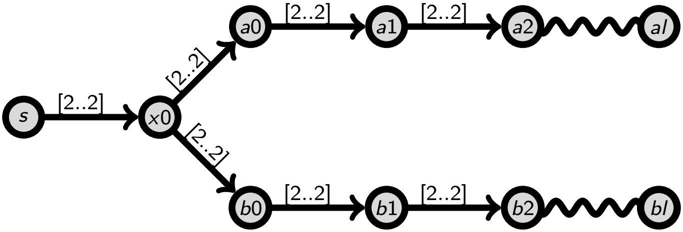
$$(Literale) \quad l_i^j = \left\{ \begin{array}{rl} \neg x_l & \text{oder} \\ x_l & \text{für ein } l : 1 \leqslant l \leqslant r \end{array} \right\} \quad \begin{array}{l} \forall\, 1 \leqslant i \leqslant k \text{ und} \\ \forall\, 1 \leqslant j \leqslant 3 \end{array}$$

Eine Belegung ist eine Funktion $W : \{x_1, x_2, ..., x_r\} \mapsto \{0, 1\}$.

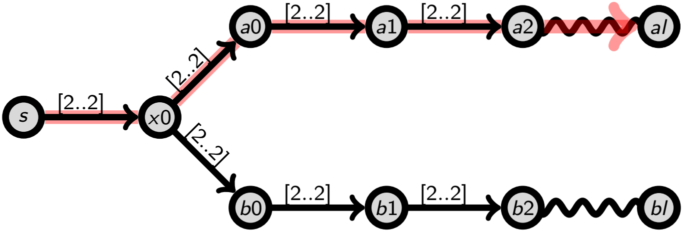
Theorem (Exakt-3-SAT)

Es ist NP-vollständig, festzustellen, ob es für $\mathcal{F}$ aus Exact-3-KNF eine erfüllende Belegung gibt, bei der in jeder Klausel genau ein Literal “true” ist.

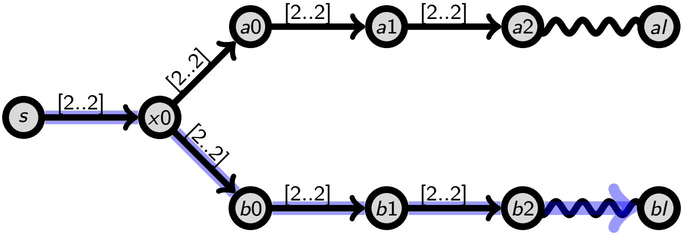
Erste Variable x_0



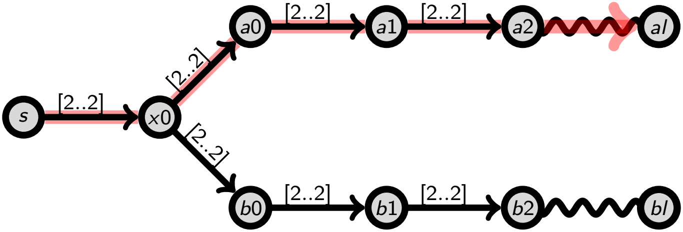
Erste Variable x_0



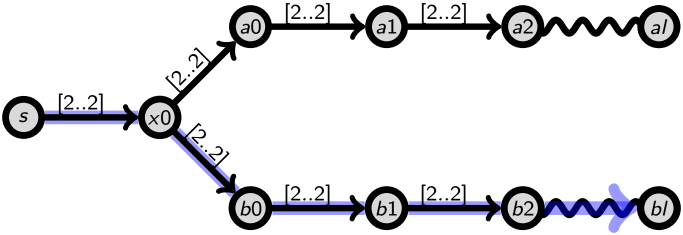
Erste Variable x_0



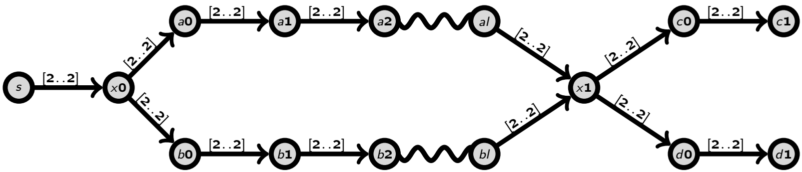
Erste Variable x_0



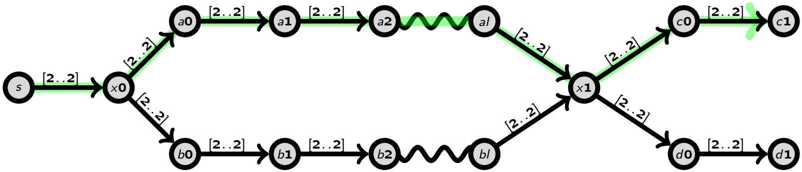
Erste Variable x_0



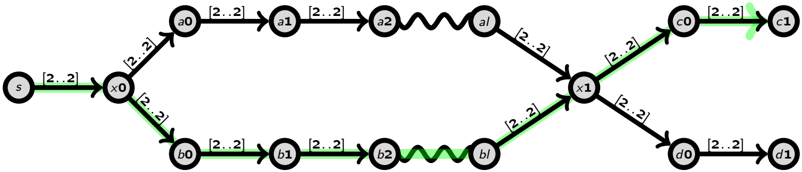
Zweite Variable x_1



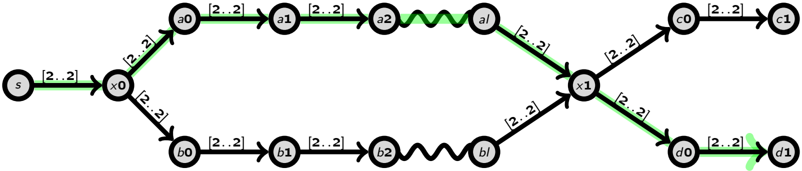
Zweite Variable x_1



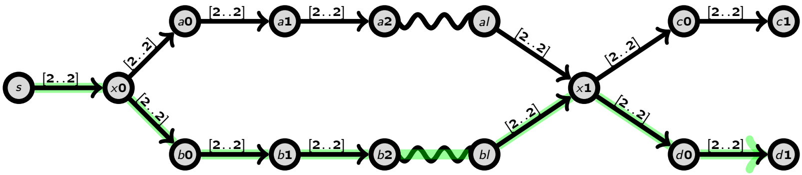
Zweite Variable x_1



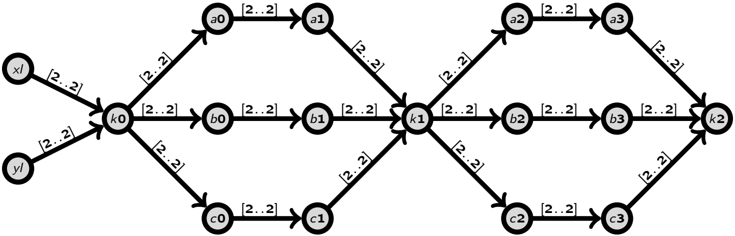
Zweite Variable x_1



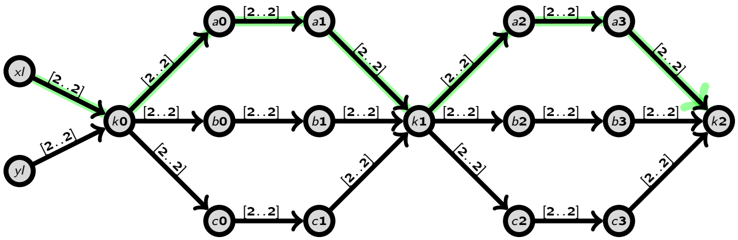
Zweite Variable x_1



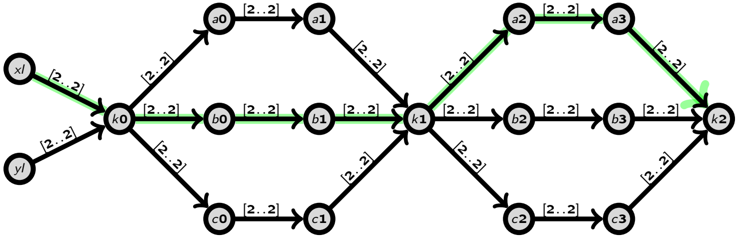
Erste zwei Klauseln k_0 und k_1



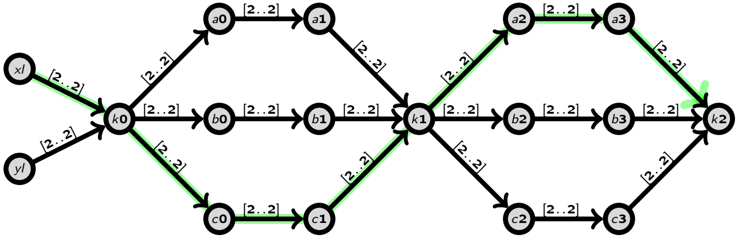
Erste zwei Klauseln k_0 und k_1



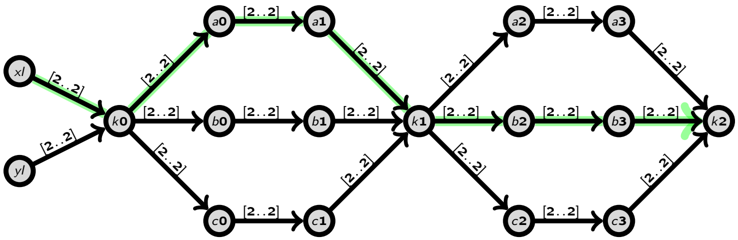
Erste zwei Klauseln k_0 und k_1



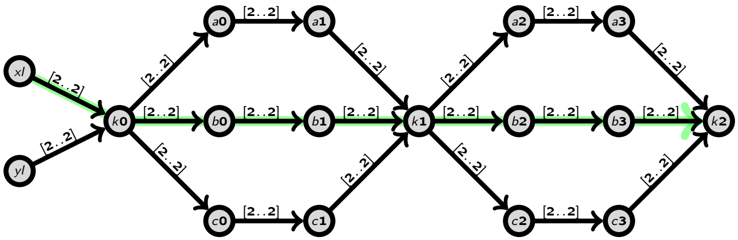
Erste zwei Klauseln k_0 und k_1



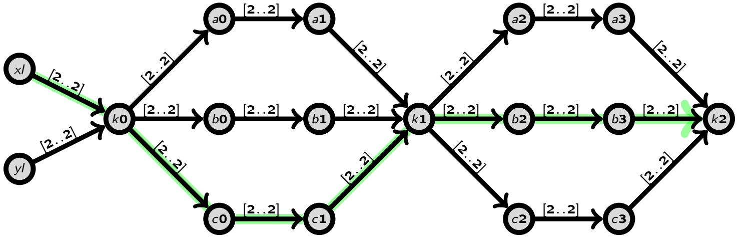
Erste zwei Klauseln k_0 und k_1



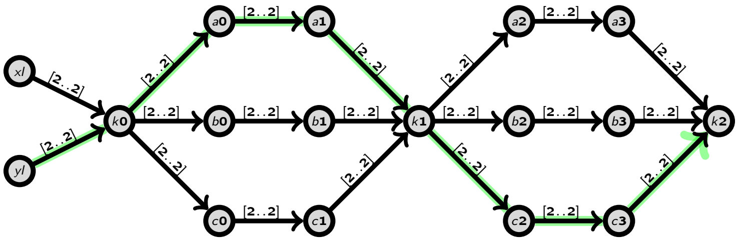
Erste zwei Klauseln k_0 und k_1



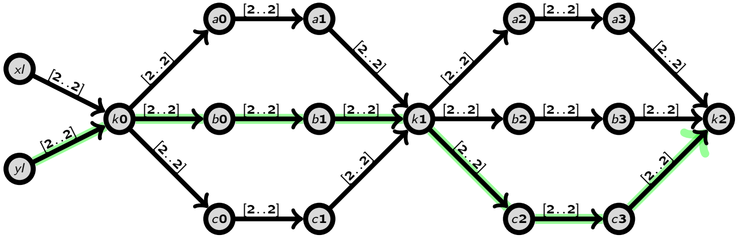
Erste zwei Klauseln k_0 und k_1



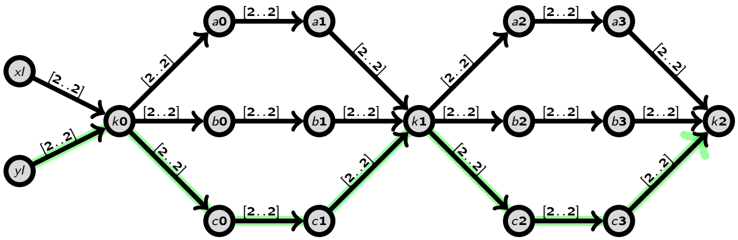
Erste zwei Klauseln k_0 und k_1



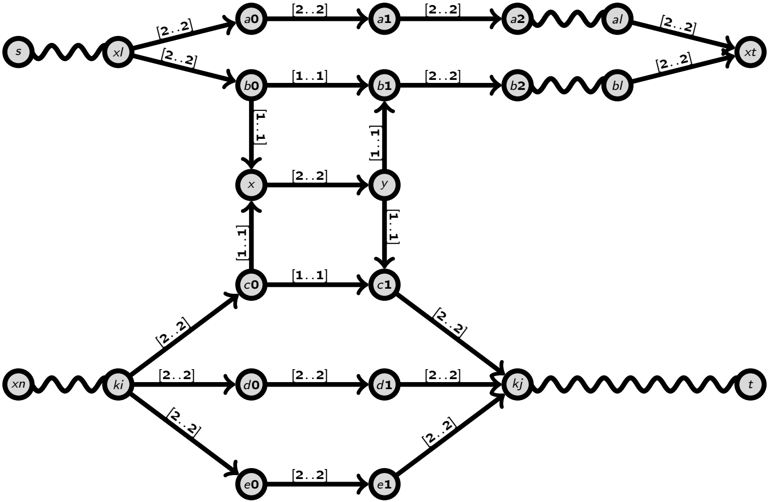
Erste zwei Klauseln k_0 und k_1



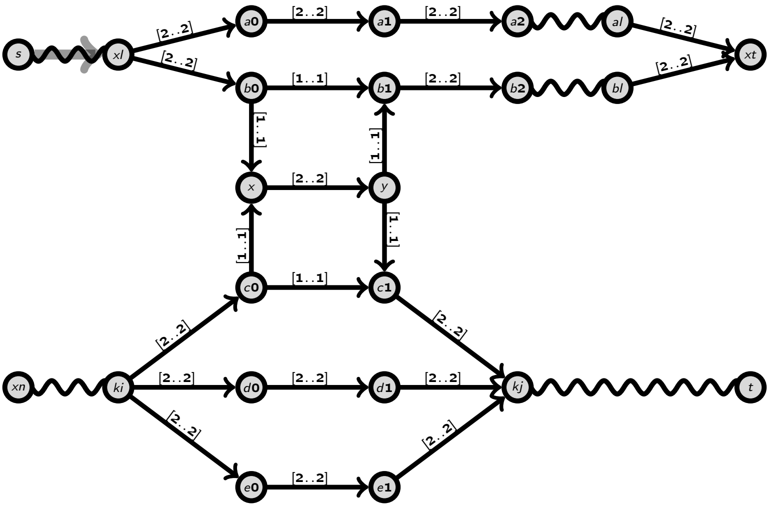
Erste zwei Klauseln k_0 und k_1



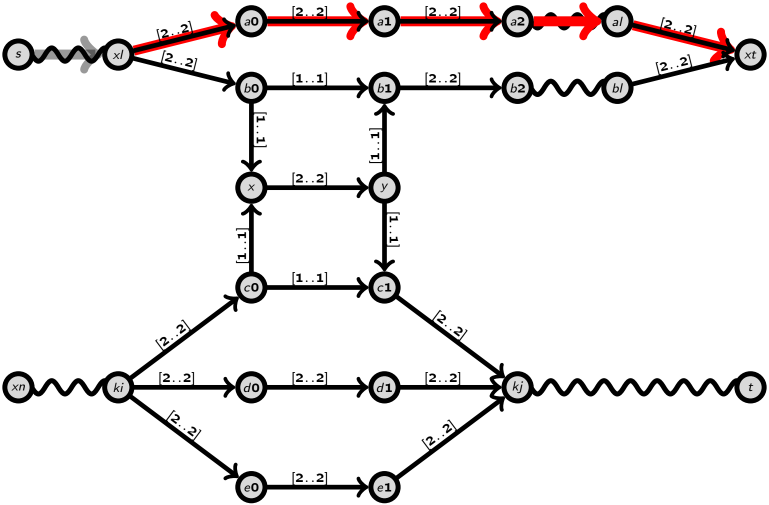
Anpassung: Variable x_l in Klausel k_i



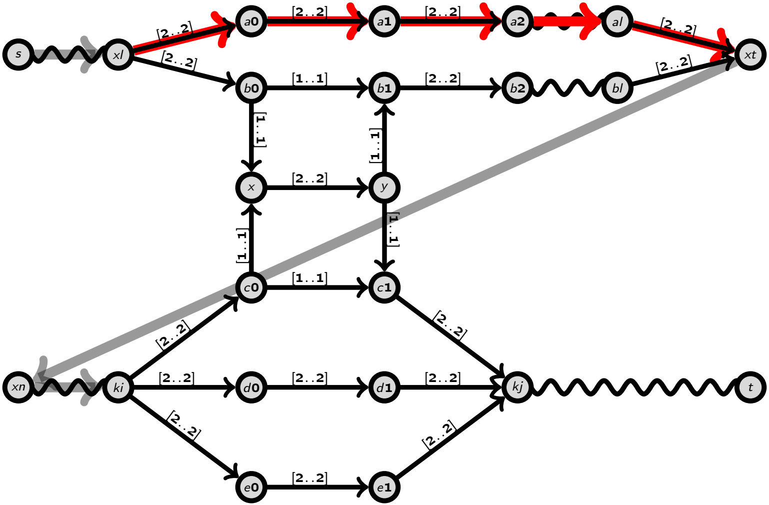
Anpassung: Variable x_l in Klausel k_i



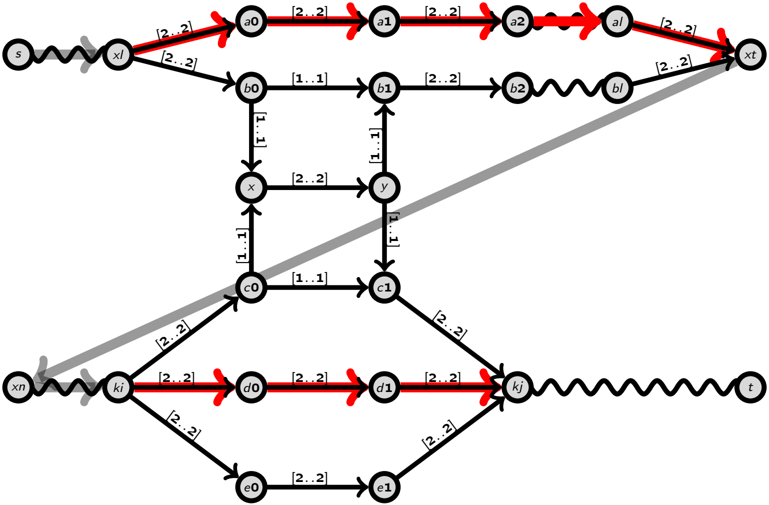
Anpassung: Variable x_l in Klausel k_i



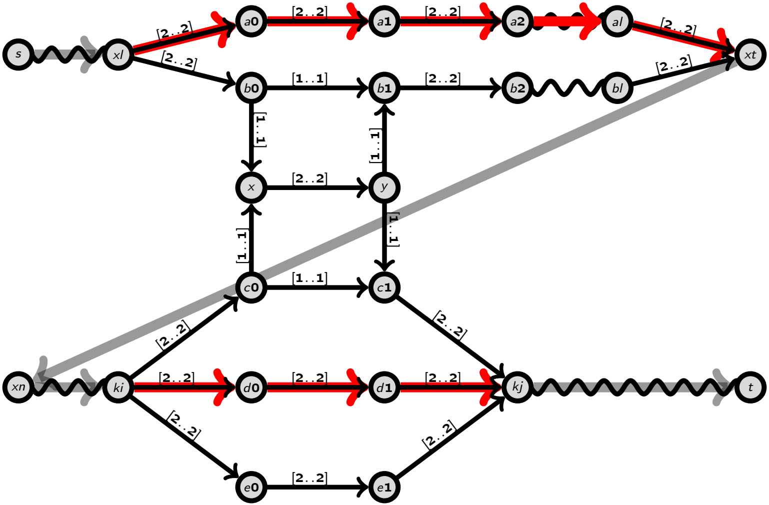
Anpassung: Variable x_l in Klausel k_i



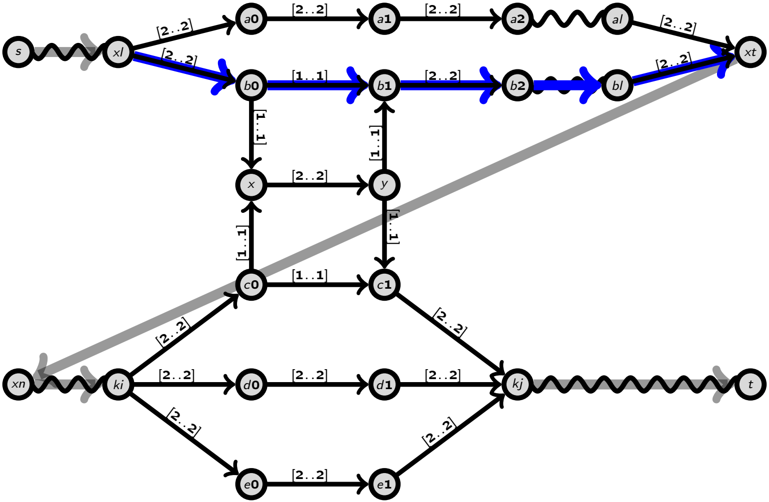
Anpassung: Variable x_l in Klausel k_i



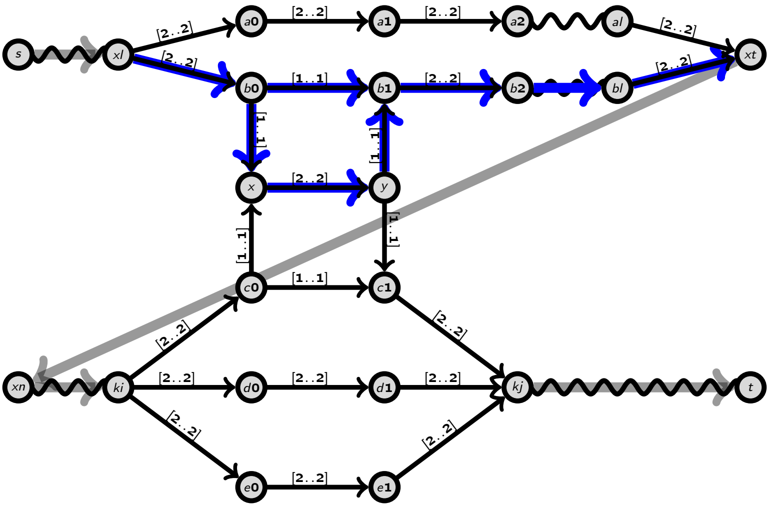
Anpassung: Variable x_l in Klausel k_i



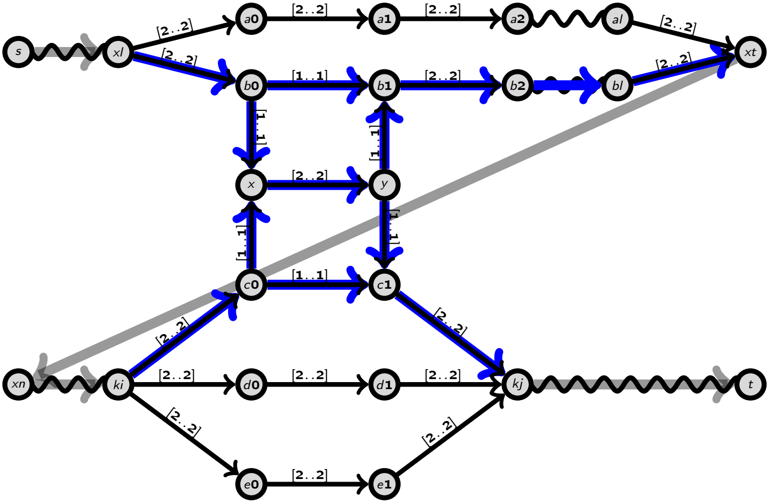
Anpassung: Variable x_l in Klausel k_i



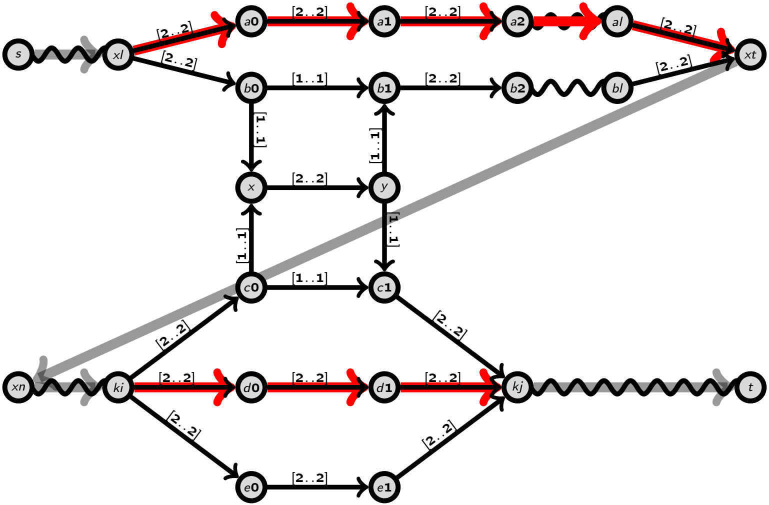
Anpassung: Variable x_l in Klausel k_i



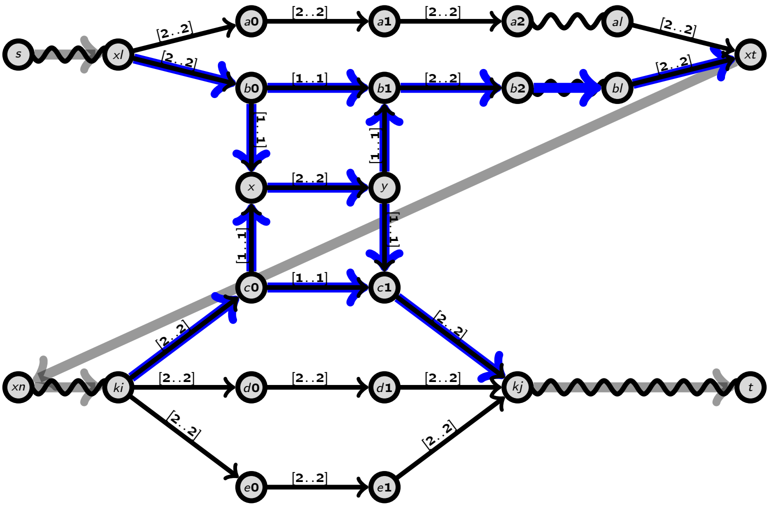
Anpassung: Variable x_l in Klausel k_i



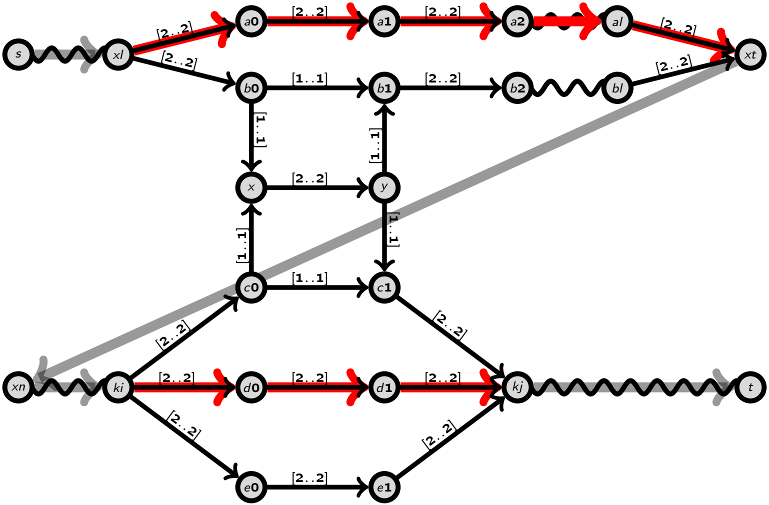
Anpassung: Variable x_l in Klausel k_i



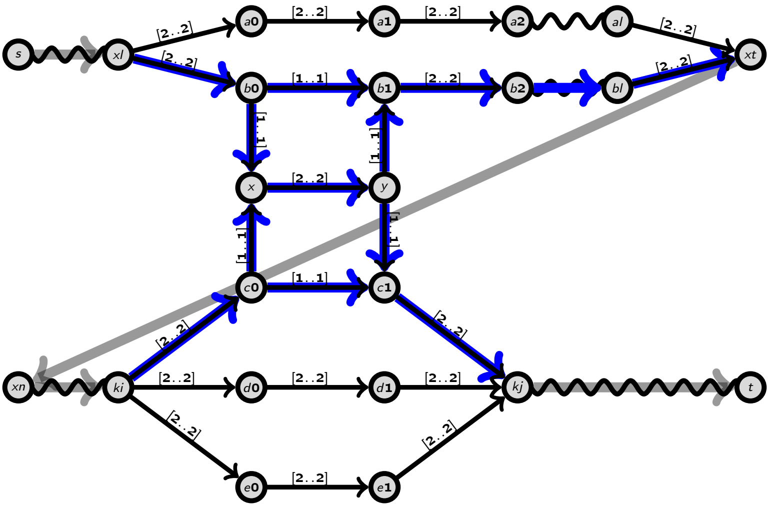
Anpassung: Variable x_l in Klausel k_i



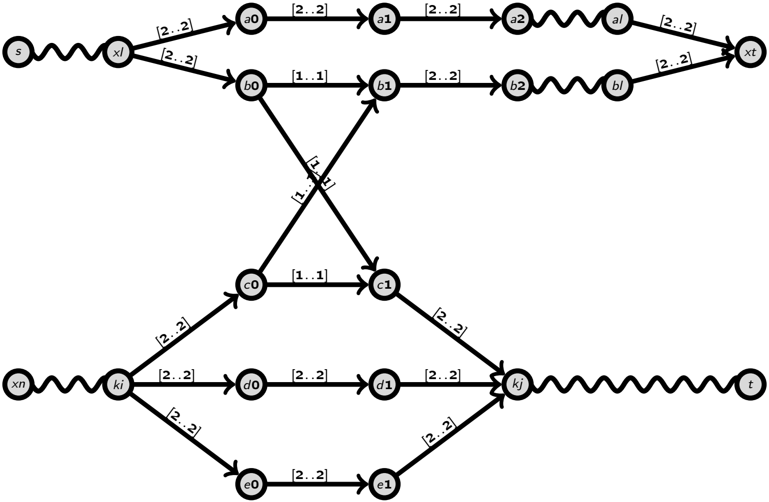
Anpassung: Variable x_l in Klausel k_i



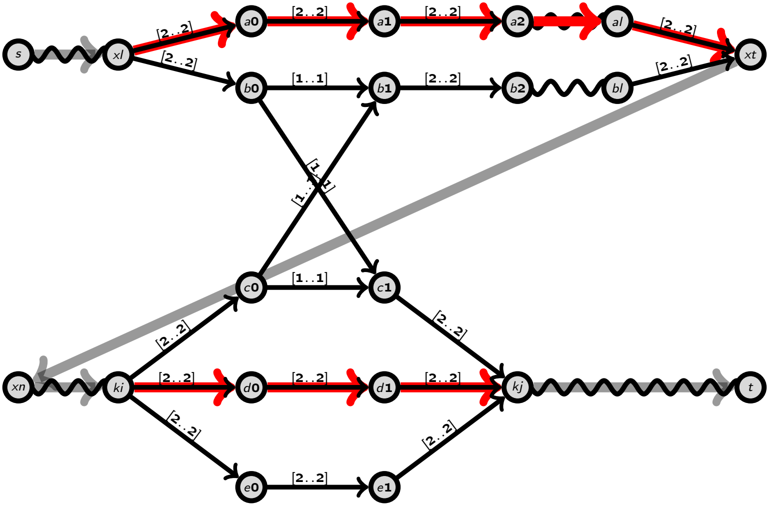
Anpassung: Variable x_l in Klausel k_i



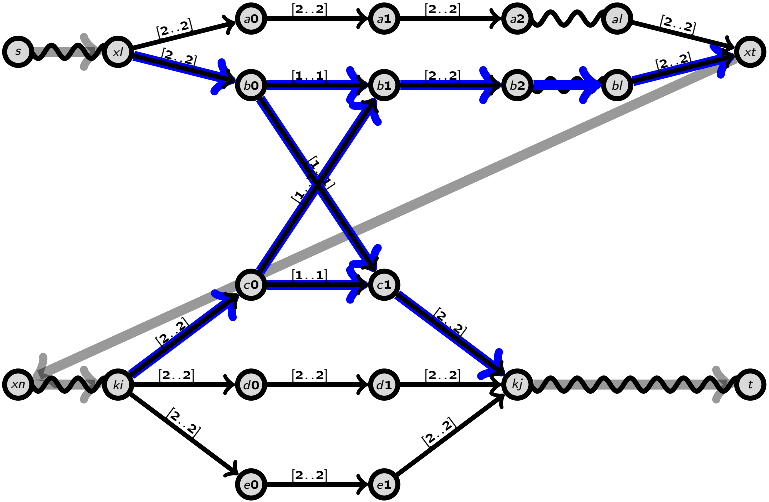
Alternative Anpassung: Variable x_l in Klausel k_i



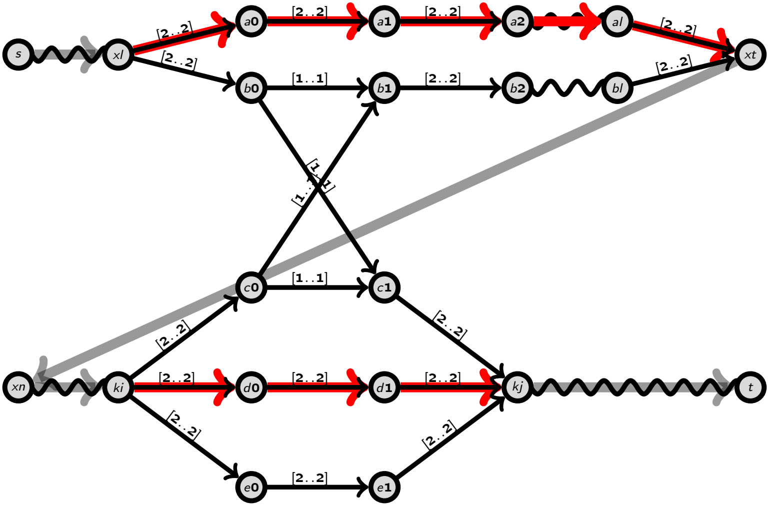
Alternative Anpassung: Variable x_l in Klausel k_i



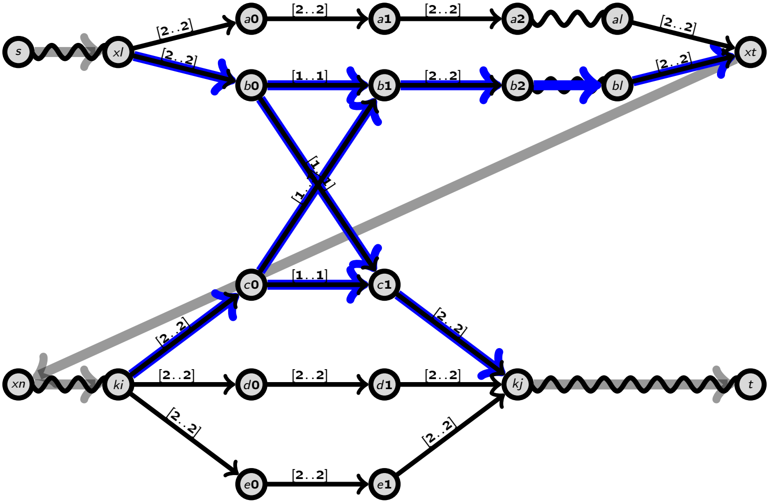
Alternative Anpassung: Variable x_l in Klausel k_i



Alternative Anpassung: Variable x_l in Klausel k_i



Alternative Anpassung: Variable x_l in Klausel k_i



Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.
2. Für jede Klausel konstruiere einen Baustein, wie oben.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.
2. Für jede Klausel konstruiere einen Baustein, wie oben.
   - Der erste Zweig entspricht dem ersten Literal in der Klausel.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.
2. Für jede Klausel konstruiere einen Baustein, wie oben.
   - Der erste Zweig entspricht dem ersten Literal in der Klausel.
   - Die weiteren Zweige dem zweiten und dem dritten Literal in der Klausel.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.
2. Für jede Klausel konstruiere einen Baustein, wie oben.
   - Der erste Zweig entspricht dem ersten Literal in der Klausel.
   - Die weiteren Zweige dem zweiten und dem dritten Literal in der Klausel.
3. Hänge alle Bausteine für die Variablen und Klauseln hintereinander.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.
2. Für jede Klausel konstruiere einen Baustein, wie oben.
   - Der erste Zweig entspricht dem ersten Literal in der Klausel.
   - Die weiteren Zweige dem zweiten und dem dritten Literal in der Klausel.
3. Hänge alle Bausteine für die Variablen und Klauseln hintereinander.
4. Für jedes Auftreten eines Literals in einer Klausel mache die obige Anpassung.

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.
2. Für jede Klausel konstruiere einen Baustein, wie oben.
   - Der erste Zweig entspricht dem ersten Literal in der Klausel.
   - Die weiteren Zweige dem zweiten und dem dritten Literal in der Klausel.
3. Hänge alle Bausteine für die Variablen und Klauseln hintereinander.
4. Für jedes Auftreten eines Literals in einer Klausel mache die obige Anpassung.
5. Falls es eine Belegung der Variablen gibt, die die Formel erfüllt, dann:

Konstruktion

1. Für jede Variable konstruiere einen Baustein, wie oben.
   - Der obere Zweig entspricht der Variablen selber.
   - Der untere Zweig entspricht der negierten Variablen.
2. Für jede Klausel konstruiere einen Baustein, wie oben.
   - Der erste Zweig entspricht dem ersten Literal in der Klausel.
   - Die weiteren Zweige dem zweiten und dem dritten Literal in der Klausel.
3. Hänge alle Bausteine für die Variablen und Klauseln hintereinander.
4. Für jedes Auftreten eines Literals in einer Klausel mache die obige Anpassung.
5. Falls es eine Belegung der Variablen gibt, die die Formel erfüllt, dann:
   - geht ein Fluss von 2 durch jeweils den Zweig, der der Belegung der Variablen entspricht.

Motivation

- Benutzung der Kanten (Transportwege) im allgemeinen nicht umsonst.

Motivation

- Benutzung der Kanten (Transportwege) im allgemeinen nicht umsonst.
- Daher sollten wir die Kosten minimieren.

Motivation

- Benutzung der Kanten (Transportwege) im allgemeinen nicht umsonst.
- Daher sollten wir die Kosten minimieren.
- Kosten sind minimal, wenn der Fluss Null ist.

Motivation

- Benutzung der Kanten (Transportwege) im allgemeinen nicht umsonst.
- Daher sollten wir die Kosten minimieren.
- Kosten sind minimal, wenn der Fluss Null ist.
- Daher suchen wir:

  Kostenminimalen Fluss mit Wert W.

Motivation

- Benutzung der Kanten (Transportwege) im allgemeinen nicht umsonst.
- Daher sollten wir die Kosten minimieren.
- Kosten sind minimal, wenn der Fluss Null ist.
- Daher suchen wir:

  Kostenminimalen Fluss mit Wert W.
- Wichtig: G sollte keine Kreise mit negativen Kosten (Gewichtssumme) enthalten.

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $l : E \mapsto \mathbb{Z}$ und $W \in \mathbb{N}$

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $l : E \mapsto \mathbb{Z}$ und $W \in \mathbb{N}$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $l : E \mapsto \mathbb{Z}$ und $W \in \mathbb{N}$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : f(e) \leqslant c(e)$.

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $l : E \mapsto \mathbb{Z}$ und $W \in \mathbb{N}$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : f(e) \leqslant c(e)$.
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Das Flussproblem mit Kosten

Definition (Min-Cost-Flow-Problem)

Eingabe: $G = (V, E, s, t, c, l), W$ mit:

- (V, E) ist ein gerichteter Graph ($n = |V|, m = |E|$)
- $s, t \in V$ mit $s \neq t$
- $c : E \mapsto \mathbb{N}^+$
- $l : E \mapsto \mathbb{Z}$ und $W \in \mathbb{N}$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : f(e) \leqslant c(e)$.
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v) \in E} f((a, v)) = \sum_{(v,a) \in E} f((v, a))$

Ziel: Bestimme Fluss f $w(f) = W$ und
minimalen $l(f) = \sum_{e \in E} f(e) \cdot l(e)$.
D.h. $l(f) = min\{l(g) \mid g$ ist Fluss mit $w(g) = W\}$.

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:
  1. Falls erstmalig $w(f) \geqslant W$ gilt, dann breche ab.

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:
  1. Falls erstmalig $w(f) \geqslant W$ gilt, dann breche ab.
  2. Sei p der Wert der letzten Erweiterung.

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:
  1. Falls erstmalig $w(f) \geqslant W$ gilt, dann breche ab.
  2. Sei p der Wert der letzten Erweiterung.
  3. Ersetze den letzten Fluss durch eine Fluss mit dem Wert $p - (w(f) - W)$.

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:
  1. Falls erstmalig $w(f) \geqslant W$ gilt, dann breche ab.
  2. Sei p der Wert der letzten Erweiterung.
  3. Ersetze den letzten Fluss durch eine Fluss mit dem Wert $p - (w(f) - W)$.
- Alternativ kann auch wie folgt vorgegangen werden:

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:
  1. Falls erstmalig $w(f) \geqslant W$ gilt, dann breche ab.
  2. Sei p der Wert der letzten Erweiterung.
  3. Ersetze den letzten Fluss durch eine Fluss mit dem Wert $p - (w(f) - W)$.
- Alternativ kann auch wie folgt vorgegangen werden:
  1. Erzeuge neue Quelle s'.

Beobachtungen

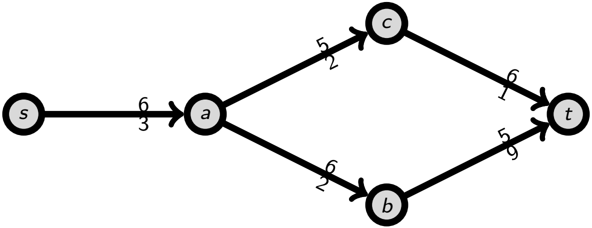
- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:
  1. Falls erstmalig $w(f) \geqslant W$ gilt, dann breche ab.
  2. Sei p der Wert der letzten Erweiterung.
  3. Ersetze den letzten Fluss durch eine Fluss mit dem Wert $p - (w(f) - W)$.
- Alternativ kann auch wie folgt vorgegangen werden:
  1. Erzeuge neue Quelle s'.
  2. Füge Kante $e' = (s', s)$ mit $c(e') = W$ hinzu.

Beobachtungen

- Falls $W = 1$, so entspricht das dem kürzesten Wege Problem.
- Falls $l(e) = 0$ für alle $e \in E$, dann können die obigen Algorithmen leicht zur Lösung des Problems adaptiert werden:
  1. Falls erstmalig $w(f) \geqslant W$ gilt, dann breche ab.
  2. Sei p der Wert der letzten Erweiterung.
  3. Ersetze den letzten Fluss durch eine Fluss mit dem Wert $p - (w(f) - W)$.
- Alternativ kann auch wie folgt vorgegangen werden:
  1. Erzeuge neue Quelle s'.
  2. Füge Kante $e' = (s', s)$ mit $c(e') = W$ hinzu.
  3. Damit wird der maximale Fluss durche W begrenzt.

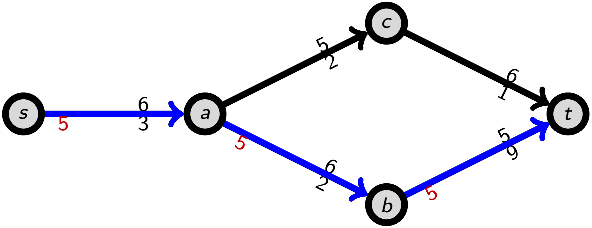
Idee (am Beispiel)

- Kantenbeschriftung:



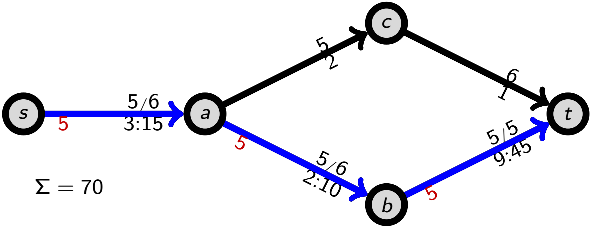
Idee (am Beispiel)

- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und



Idee (am Beispiel)

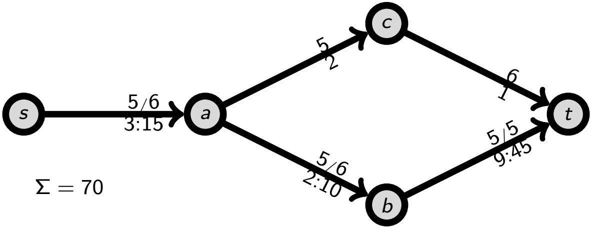
- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten



$\Sigma = 70$

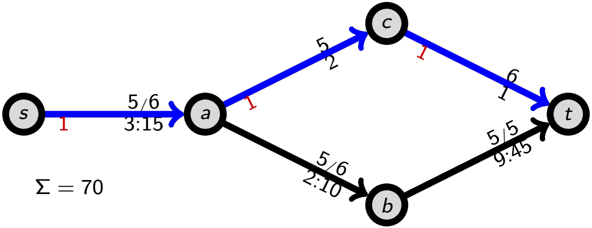
Idee (am Beispiel)

- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten
- Bestimme erst einen Fluss (mit Wert $W = 6$).



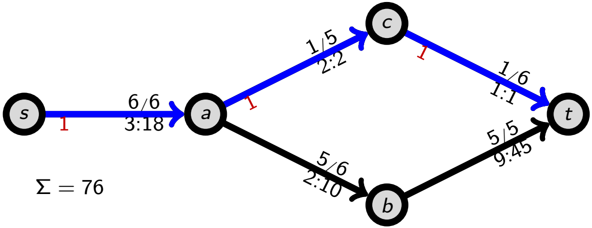
Idee (am Beispiel)

- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten
- Bestimme erst einen Fluss (mit Wert $W = 6$).



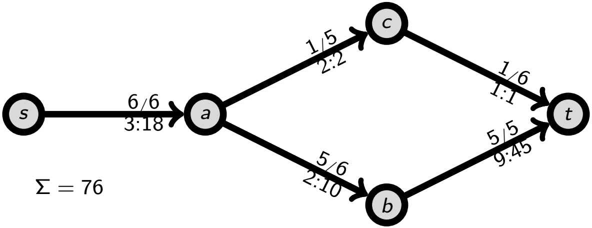
Idee (am Beispiel)

- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten
- Bestimme erst einen Fluss (mit Wert $W = 6$).



Idee (am Beispiel)

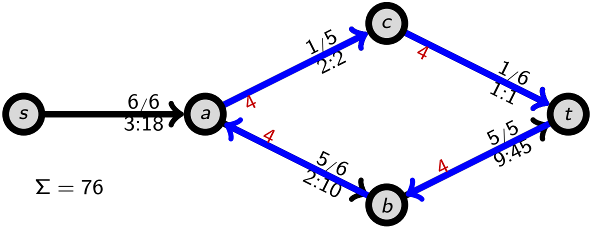
- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten
- Bestimme erst einen Fluss (mit Wert $W = 6$).
- Der Fluss muss nicht kostenoptimal sein.



$\Sigma = 76$

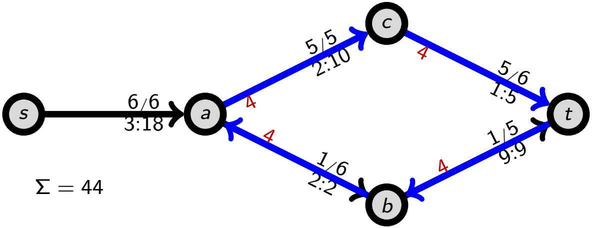
Idee (am Beispiel)

- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten
- Bestimme erst einen Fluss (mit Wert $W = 6$).
- Der Fluss muss nicht kostenoptimal sein.
- Verbessere die Kosten.



Idee (am Beispiel)

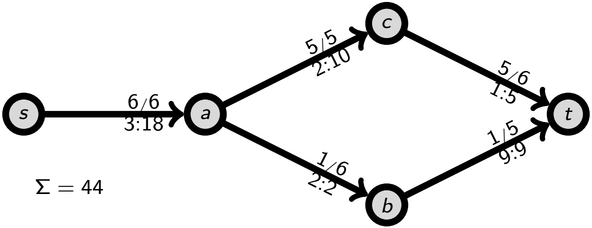
- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten
- Bestimme erst einen Fluss (mit Wert $W = 6$).
- Der Fluss muss nicht kostenoptimal sein.
- Verbessere die Kosten.



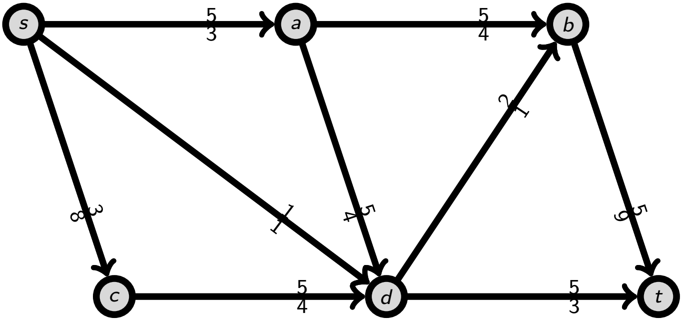
$\Sigma = 44$

Idee (am Beispiel)

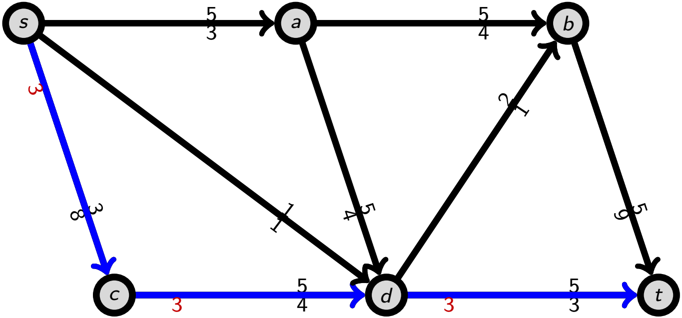
- Kantenbeschriftung:
  - oben: Fluss/MaxFluss und
  - darunter Kosten:Aktuelle Kosten
- Bestimme erst einen Fluss (mit Wert $W = 6$).
- Der Fluss muss nicht kostenoptimal sein.
- Verbessere die Kosten.



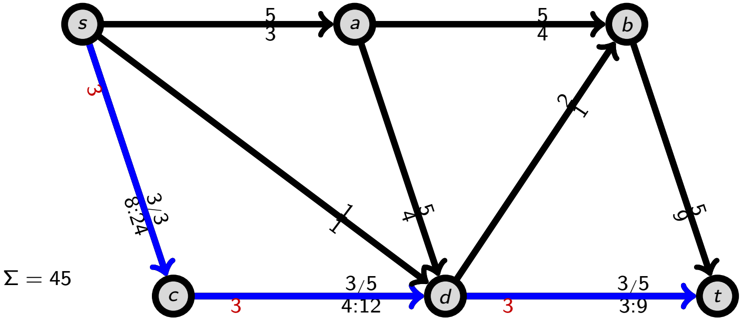
Beispiel ($W = 5$)



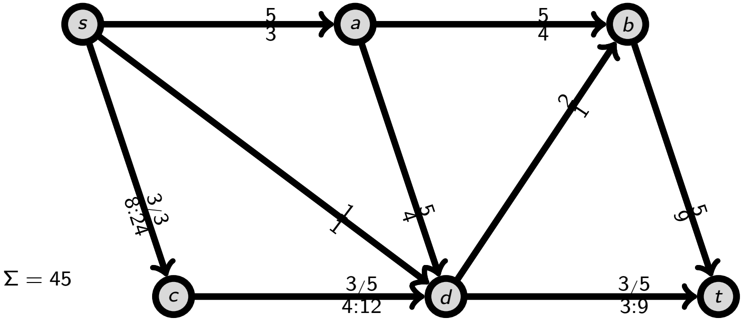
Beispiel ($W = 5$)



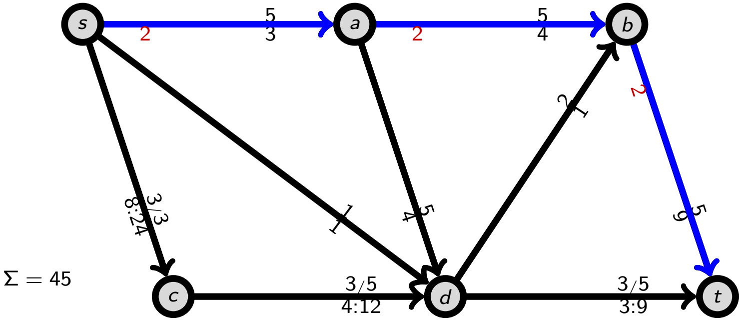
Beispiel ($W = 5$)



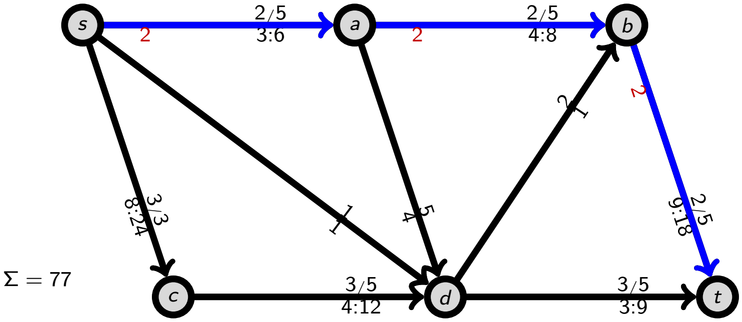
Beispiel ($W = 5$)



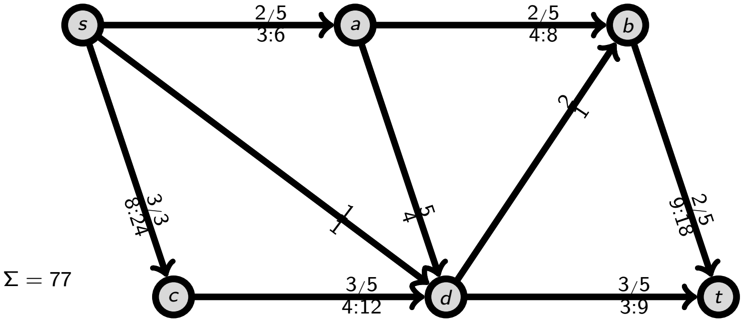
Beispiel $(W = 5)$



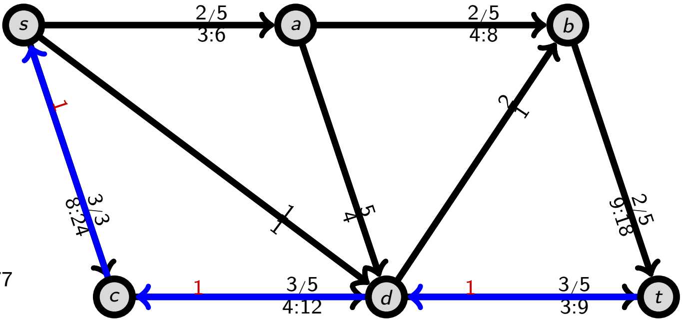
Beispiel ($W = 5$)



Beispiel ($W = 5$)

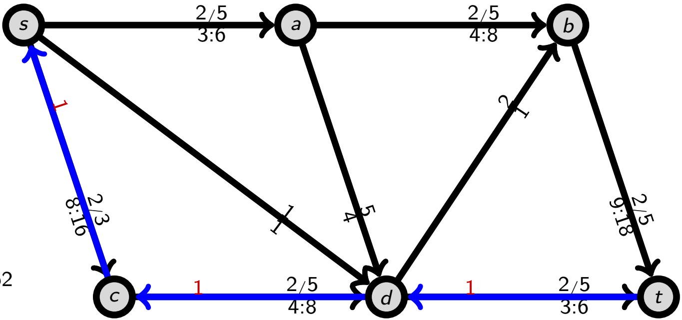


Beispiel ($W = 5$)



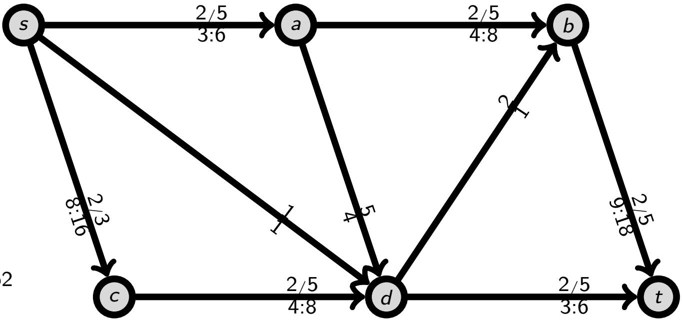
$\Sigma = 77$

Beispiel ($W = 5$)



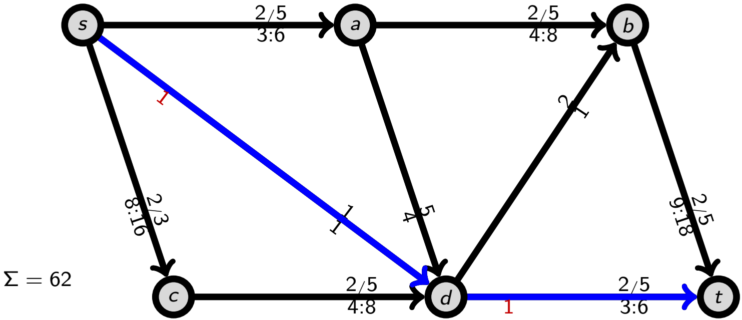
$\Sigma = 62$

Beispiel ($W = 5$)



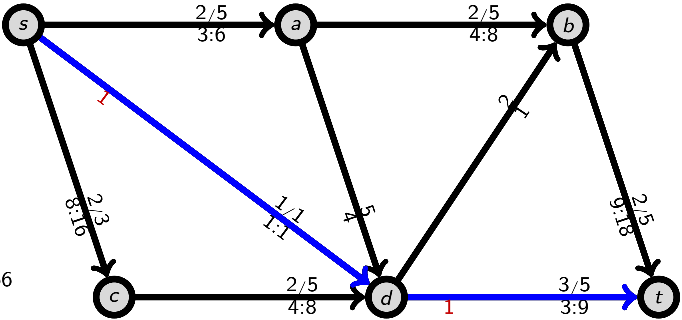
$\Sigma = 62$

Beispiel ($W = 5$)

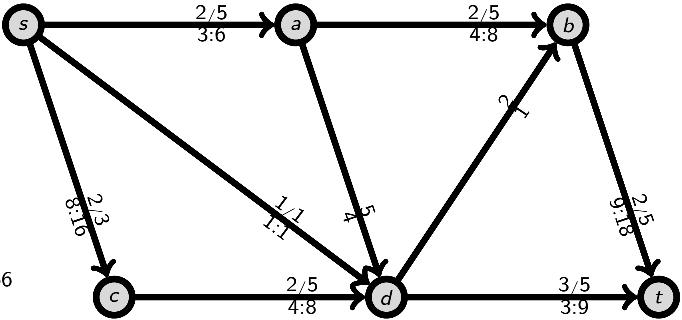


Beispiel ($W = 5$)

$\Sigma = 66$

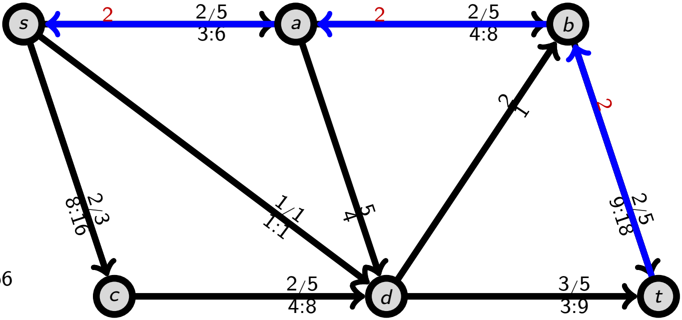


Beispiel ($W = 5$)



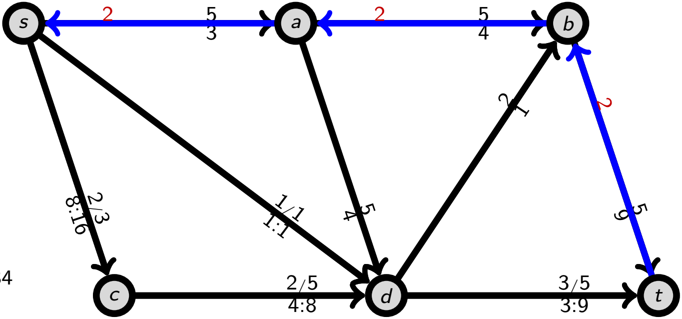
$\Sigma = 66$

Beispiel ($W = 5$)



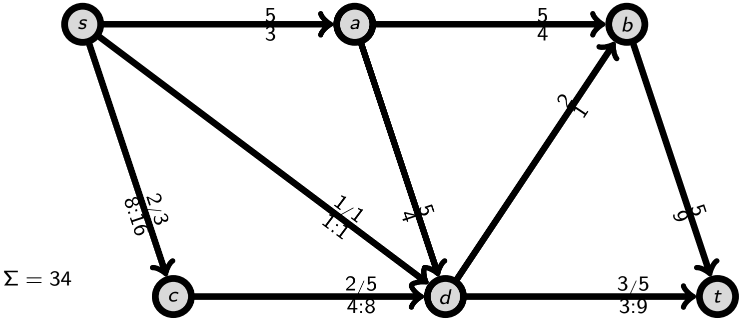
$\Sigma = 66$

Beispiel ($W = 5$)

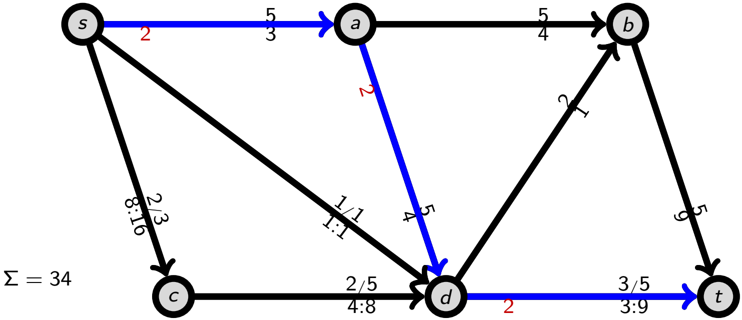


$\Sigma = 34$

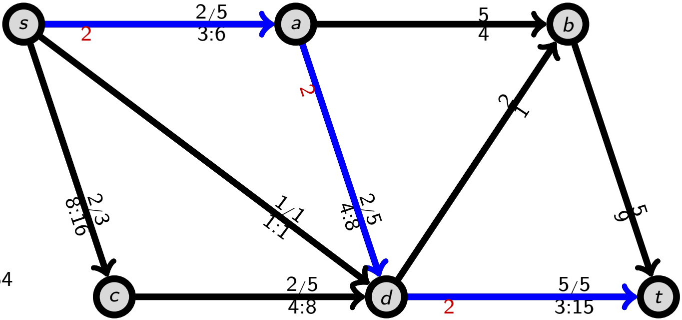
Beispiel ($W = 5$)



Beispiel ($W = 5$)

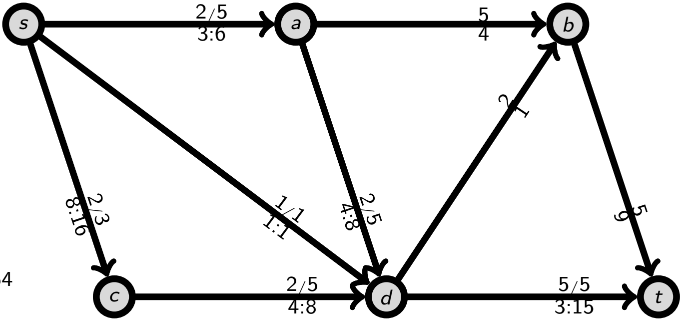


Beispiel $(W = 5)$



$\Sigma = 54$

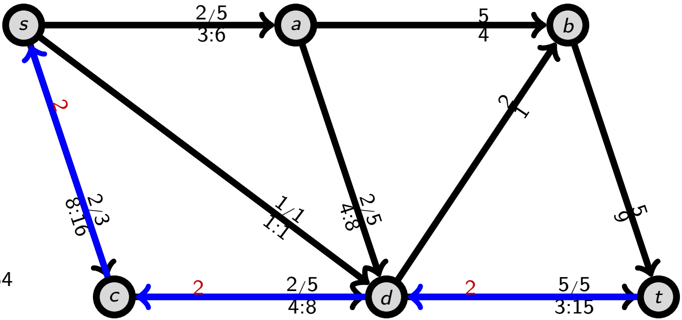
Beispiel ($W = 5$)



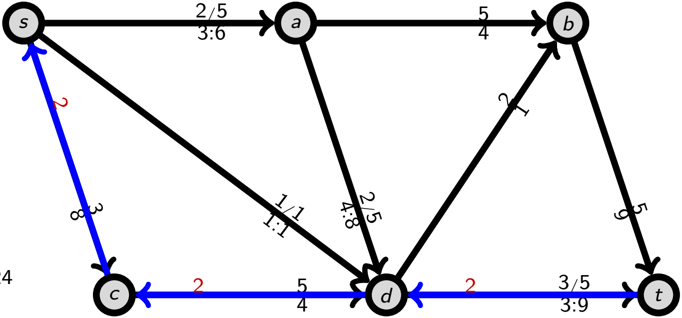
$\Sigma = 54$

Beispiel ($W = 5$)

$\Sigma = 54$

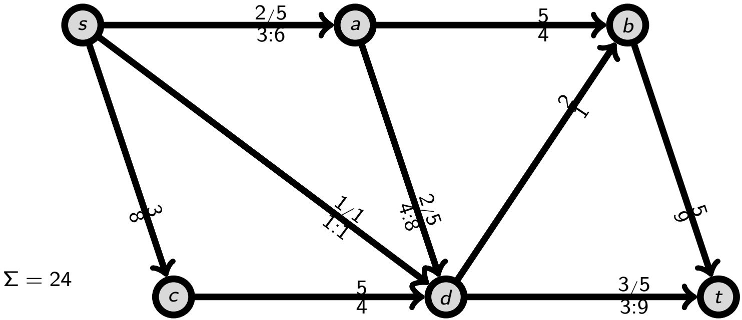


Beispiel ($W = 5$)

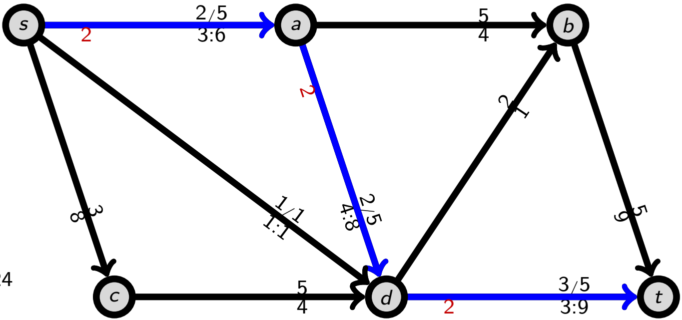


$\Sigma = 24$

Beispiel ($W = 5$)

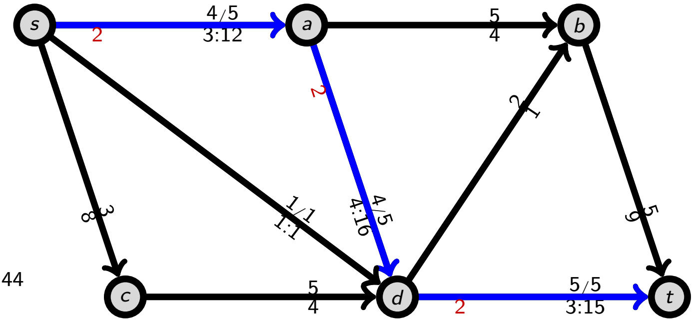


Beispiel ($W = 5$)



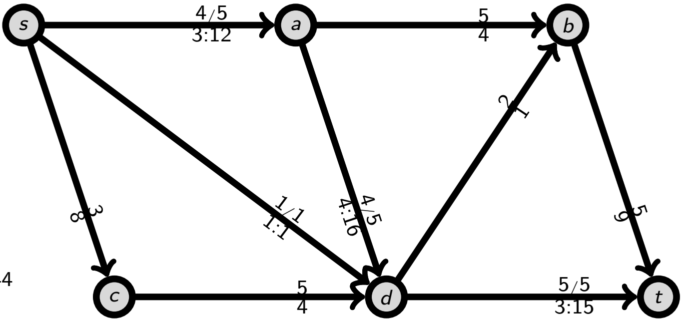
$\Sigma = 24$

Beispiel ($W = 5$)



$\Sigma = 44$

Beispiel ($W = 5$)



$\Sigma = 44$

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.
    - Für f' und f gilt die Flusserhaltung.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.
    - Für f' und f gilt die Flusserhaltung.
    - Und $f_{out}(s) = f'_{out}(s) = f_{in}(t) = f'_{in}(t)$.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.
    - Für f' und f gilt die Flusserhaltung.
    - Und $f_{out}(s) = f'_{out}(s) = f_{in}(t) = f'_{in}(t)$.
    - Damit gilt: Falls $f(a, b) \neq f'(a, b)$, dann gibt es $c \in V \setminus \{a, b\}$ mit: $f(a, c) \neq f'(a, c)$.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.
    - Für f' und f gilt die Flusserhaltung.
    - Und $f_{out}(s) = f'_{out}(s) = f_{in}(t) = f'_{in}(t)$.
    - Damit gilt: Falls $f(a,b) \neq f'(a,b)$, dann gibt es $c \in V \setminus \{a,b\}$ mit: $f(a,c) \neq f'(a,c)$.
    - Damit gibt es mindestens einen Kreis mit Fluss g über Kanten e mit $f(e) \neq f'(e)$.

Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.
    - Für f' und f gilt die Flusserhaltung.
    - Und $f_{out}(s) = f'_{out}(s) = f_{in}(t) = f'_{in}(t)$.
    - Damit gilt: Falls $f(a,b) \neq f'(a,b)$, dann gibt es $c \in V \setminus \{a,b\}$ mit: $f(a,c) \neq f'(a,c)$.
    - Damit gibt es mindestens einen Kreis mit Fluss g über Kanten e mit $f(e) \neq f'(e)$.
  - Dieser zyklische Fluss besteht aus einer Summe von Kreisen.

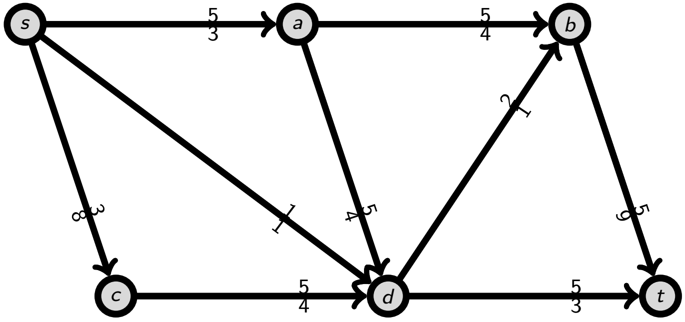
Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.
    - Für f' und f gilt die Flusserhaltung.
    - Und $f_{out}(s) = f'_{out}(s) = f_{in}(t) = f'_{in}(t)$.
    - Damit gilt: Falls $f(a,b) \neq f'(a,b)$, dann gibt es $c \in V \setminus \{a,b\}$ mit: $f(a,c) \neq f'(a,c)$.
    - Damit gibt es mindestens einen Kreis mit Fluss g über Kanten e mit $f(e) \neq f'(e)$.
  - Dieser zyklische Fluss besteht aus einer Summe von Kreisen.
  - Einer dieser Kreise muss die Kosten für f' verbessern.

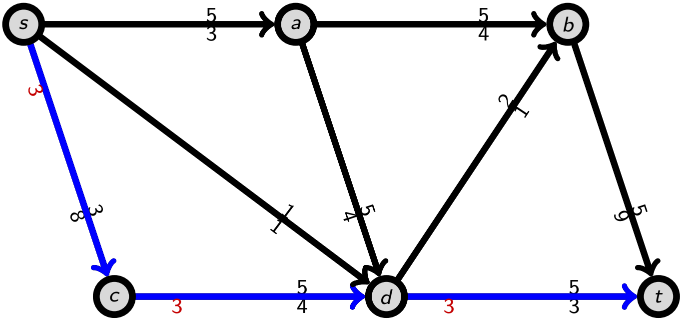
Idee

- Bestimme einen beliebigen Fluss f mit $w(f) = W$.
- Verbessere schrittweise die Kosten des Flusses:
  - Annahme: es gibt Fluss f' mit $l(f') < l(f)$ und $w(f') = w(f)$.
  - Dann gibt es einen Unterschied zwischen f und f'.
  - Betrachte diesen Unterschied.
  - Das muss ein zyklischer Fluss sein, d.h. ein Fluss ohne Quelle und Senke.
    - Für f' und f gilt die Flusserhaltung.
    - Und $f_{out}(s) = f'_{out}(s) = f_{in}(t) = f'_{in}(t)$.
    - Damit gilt: Falls $f(a,b) \neq f'(a,b)$, dann gibt es $c \in V \setminus \{a,b\}$ mit: $f(a,c) \neq f'(a,c)$.
    - Damit gibt es mindestens einen Kreis mit Fluss g über Kanten e mit $f(e) \neq f'(e)$.
  - Dieser zyklische Fluss besteht aus einer Summe von Kreisen.
  - Einer dieser Kreise muss die Kosten für f' verbessern.
- Idee: suche diese verbessernden Kreise.

Beispiel mit Kreisen ($W = 5$)

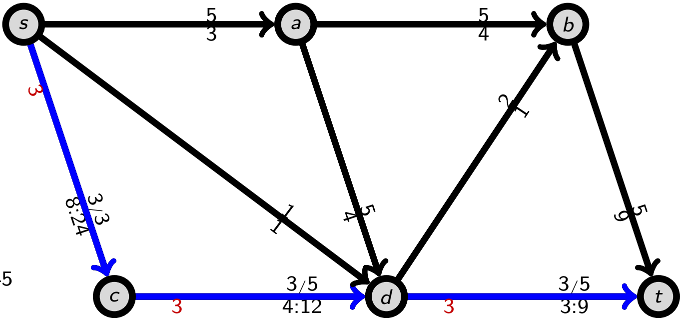


Beispiel mit Kreisen ($W = 5$)



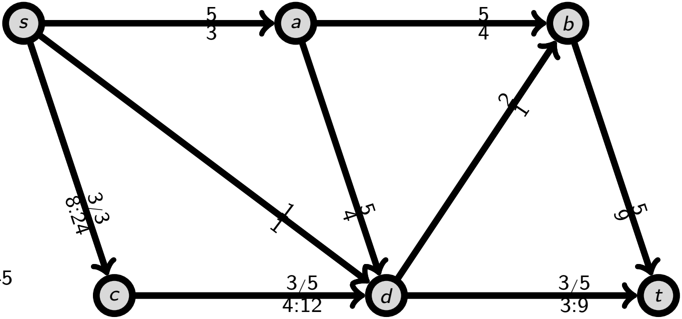
Beispiel mit Kreisen ($W = 5$)

$\Sigma = 45$

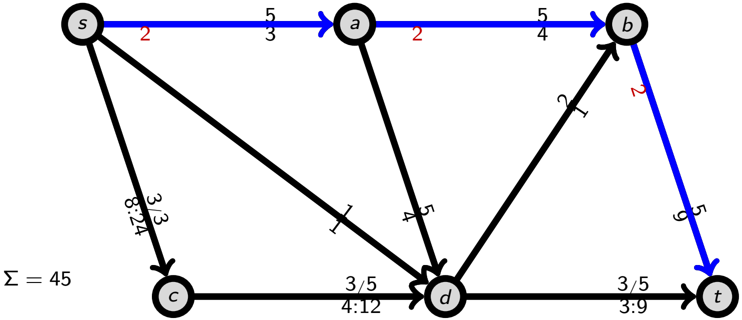


Beispiel mit Kreisen ($W = 5$)

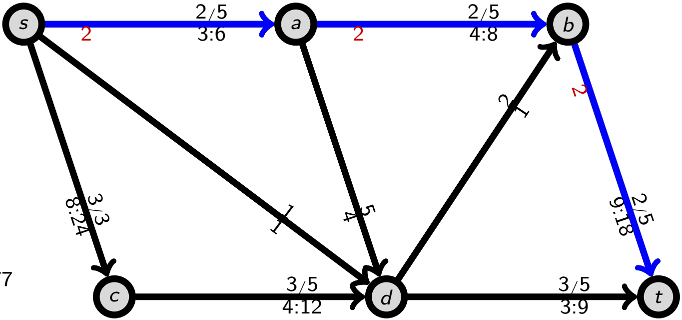
$\Sigma = 45$



Beispiel mit Kreisen ($W = 5$)

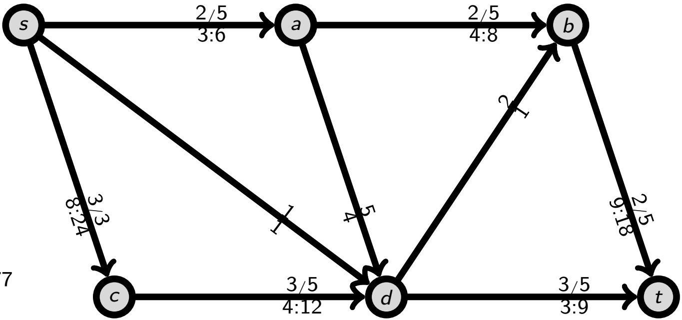


Beispiel mit Kreisen ($W = 5$)



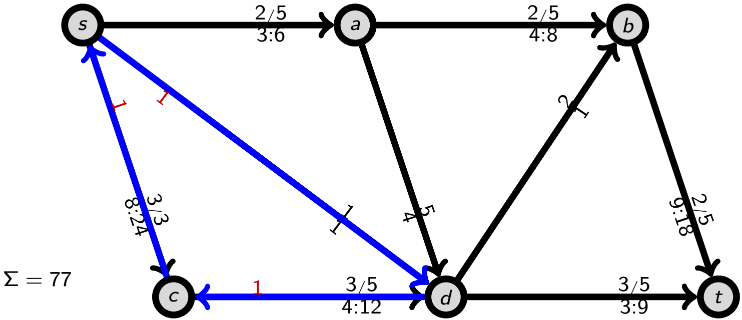
$\Sigma = 77$

Beispiel mit Kreisen ($W = 5$)

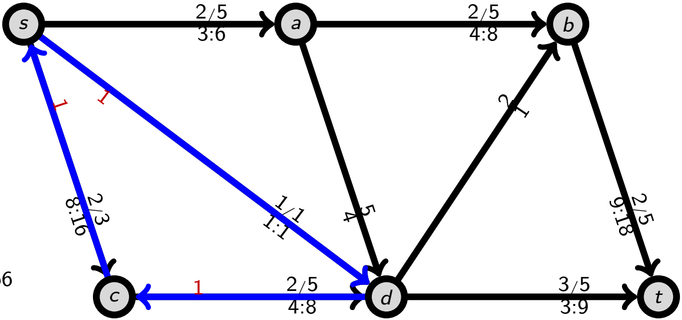


$\Sigma = 77$

Beispiel mit Kreisen ($W = 5$)

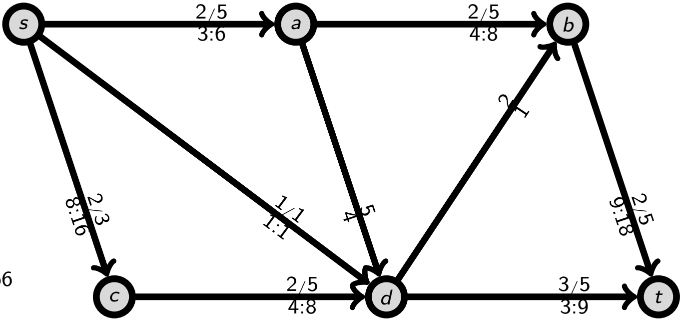


Beispiel mit Kreisen ($W = 5$)



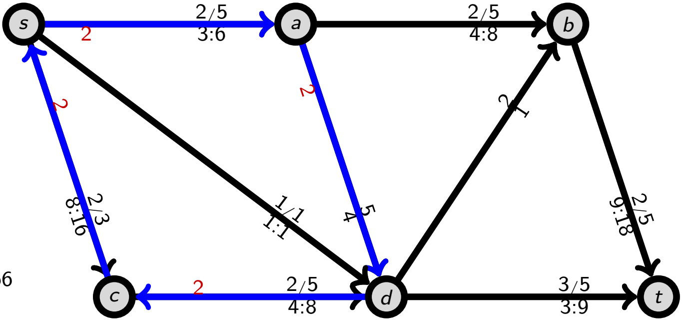
$\Sigma = 66$

Beispiel mit Kreisen ($W = 5$)



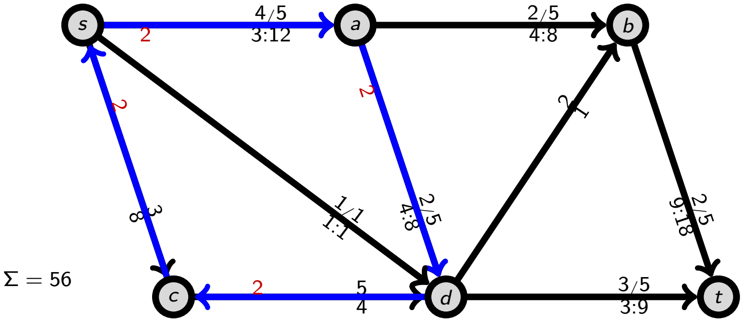
$\Sigma = 66$

Beispiel mit Kreisen ($W = 5$)

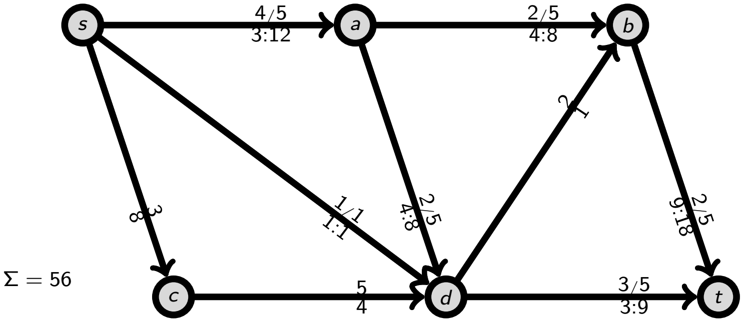


$\Sigma = 66$

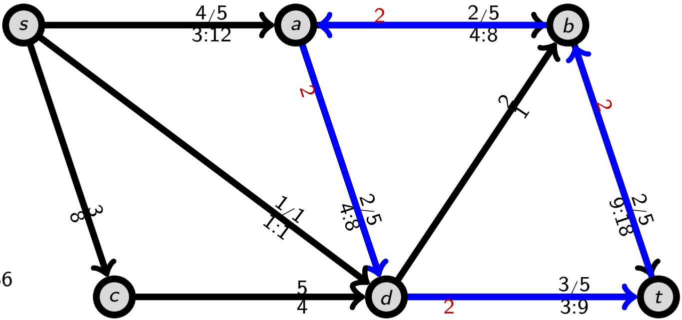
Beispiel mit Kreisen ($W = 5$)



Beispiel mit Kreisen ($W = 5$)

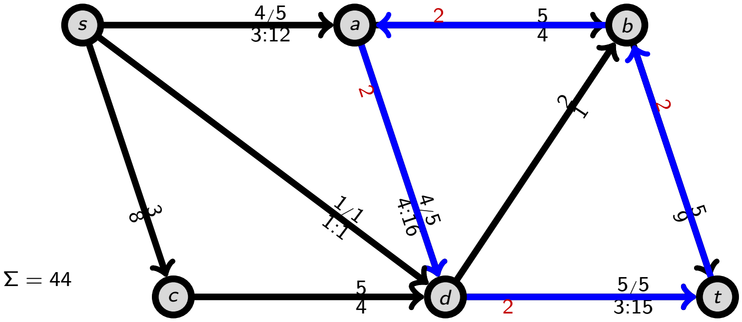


Beispiel mit Kreisen ($W = 5$)

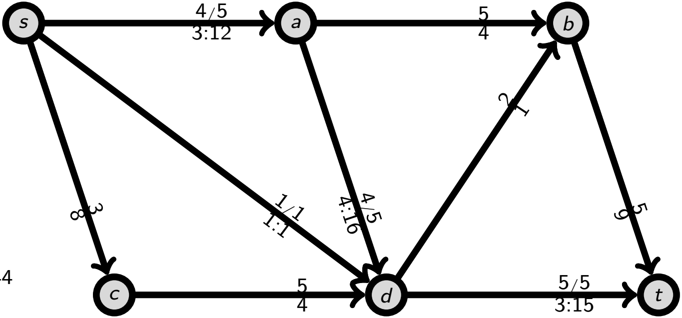


$\Sigma = 56$

Beispiel mit Kreisen ($W = 5$)



Beispiel mit Kreisen ($W = 5$)



$\Sigma = 44$

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.
- Vorteil bei Kreisen:

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.
- Vorteil bei Kreisen:
  - Gewinn entspricht direkt den Kosten des Kreises.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.
- Vorteil bei Kreisen:
  - Gewinn entspricht direkt den Kosten des Kreises.
  - Einfachere Algorithmen bei sich ändernden Kosten.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.
- Vorteil bei Kreisen:
  - Gewinn entspricht direkt den Kosten des Kreises.
  - Einfachere Algorithmen bei sich ändernden Kosten.
    - Optimale Lösung bekannt.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.
- Vorteil bei Kreisen:
  - Gewinn entspricht direkt den Kosten des Kreises.
  - Einfachere Algorithmen bei sich ändernden Kosten.
    - Optimale Lösung bekannt.
    - “Anbieter” verändert die Kosten einer Kante.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.
- Vorteil bei Kreisen:
  - Gewinn entspricht direkt den Kosten des Kreises.
  - Einfachere Algorithmen bei sich ändernden Kosten.
    - Optimale Lösung bekannt.
    - "Anbieter" verändert die Kosten einer Kante.
    - Passe bisherige Lösung durch das Suchen von Kreisen an.

Frage: Warum eine Suche nach Kreisen?

- Ein verbessernder Kreis entspricht zwei Wegen:
  - Ein Weg, der gelöscht wird,
  - Ein Weg, der hinzugefügt wird.
  - Die Differenz der Wege ist der Kreis.
- Man könnte also auch nach Wegen suchen.
- Vorteil bei Kreisen:
  - Gewinn entspricht direkt den Kosten des Kreises.
  - Einfachere Algorithmen bei sich ändernden Kosten.
    - Optimale Lösung bekannt.
    - "Anbieter" verändert die Kosten einer Kante.
    - Passe bisherige Lösung durch das Suchen von Kreisen an.
  - Die folgenden Beweise werden dadurch einfacher.

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a, b) = c(b, a)$ und $l'(a, b) = -l(b, a)$ für $(a, b) \in E' \setminus E$

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a, b) = c(b, a)$ und $l'(a, b) = -l(b, a)$ für $(a, b) \in E' \setminus E$
- f' ist eine Zirkulation, gdw.: $\forall v \in V : f'_{in}(v) = f'_{out}(v)$.

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a, b) = c(b, a)$ und $l'(a, b) = -l(b, a)$ für $(a, b) \in E' \setminus E$
- f' ist eine Zirkulation, gdw.: $\forall v \in V : f'_{in}(v) = f'_{out}(v)$.
- Wert einer Zirkulation f' über Schnitt (S, T):

$$f'(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} f'(v, w) - \sum_{(w,v) \in E, v \in S, w \in T} f'(w, v).$$

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a,b) = c(b,a)$ und $l'(a,b) = -l(b,a)$ für $(a,b) \in E' \setminus E$
- f' ist eine Zirkulation, gdw.: $\forall v \in V : f'_{in}(v) = f'_{out}(v)$.
- Wert einer Zirkulation f' über Schnitt (S,T):

$$f'(S,T) = \sum_{(v,w)\in E, v\in S, w\in T} f'(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f'(w,v).$$

- $w(f') = f'(\{s\}, V \setminus \{s\})$.

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a, b) = c(b, a)$ und $l'(a, b) = -l(b, a)$ für $(a, b) \in E' \setminus E$
- f' ist eine Zirkulation, gdw.: $\forall v \in V : f'_{in}(v) = f'_{out}(v)$.
- Wert einer Zirkulation f' über Schnitt (S, T):

$$f'(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f'(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f'(w, v).$$

- $w(f') = f'(\{s\}, V \setminus \{s\})$.

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a, b) = c(b, a)$ und $l'(a, b) = -l(b, a)$ für $(a, b) \in E' \setminus E$
- f' ist eine Zirkulation, gdw.: $\forall v \in V : f'_{in}(v) = f'_{out}(v)$.
- Wert einer Zirkulation f' über Schnitt (S, T):

$$f'(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f'(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f'(w, v).$$

- $w(f') = f'(\{s\}, V \setminus \{s\})$.

Lemma

Es gilt:

Beweis: Flusserhaltung und Induktion über Größe von S.

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a, b) = c(b, a)$ und $l'(a, b) = -l(b, a)$ für $(a, b) \in E' \setminus E$
- f' ist eine Zirkulation, gdw.: $\forall v \in V : f'_{in}(v) = f'_{out}(v)$.
- Wert einer Zirkulation f' über Schnitt (S, T):

$$f'(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f'(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f'(w, v).$$

- $w(f') = f'(\{s\}, V \setminus \{s\})$.

Lemma

Es gilt:

- $w(f') = 0$ *und*

Beweis: Flusserhaltung und Induktion über Größe von S.

Definitionen

- Gegeben $G = (V, E, s, t, c, l)$ und Restnetzwerk $G_f = (V', E', s, t, c', l')$.
- $c'(e) = c(e)$ für $e \in E \cap E'$
- $c'(a, b) = c(b, a)$ und $l'(a, b) = -l(b, a)$ für $(a, b) \in E' \setminus E$
- f' ist eine Zirkulation, gdw.: $\forall v \in V : f'_{in}(v) = f'_{out}(v)$.
- Wert einer Zirkulation f' über Schnitt (S, T):

$$f'(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f'(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f'(w, v).$$

- $w(f') = f'(\{s\}, V \setminus \{s\})$.

Lemma

Es gilt:

- $w(f') = 0$ *und*
- *für jeden Schnitt* (S, T) *gilt:* $f'(S, T) = 0$.

Beweis: Flusserhaltung und Induktion über Größe von S.

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

- Sei f^* Fluss auf G mit $l(f^*) < l(f)$.

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

- Sei f^* Fluss auf G mit $l(f^*) < l(f)$.
- Damit unterscheiden sich f^* und f.

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

- Sei f^* Fluss auf G mit $l(f^*) < l(f)$.
- Damit unterscheiden sich f^* und f.
- Für alle $e \in E$ setze $f'(e) = f^*(e) - f(e)$.

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

- Sei f^* Fluss auf G mit $l(f^*) < l(f)$.
- Damit unterscheiden sich f^* und f.
- Für alle $e \in E$ setze $f'(e) = f^*(e) - f(e)$.
- f' ist dann zyklischer Fluss.

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

- Sei f^* Fluss auf G mit $l(f^*) < l(f)$.
- Damit unterscheiden sich f^* und f.
- Für alle $e \in E$ setze $f'(e) = f^*(e) - f(e)$.
- f' ist dann zyklischer Fluss.
- f' wird durch höchstens $m' < m$ viele Kreisflüsse f_i' $(1 \leqslant i \leqslant m')$ gebildet.

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

- Sei f^* Fluss auf G mit $l(f^*) < l(f)$.
- Damit unterscheiden sich f^* und f.
- Für alle $e \in E$ setze $f'(e) = f^*(e) - f(e)$.
- f' ist dann zyklischer Fluss.
- f' wird durch höchstens $m' < m$ viele Kreisflüsse f_i' $(1 \leqslant i \leqslant m')$ gebildet.
- Damit gilt: $l(f') = l(f^*) - l(f) = \sum_{1 \leqslant i \leqslant m'} l(f_i')$

Zirkulation und Kostenminimalität

Lemma

Falls f nicht kostenminimal ist, dann gibt es einen Kreis C in G_f und f hat auf C negative Kosten.

Beweis:

- Sei f^* Fluss auf G mit $l(f^*) < l(f)$.
- Damit unterscheiden sich f^* und f.
- Für alle $e \in E$ setze $f'(e) = f^*(e) - f(e)$.
- f' ist dann zyklischer Fluss.
- f' wird durch höchstens $m' < m$ viele Kreisflüsse f_i' $(1 \leqslant i \leqslant m')$ gebildet.
- Damit gilt: $l(f') = l(f^*) - l(f) = \sum_{1 \leqslant i \leqslant m'} l(f_i')$
- Damit existiert j mit $l(f_j') < 0$.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
   1. $f = f + f'(C)$.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
    1. $f = f + f'(C)$.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
   1. $f = f + f'(C)$.

- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
   1. $f = f + f'(C)$.

- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.
  - D.h. der Wert des Flusses bleibt gleich.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
   1. $f = f + f'(C)$.

- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.
  - D.h. der Wert des Flusses bleibt gleich.
- Es gilt: $l(f + f'(C)) = l(f) + l(f'(C)) < l(f)$.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
   1. $f = f + f'(C)$.

- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.
  - D.h. der Wert des Flusses bleibt gleich.
- Es gilt: $l(f + f'(C)) = l(f) + l(f'(C)) < l(f)$.
  - D.h. die Kosten verringern sich.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
   1. $f = f + f'(C)$.

- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.
  - D.h. der Wert des Flusses bleibt gleich.
- Es gilt: $l(f + f'(C)) = l(f) + l(f'(C)) < l(f)$.
  - D.h. die Kosten verringern sich.

Algorithmus (Min-Cost-Flow)

Theorem

Ein Fluss f ist kostenminimal, wenn G_f keinen Kreis mit negativen Kosten enthält.

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f' mit $w(f') = W$.
3. Solange es in G_f einen negativen Kreis C gibt:
   1. $f = f + f'(C)$.

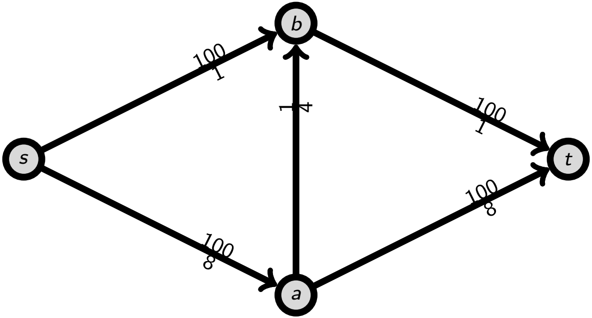
- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.
  - D.h. der Wert des Flusses bleibt gleich.
- Es gilt: $l(f + f'(C)) = l(f) + l(f'(C)) < l(f)$.
  - D.h. die Kosten verringern sich.

Theorem

Der obige Algorithmus bestimmt einen kostenminimalen Fluss.

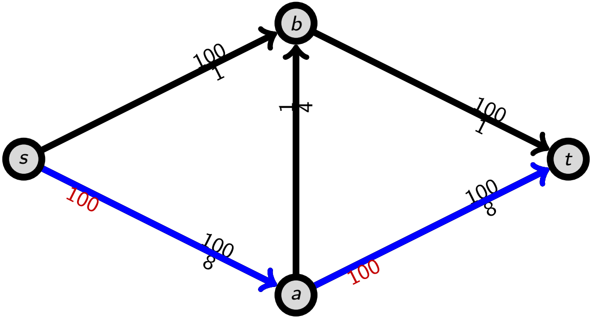
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



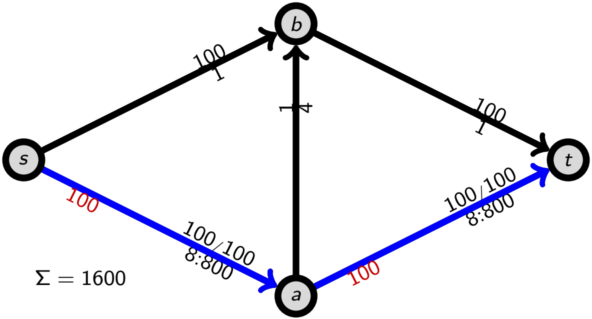
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



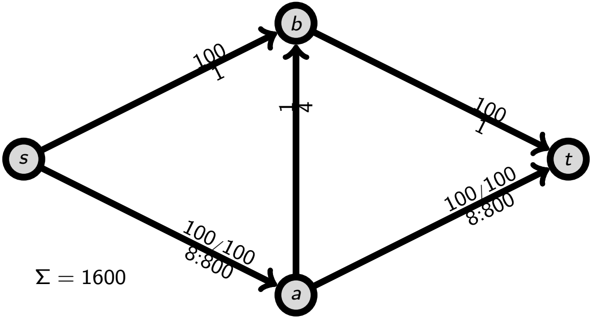
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



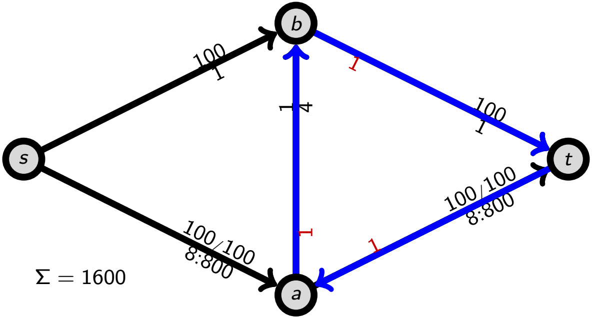
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



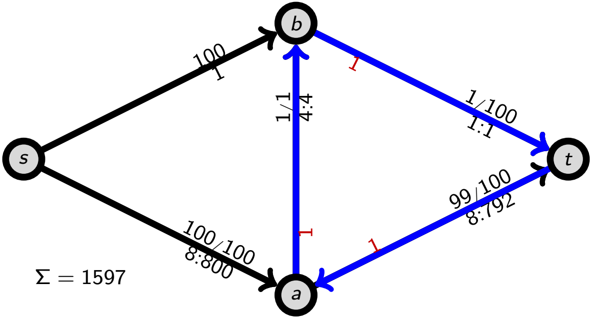
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

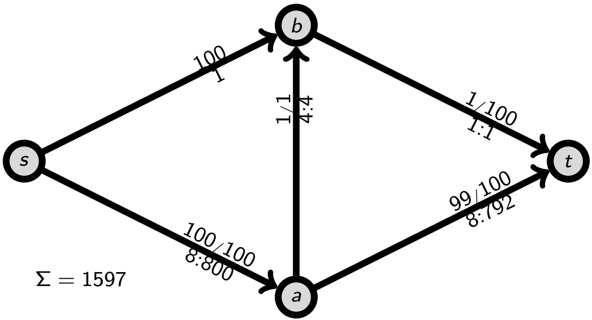
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1597$

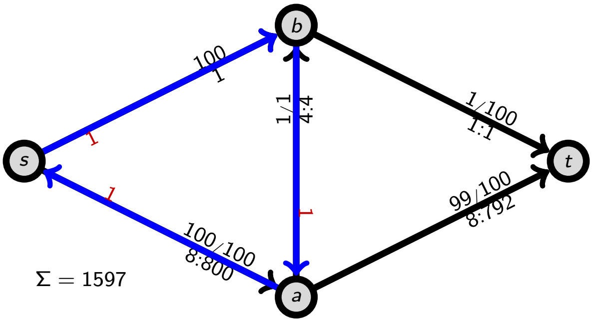
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



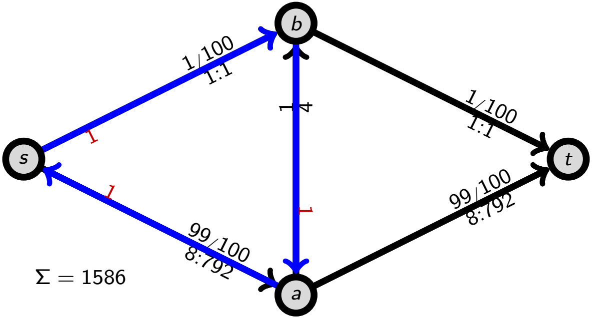
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

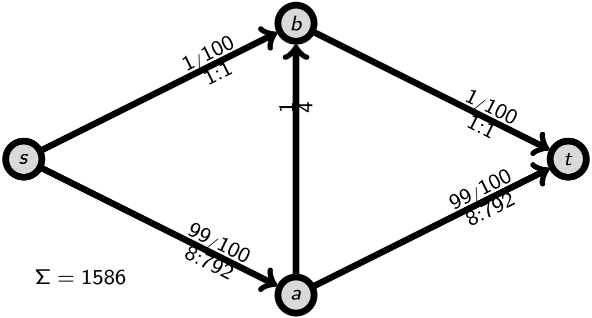
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1586$

Beispiel zur Laufzeit ($W = 100$)

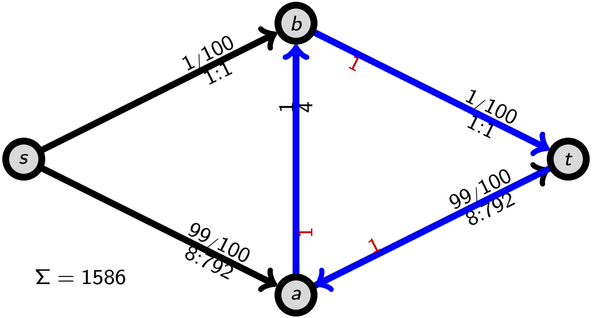
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1586$

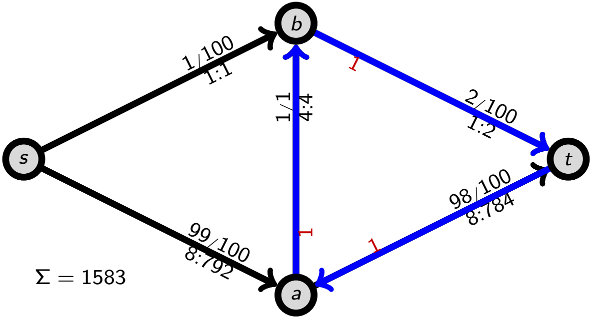
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



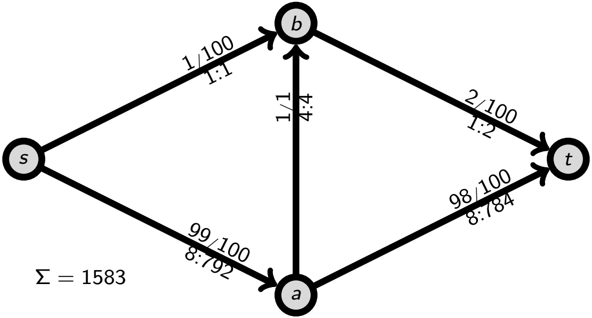
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

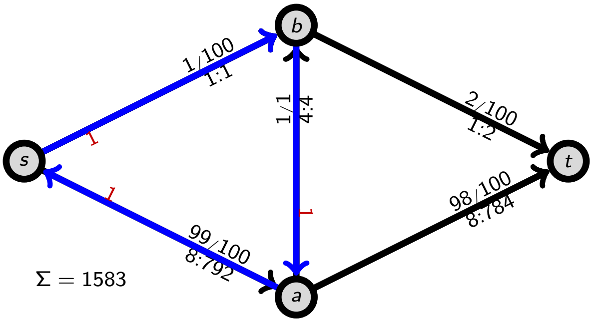
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1583$

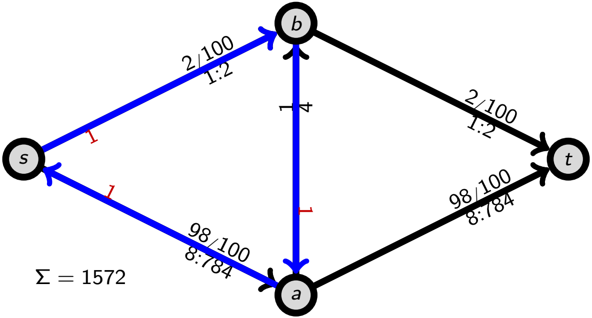
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

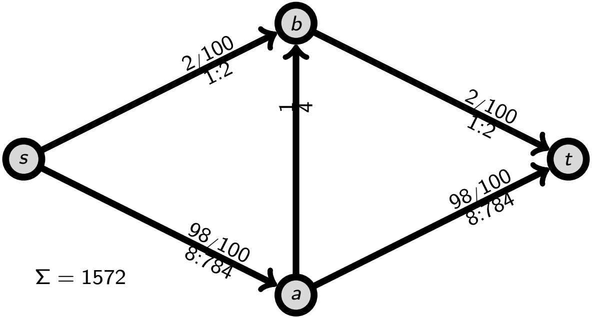
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1572$

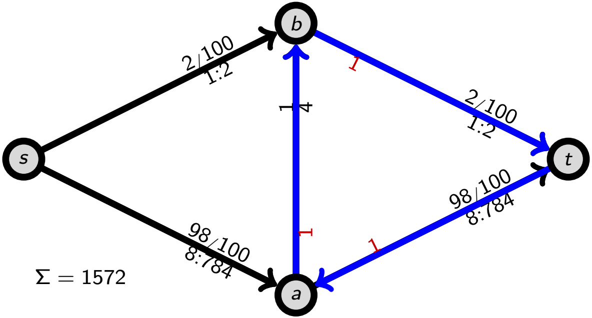
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

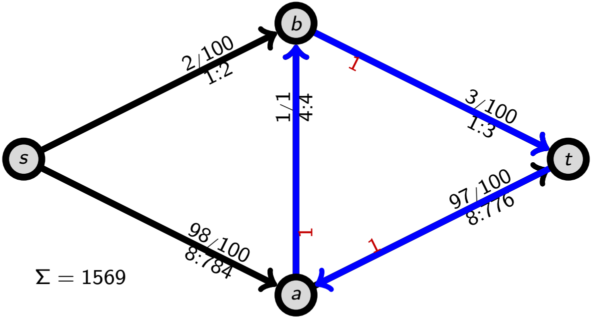
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1572$

Beispiel zur Laufzeit ($W = 100$)

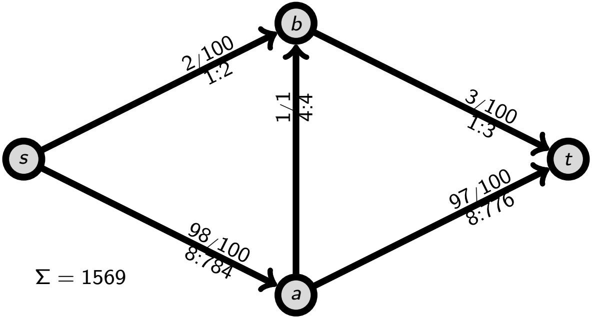
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1569$

Beispiel zur Laufzeit ($W = 100$)

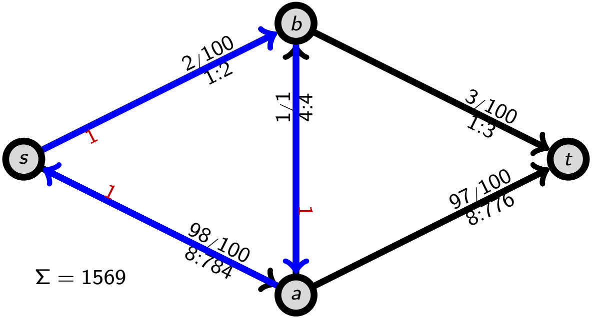
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1569$

Beispiel zur Laufzeit ($W = 100$)

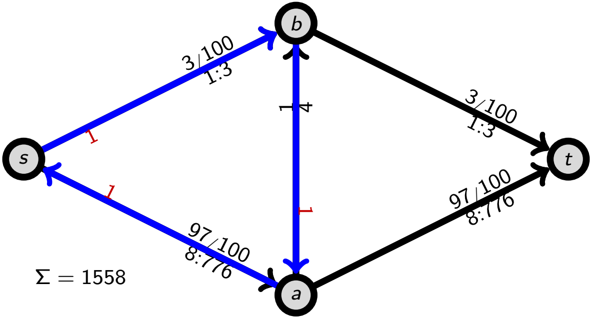
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1569$

Beispiel zur Laufzeit ($W = 100$)

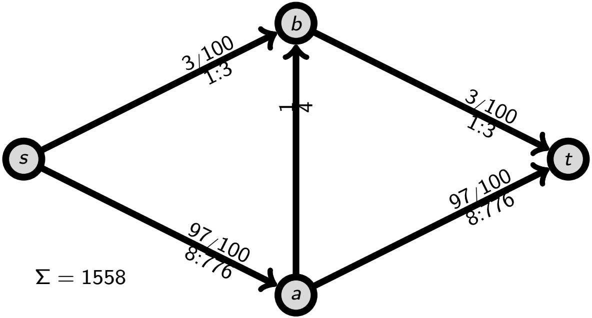
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1558$

Beispiel zur Laufzeit ($W = 100$)

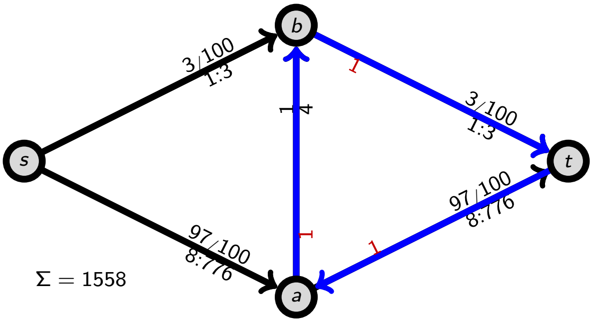
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1558$

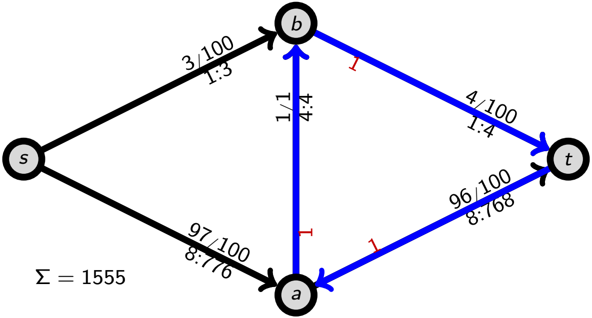
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



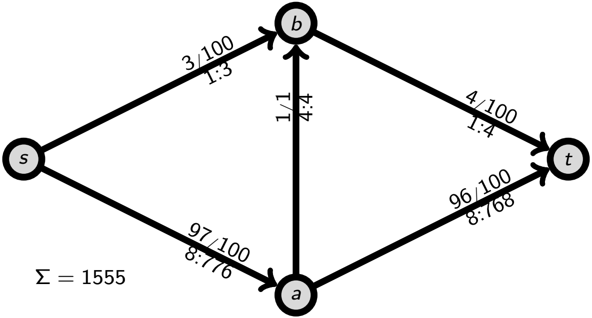
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

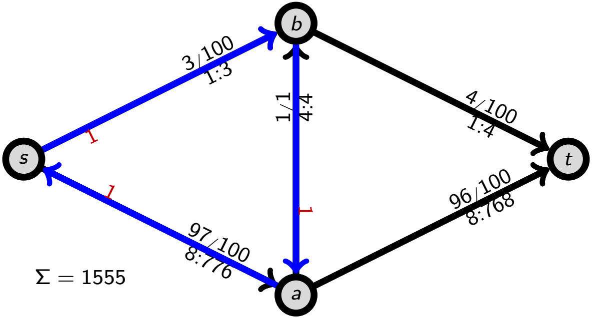
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1555$

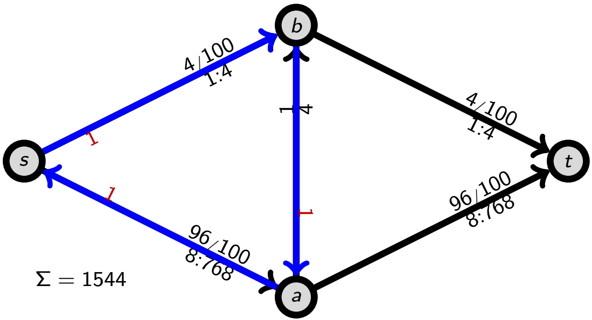
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



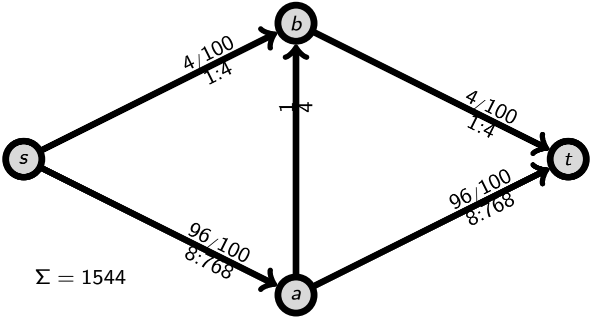
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

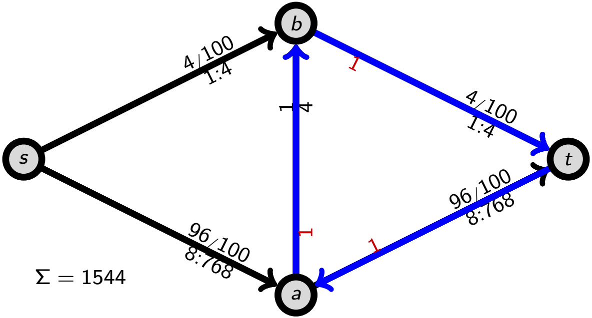
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1544$

Beispiel zur Laufzeit ($W = 100$)

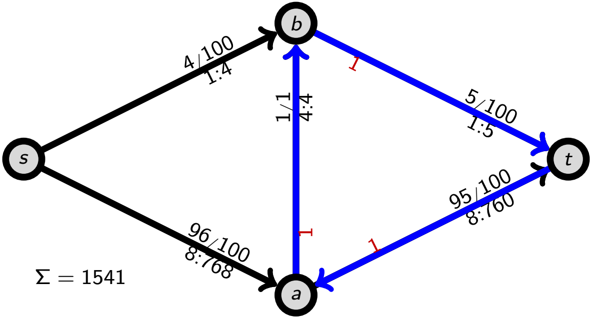
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1544$

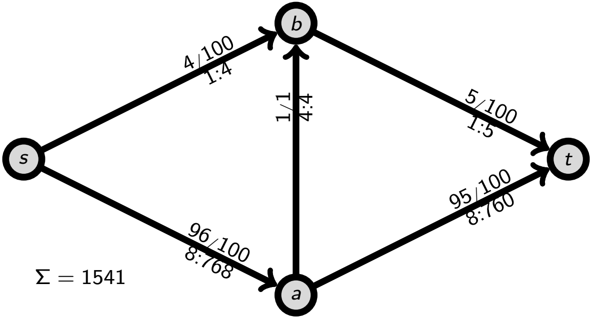
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

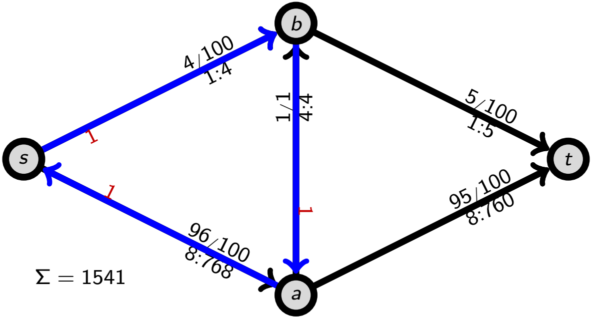
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1541$

Beispiel zur Laufzeit ($W = 100$)

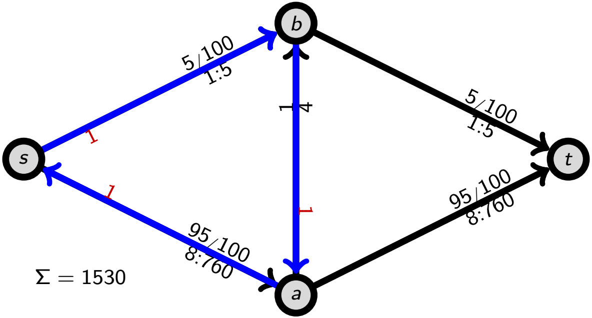
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1541$

Beispiel zur Laufzeit ($W = 100$)

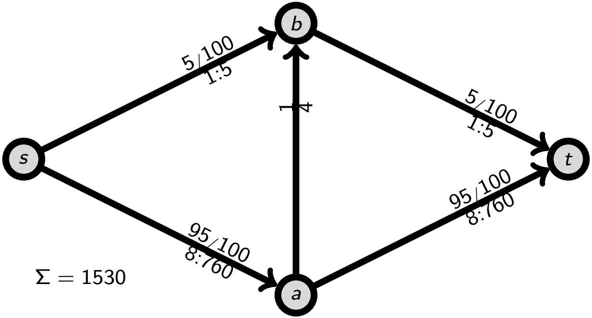
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1530$

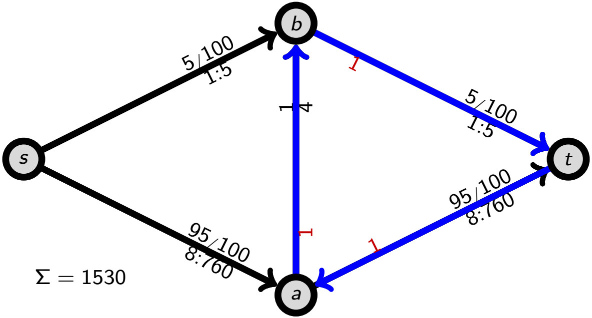
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



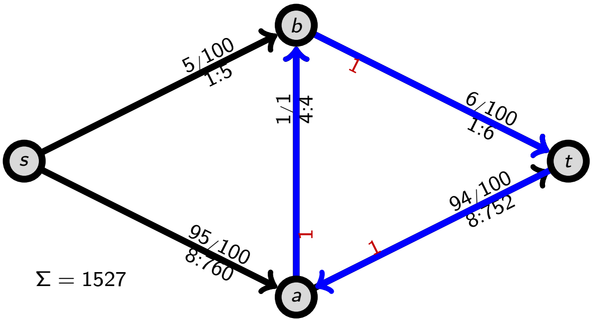
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



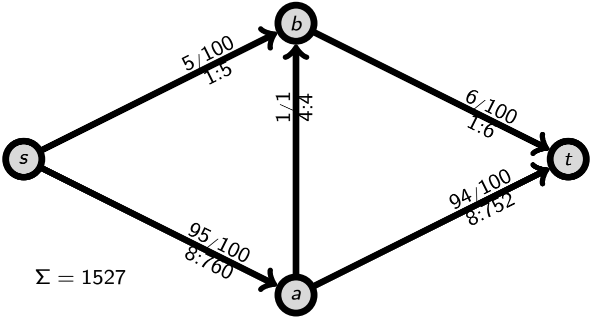
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



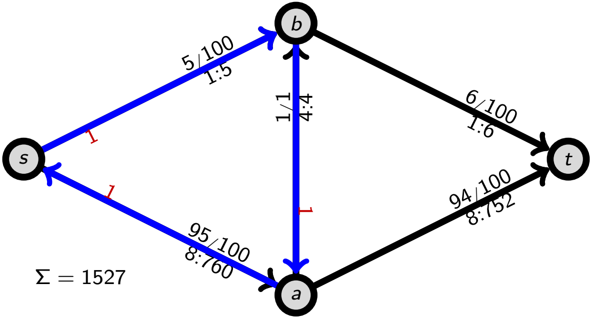
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



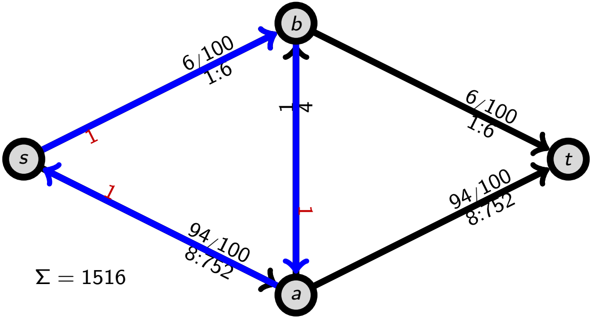
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

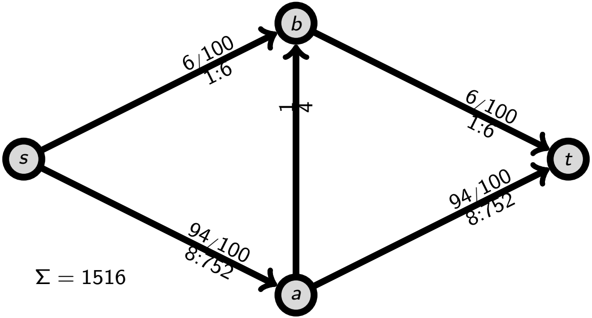
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1516$

Beispiel zur Laufzeit ($W = 100$)

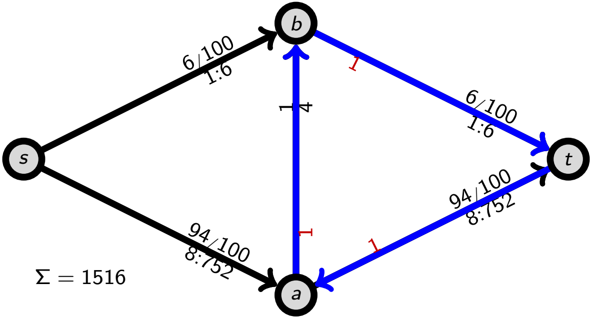
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1516$

Beispiel zur Laufzeit ($W = 100$)

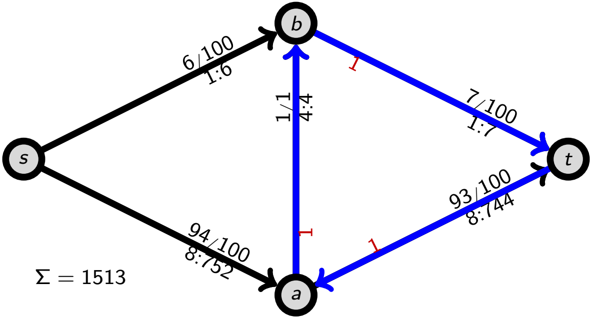
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1516$

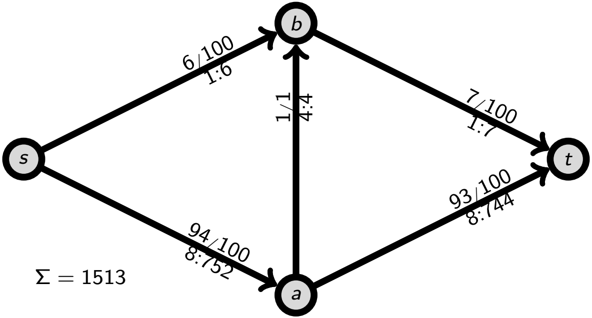
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

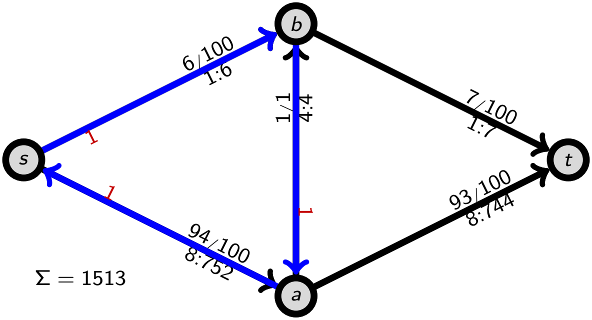
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1513$

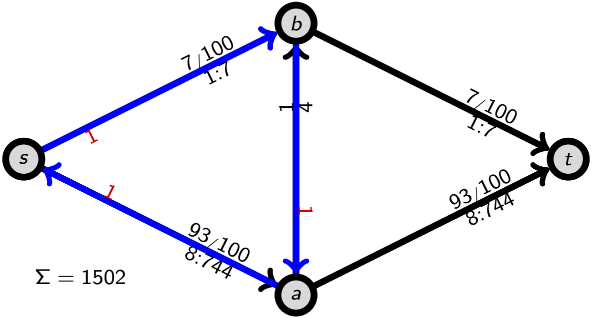
Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



Beispiel zur Laufzeit ($W = 100$)

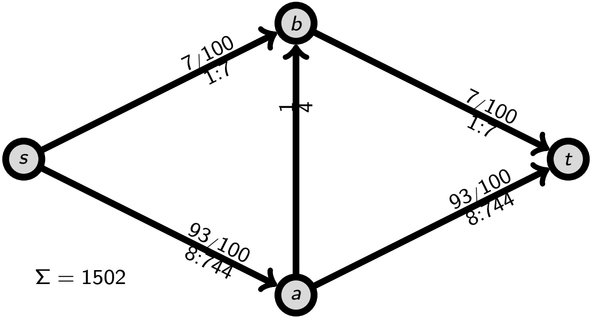
Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.



$\Sigma = 1502$

Beispiel zur Laufzeit ($W = 100$)

Bei schlecht gewählten Kreisen kann die Laufzeit sehr lang werden.

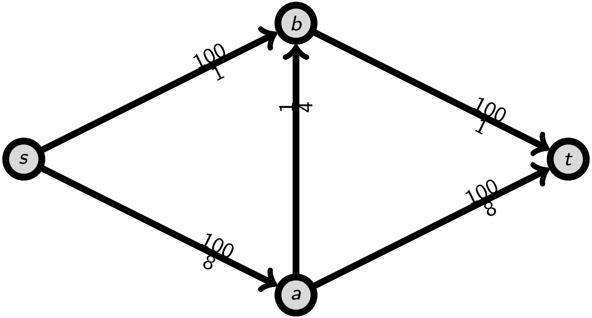


$\Sigma = 1502$

Hier geben wir auf.

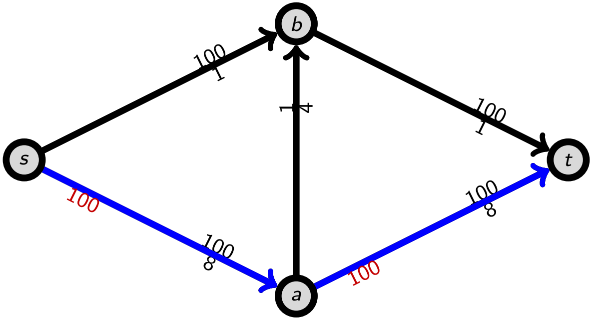
Nochmal Beispiel zur Laufzeit ($W = 100$)

Bei gut gewählten Kreisen kann die Laufzeit ggf. besser sein.



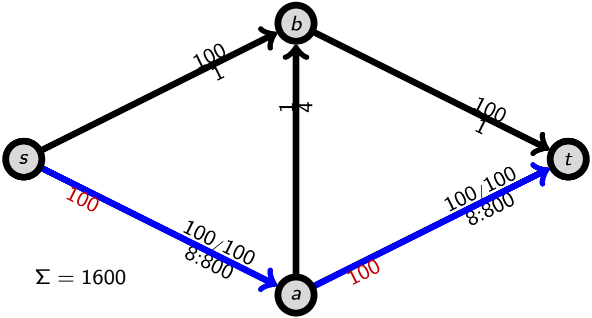
Nochmal Beispiel zur Laufzeit ($W = 100$)

Bei gut gewählten Kreisen kann die Laufzeit ggf. besser sein.



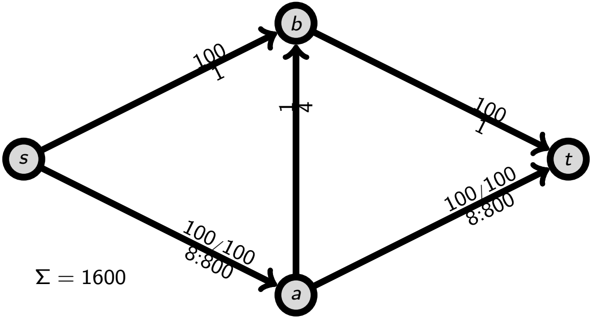
Nochmal Beispiel zur Laufzeit ($W = 100$)

Bei gut gewählten Kreisen kann die Laufzeit ggf. besser sein.



Nochmal Beispiel zur Laufzeit ($W = 100$)

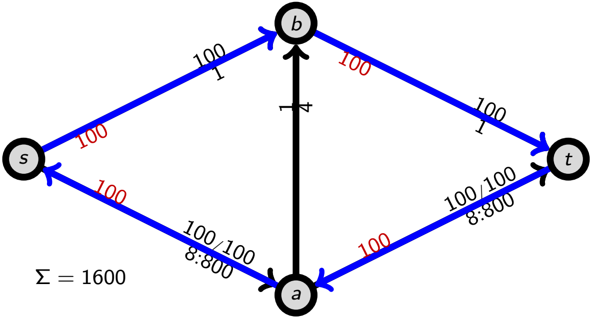
Bei gut gewählten Kreisen kann die Laufzeit ggf. besser sein.



$\Sigma = 1600$

Nochmal Beispiel zur Laufzeit ($W = 100$)

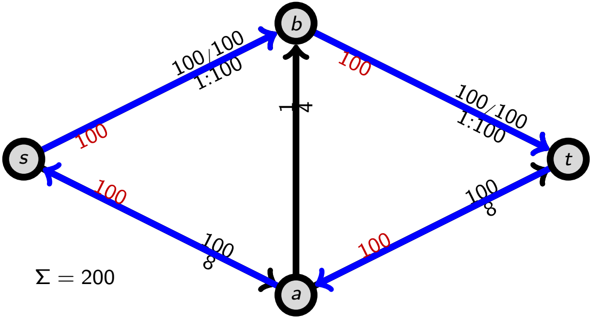
Bei gut gewählten Kreisen kann die Laufzeit ggf. besser sein.



$\Sigma = 1600$

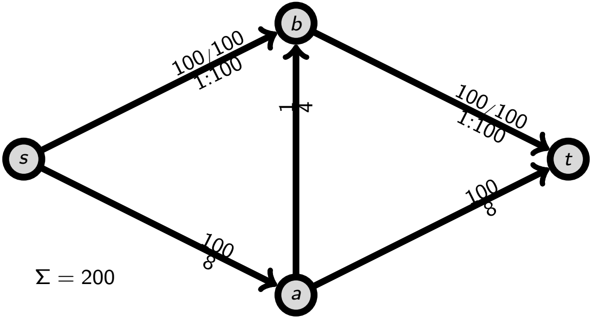
Nochmal Beispiel zur Laufzeit ($W = 100$)

Bei gut gewählten Kreisen kann die Laufzeit ggf. besser sein.



Nochmal Beispiel zur Laufzeit ($W = 100$)

Bei gut gewählten Kreisen kann die Laufzeit ggf. besser sein.



Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.
  - Wähle Kreise über kostengünstige Kanten.

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.
  - Wähle Kreise über kostengünstige Kanten.
  - Also unabhängig von der Kreislänge.

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.
  - Wähle Kreise über kostengünstige Kanten.
  - Also unabhängig von der Kreislänge.
- Dazu werden die Kreise gewählt, die die durchschnittlichen Kantenkosten minimieren.

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.
  - Wähle Kreise über kostengünstige Kanten.
  - Also unabhängig von der Kreislänge.
- Dazu werden die Kreise gewählt, die die durchschnittlichen Kantenkosten minimieren.
- Setze:

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.
  - Wähle Kreise über kostengünstige Kanten.
  - Also unabhängig von der Kreislänge.
- Dazu werden die Kreise gewählt, die die durchschnittlichen Kantenkosten minimieren.
- Setze:

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

- Setze weiter: $\mu(f) = -\bar{l}(C)$.

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.
  - Wähle Kreise über kostengünstige Kanten.
  - Also unabhängig von der Kreislänge.
- Dazu werden die Kreise gewählt, die die durchschnittlichen Kantenkosten minimieren.
- Setze:

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

- Setze weiter: $\mu(f) = -\bar{l}(C)$.
- Falls C negative Kosten hat, so ist $\mu(C)$ positiv.

Laufzeit und Verbesserung selbiger

- Der bisherige Algorithmus hat in obiger Form eine pseudopolynomielle Laufzeit.
- Dies kann aber verbessert werden:
  - Wähle kostengünstige Kreise.
  - Wähle Kreise über kostengünstige Kanten.
  - Also unabhängig von der Kreislänge.
- Dazu werden die Kreise gewählt, die die durchschnittlichen Kantenkosten minimieren.
- Setze:

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

- Setze weiter: $\mu(f) = -\bar{l}(C)$.
- Falls C negative Kosten hat, so ist $\mu(C)$ positiv.
- $\mu(f)$ gibt also die Verbesserung eines Kreisflusses an.

Mean-Algorithmus (Min-Cost-Flow)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

1. Eingabe: $G = (V, E, s, t, c, l), W$.

Mean-Algorithmus (Min-Cost-Flow)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f mit $w(f) = W$.

Mean-Algorithmus (Min-Cost-Flow)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f mit $w(f) = W$.
3. Solange es in G_f einen negativen Kreis C gibt,

Mean-Algorithmus (Min-Cost-Flow)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e\in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f mit $w(f) = W$.
3. Solange es in G_f einen negativen Kreis C gibt,
    1. Wähle C mit $\bar{l}(C)$ minimal.

Mean-Algorithmus (Min-Cost-Flow)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f mit $w(f) = W$.
3. Solange es in G_f einen negativen Kreis C gibt,
   1. Wähle C mit $\bar{l}(C)$ minimal.
      1. Bestimme maximalen zyklischen Fluss f' auf C.

Mean-Algorithmus (Min-Cost-Flow)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f mit $w(f) = W$.
3. Solange es in G_f einen negativen Kreis C gibt,
    1. Wähle C mit $\bar{l}(C)$ minimal.
        1. Bestimme maximalen zyklischen Fluss f' auf C.
        2. $f = f + f'(C)$.

Mean-Algorithmus (Min-Cost-Flow)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f mit $w(f) = W$.
3. Solange es in G_f einen negativen Kreis C gibt,
   1. Wähle C mit $\bar{l}(C)$ minimal.
      1. Bestimme maximalen zyklischen Fluss f' auf C.
      2. $f = f + f'(C)$.

- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.

Mean-Algorithmus (Min-Cost-Flow)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$

1. Eingabe: $G = (V, E, s, t, c, l), W$.
2. Bestimme Fluss f mit $w(f) = W$.
3. Solange es in G_f einen negativen Kreis C gibt,
   1. Wähle C mit $\bar{l}(C)$ minimal.
      1. Bestimme maximalen zyklischen Fluss f' auf C.
      2. $f = f + f'(C)$.

- Es gilt: $w(f + f'(C)) = w(f) + w(f'(C)) = w(f) = W$.
- Es gilt: $l(f + f'(C)) = l(f) + l(f'(C)) < l(f)$.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x},\ x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.
  - Verändere l so, dass Annahme immer gilt und Algorithmus analog vorgeht.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.
  - Verändere l so, dass Annahme immer gilt und Algorithmus analog vorgeht.
  - Bestimme dazu Potential der Knoten, und addiere dies zu den Kantenkosten.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.
  - Verändere l so, dass Annahme immer gilt und Algorithmus analog vorgeht.
  - Bestimme dazu Potential der Knoten, und addiere dies zu den Kantenkosten.
- Bestimme Laufzeit für eine Iteration.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.
  - Verändere l so, dass Annahme immer gilt und Algorithmus analog vorgeht.
  - Bestimme dazu Potential der Knoten, und addiere dies zu den Kantenkosten.
- Bestimme Laufzeit für eine Iteration.
- Zeige, wie man einfach einen Min-Mean-Kreis C findet (Dynamisches Programmieren).

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.
  - Verändere l so, dass Annahme immer gilt und Algorithmus analog vorgeht.
  - Bestimme dazu Potential der Knoten, und addiere dies zu den Kantenkosten.
- Bestimme Laufzeit für eine Iteration.
- Zeige, wie man einfach einen Min-Mean-Kreis C findet (Dynamisches Programmieren).
  - Bestimme dazu vorab den Wert von C.

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.
  - Verändere l so, dass Annahme immer gilt und Algorithmus analog vorgeht.
  - Bestimme dazu Potential der Knoten, und addiere dies zu den Kantenkosten.
- Bestimme Laufzeit für eine Iteration.
- Zeige, wie man einfach einen Min-Mean-Kreis C findet (Dynamisches Programmieren).
  - Bestimme dazu vorab den Wert von C.
  - Dann wird Suche nach C einfach (passe Kosten an).

Überblick zum Beweis

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Teile Iterationen in Phasen auf.
- Bestimme die Anzahl der Phasen.
- Bestimme die Anzahl der Iterationen pro Phase.
  - Zeige dies unter Verwendung einer besonderen Annahme.
  - Verändere l so, dass Annahme immer gilt und Algorithmus analog vorgeht.
  - Bestimme dazu Potential der Knoten, und addiere dies zu den Kantenkosten.
- Bestimme Laufzeit für eine Iteration.
- Zeige, wie man einfach einen Min-Mean-Kreis C findet (Dynamisches Programmieren).
  - Bestimme dazu vorab den Wert von C.
  - Dann wird Suche nach C einfach (passe Kosten an).
- Beachte für verbessernde Kreise C gilt: $\bar{l}(C) \leqslant -1/n$.

Laufzeit (Aufteilung in Phasen)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$1 - x \leqslant e^{-x}, x = 1/n$$

- Sei f der Fluss zu Beginn der i-ten Phase.

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.
- Sei μ_0 der Wert von μ zu Beginn der ersten Phase.

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.
- Sei μ_0 der Wert von μ zu Beginn der ersten Phase.
- Sei μ_i der Wert von μ am Ende der i-ten Phase $(1 \leqslant i \leqslant T)$.

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.
- Sei μ_0 der Wert von μ zu Beginn der ersten Phase.
- Sei μ_i der Wert von μ am Ende der i-ten Phase $(1 \leqslant i \leqslant T)$.
- Damit gilt:

$$\mu_i \leqslant (1 - 1/n) \cdot \mu_{i-1} \leqslant \frac{\mu_{i-1}}{e^{1/n}}.$$

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.
- Sei μ_0 der Wert von μ zu Beginn der ersten Phase.
- Sei μ_i der Wert von μ am Ende der i-ten Phase $(1 \leqslant i \leqslant T)$.
- Damit gilt:

$$\mu_i \leqslant (1 - 1/n) \cdot \mu_{i-1} \leqslant \frac{\mu_{i-1}}{e^{1/n}}.$$

- Weiter gilt: $\mu_0 \leqslant L = \sum_{e \in E} |l(e)|$

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.
- Sei μ_0 der Wert von μ zu Beginn der ersten Phase.
- Sei μ_i der Wert von μ am Ende der i-ten Phase $(1 \leqslant i \leqslant T)$.
- Damit gilt:

$$\mu_i \leqslant (1 - 1/n) \cdot \mu_{i-1} \leqslant \frac{\mu_{i-1}}{e^{1/n}}.$$

- Weiter gilt: $\mu_0 \leqslant L = \sum_{e \in E} |l(e)|$
- Wegen der Ganzzahligkeit gilt: $\mu_{T-1} \geqslant 1/n$.

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.
- Sei μ_0 der Wert von μ zu Beginn der ersten Phase.
- Sei μ_i der Wert von μ am Ende der i-ten Phase $(1 \leqslant i \leqslant T)$.
- Damit gilt:

$$\mu_i \leqslant (1 - 1/n) \cdot \mu_{i-1} \leqslant \frac{\mu_{i-1}}{e^{1/n}}.$$

- Weiter gilt: $\mu_0 \leqslant L = \sum_{e \in E} |l(e)|$
- Wegen der Ganzzahligkeit gilt: $\mu_{T-1} \geqslant 1/n$.
- Es folgt:

$$T - 1 \leqslant \log_{e^{1/n}}(nL) = \frac{\ln(nL)}{\ln(e^{1/n})} = n \ln(nL)$$

Laufzeit (Aufteilung in Phasen)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$1 - x \leqslant e^{-x}, x = 1/n$

- Sei f der Fluss zu Beginn der i-ten Phase.
- Die Phase i endet, falls ein Fluss g gefunden wird mit:

$$\mu(g) \leqslant (1 - 1/n) \cdot \mu(f) \text{ oder } \mu(g) \leqslant 0.$$

- Falls $\mu(g) \leqslant 0$ terminiert der Algorithmus.
- Sei μ_0 der Wert von μ zu Beginn der ersten Phase.
- Sei μ_i der Wert von μ am Ende der i-ten Phase $(1 \leqslant i \leqslant T)$.
- Damit gilt:

$$\mu_i \leqslant (1 - 1/n) \cdot \mu_{i-1} \leqslant \frac{\mu_{i-1}}{e^{1/n}}.$$

- Weiter gilt: $\mu_0 \leqslant L = \sum_{e \in E} |l(e)|$
- Wegen der Ganzzahligkeit gilt: $\mu_{T-1} \geqslant 1/n$.
- Es folgt:

$$T - 1 \leqslant \log_{e^{1/n}}(nL) = \frac{\ln(nL)}{\ln(e^{1/n})} = n \ln(nL)$$

- Und: $T \leqslant n \ln(nL) + 1$.

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

- Zeige im Folgenden:

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder
  - Die nächste Phase startet nach spätestens $m-1$ Iterationen.

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder
  - Die nächste Phase startet nach spätestens $m-1$ Iterationen.
    - D.h. war der initiale Fluss f zu Beginn der Phase,

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder
  - Die nächste Phase startet nach spätestens $m-1$ Iterationen.
    - D.h. war der initiale Fluss f zu Beginn der Phase,
    - dann ist ein Fluss g nach spätestens $m-1$ Iterationen erreicht mit:

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder
  - Die nächste Phase startet nach spätestens $m-1$ Iterationen.
    - D.h. war der initiale Fluss f zu Beginn der Phase,
    - dann ist ein Fluss g nach spätestens $m-1$ Iterationen erreicht mit:
    - $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder
  - Die nächste Phase startet nach spätestens $m-1$ Iterationen.
    - D.h. war der initiale Fluss f zu Beginn der Phase,
    - dann ist ein Fluss g nach spätestens $m-1$ Iterationen erreicht mit:
    - $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.
- Vorgehen:

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder
  - Die nächste Phase startet nach spätestens $m-1$ Iterationen.
    - D.h. war der initiale Fluss f zu Beginn der Phase,
    - dann ist ein Fluss g nach spätestens $m-1$ Iterationen erreicht mit:
    - $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.
- Vorgehen:
  - Zeige Behauptung unter der Annahme: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$.

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

- Zeige im Folgenden:
  - Die Phase (und der Algorithmus) terminiert nach spätestens m Iterationen, oder
  - Die nächste Phase startet nach spätestens $m - 1$ Iterationen.
    - D.h. war der initiale Fluss f zu Beginn der Phase,
    - dann ist ein Fluss g nach spätestens $m - 1$ Iterationen erreicht mit:
    - $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.
- Vorgehen:
  - Zeige Behauptung unter der Annahme: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$.
  - Zeige Behauptung: Verändere dann l, so dass die Annahme immer gilt.

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$$

- Typ 1 Iteration: Kreis C enthält nur Kanten mit negativen Kosten.

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Typ 1 Iteration: Kreis C enthält nur Kanten mit negativen Kosten.
- Typ 2 Iteration: Kreis C enthält mindestens eine Kante mit positiven Kosten.

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$$

- Typ 1 Iteration: Kreis C enthält nur Kanten mit negativen Kosten.
- Typ 2 Iteration: Kreis C enthält mindestens eine Kante mit positiven Kosten.
- Bei jeder Typ 1 Iteration wird mindestens eine Kante saturiert und entfernt.

Laufzeit (Iterationen pro Phase)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$$

- Typ 1 Iteration: Kreis C enthält nur Kanten mit negativen Kosten.
- Typ 2 Iteration: Kreis C enthält mindestens eine Kante mit positiven Kosten.
- Bei jeder Typ 1 Iteration wird mindestens eine Kante saturiert und entfernt.
- Alle dabei neu entstehenden Kanten haben positive Kosten (andere Richtung).

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Typ 1 Iteration: Kreis C enthält nur Kanten mit negativen Kosten.
- Typ 2 Iteration: Kreis C enthält mindestens eine Kante mit positiven Kosten.
- Bei jeder Typ 1 Iteration wird mindestens eine Kante saturiert und entfernt.
- Alle dabei neu entstehenden Kanten haben positive Kosten (andere Richtung).
- Nach spätestens m konsekutiven Typ 1 Iterationen terminiert das Verfahren.

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Typ 1 Iteration: Kreis C enthält nur Kanten mit negativen Kosten.
- Typ 2 Iteration: Kreis C enthält mindestens eine Kante mit positiven Kosten.
- Bei jeder Typ 1 Iteration wird mindestens eine Kante saturiert und entfernt.
- Alle dabei neu entstehenden Kanten haben positive Kosten (andere Richtung).
- Nach spätestens m konsekutiven Typ 1 Iterationen terminiert das Verfahren.
- Wenn also die Phase mehr als m Iterationen hat, folgt nach spätestens $m-1$ Iterationen eine Typ 2 Iteration.

Laufzeit (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Typ 1 Iteration: Kreis C enthält nur Kanten mit negativen Kosten.
- Typ 2 Iteration: Kreis C enthält mindestens eine Kante mit positiven Kosten.
- Bei jeder Typ 1 Iteration wird mindestens eine Kante saturiert und entfernt.
- Alle dabei neu entstehenden Kanten haben positive Kosten (andere Richtung).
- Nach spätestens m konsekutiven Typ 1 Iterationen terminiert das Verfahren.
- Wenn also die Phase mehr als m Iterationen hat, folgt nach spätestens $m - 1$ Iterationen eine Typ 2 Iteration.
- Wir zeigen nun, dass dieser Fall unter der Annahme nicht auftritt.

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

- Sei C der Min-Mean-Kreis in G_g und H die Kanten mit negativen Kosten in C.

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

- Sei C der Min-Mean-Kreis in G_g und H die Kanten mit negativen Kosten in C.
- Damit gilt:

$$\mu(g) = \sum_{e \in C} \frac{-l(e)}{|C|} \leqslant \sum_{e \in H} \frac{-l(e)}{|C|} \leqslant |H| \cdot \frac{\mu(f)}{|C|}$$

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e\in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

- Sei C der Min-Mean-Kreis in G_g und H die Kanten mit negativen Kosten in C.
- Damit gilt:

$$\mu(g) = \sum_{e\in C} \frac{-l(e)}{|C|} \leqslant \sum_{e\in H} \frac{-l(e)}{|C|} \leqslant |H| \cdot \frac{\mu(f)}{|C|}$$

- Wegen $|H| \leqslant |C| - 1$ folgt:

$$|H|/|C| \leqslant 1 - 1/|C| \leqslant 1 - 1/n$$

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

- Sei C der Min-Mean-Kreis in G_g und H die Kanten mit negativen Kosten in C.
- Damit gilt:

$$\mu(g) = \sum_{e \in C} \frac{-l(e)}{|C|} \leqslant \sum_{e \in H} \frac{-l(e)}{|C|} \leqslant |H| \cdot \frac{\mu(f)}{|C|}$$

- Wegen $|H| \leqslant |C| - 1$ folgt:

$$|H|/|C| \leqslant 1 - 1/|C| \leqslant 1 - 1/n$$

- Damit: $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

- Sei C der Min-Mean-Kreis in G_g und H die Kanten mit negativen Kosten in C.
- Damit gilt:

$$\mu(g) = \sum_{e \in C} \frac{-l(e)}{|C|} \leqslant \sum_{e \in H} \frac{-l(e)}{|C|} \leqslant |H| \cdot \frac{\mu(f)}{|C|}$$

- Wegen $|H| \leqslant |C| - 1$ folgt:

$$|H|/|C| \leqslant 1 - 1/|C| \leqslant 1 - 1/n$$

- Damit: $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.
- Widerspruch: sind schon am Ende der Phase.

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

- Sei C der Min-Mean-Kreis in G_g und H die Kanten mit negativen Kosten in C.
- Damit gilt:

$$\mu(g) = \sum_{e \in C} \frac{-l(e)}{|C|} \leqslant \sum_{e \in H} \frac{-l(e)}{|C|} \leqslant |H| \cdot \frac{\mu(f)}{|C|}$$

- Wegen $|H| \leqslant |C| - 1$ folgt:

$$|H|/|C| \leqslant 1 - 1/|C| \leqslant 1 - 1/n$$

- Damit: $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.
- Widerspruch: sind schon am Ende der Phase.
- Also gibt es in der Phase keine Typ 2 Iterationen.

Laufzeit mit Annahmen (Iterationen pro Phase)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Sei g der Fluss vor der ersten Typ 2 Iteration.
- Die Annahme gilt weiter (keine neuen Kanten mit negativen Kosten):

$$\forall e \in E(G_g) : l(e) \geqslant -\mu(f)$$

- Sei C der Min-Mean-Kreis in G_g und H die Kanten mit negativen Kosten in C.
- Damit gilt:

$$\mu(g) = \sum_{e \in C} \frac{-l(e)}{|C|} \leqslant \sum_{e \in H} \frac{-l(e)}{|C|} \leqslant |H| \cdot \frac{\mu(f)}{|C|}$$

- Wegen $|H| \leqslant |C| - 1$ folgt:

$$|H|/|C| \leqslant 1 - 1/|C| \leqslant 1 - 1/n$$

- Damit: $\mu(g) \leqslant (1 - 1/n) \cdot \mu(f)$.
- Widerspruch: sind schon am Ende der Phase.
- Also gibt es in der Phase keine Typ 2 Iterationen.
- Falls die Annahme gilt, so endet die Phase nach $m - 1$ Typ 1 Iterationen.

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.
- Verhalten des Algorithmus sollte unverändert sein.

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.
- Verhalten des Algorithmus sollte unverändert sein.
- Sei $p : V \mapsto \mathbb{Z}$ ein Potential für die Knoten.

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.
- Verhalten des Algorithmus sollte unverändert sein.
- Sei $p : V \mapsto \mathbb{Z}$ ein Potential für die Knoten.
- Setze für alle $e = (v, w) \in E(G_f)$ setze: $l'(e) = l(e) + p(v) - p(w)$.

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.
- Verhalten des Algorithmus sollte unverändert sein.
- Sei $p : V \mapsto \mathbb{Z}$ ein Potential für die Knoten.
- Setze für alle $e = (v, w) \in E(G_f)$ setze: $l'(e) = l(e) + p(v) - p(w)$.
- Für $\overline{e} = (w, v)$ gilt:

$$\begin{aligned} l'(\overline{e}) &= l(\overline{e}) + p(w) - p(v) \\ & -l(e) + p(w) - p(v) = -l'(e) \end{aligned}$$

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.
- Verhalten des Algorithmus sollte unverändert sein.
- Sei $p : V \mapsto \mathbb{Z}$ ein Potential für die Knoten.
- Setze für alle $e = (v, w) \in E(G_f)$ setze: $l'(e) = l(e) + p(v) - p(w)$.
- Für $\overline{e} = (w, v)$ gilt:

$$\begin{array}{rcl} l'(\overline{e}) & = & l(\overline{e}) + p(w) - p(v) \\ & & -l(e) + p(w) - p(v) = -l'(e) \end{array}$$

- Potentiale ändern die Kostensumme auf Kreisen C nicht: $l(C) = l'(C)$.

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.
- Verhalten des Algorithmus sollte unverändert sein.
- Sei $p : V \mapsto \mathbb{Z}$ ein Potential für die Knoten.
- Setze für alle $e = (v, w) \in E(G_f)$ setze: $l'(e) = l(e) + p(v) - p(w)$.
- Für $\overline{e} = (w, v)$ gilt:

$$\begin{aligned} l'(\overline{e}) &= l(\overline{e}) + p(w) - p(v) \\ & \quad -l(e) + p(w) - p(v) = -l'(e) \end{aligned}$$

- Potentiale ändern die Kostensumme auf Kreisen C nicht: $l(C) = l'(C)$.
- Potentiale ändern damit auch nicht den Ablauf des Verfahrens.

Laufzeit (Erzwinge die Annahme)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

- Wir sorgen dafür, dass die Annahme erfüllt werden kann:
- Wir verändern l geeignet.
- Verhalten des Algorithmus sollte unverändert sein.
- Sei $p : V \mapsto \mathbb{Z}$ ein Potential für die Knoten.
- Setze für alle $e = (v, w) \in E(G_f)$ setze: $l'(e) = l(e) + p(v) - p(w)$.
- Für $\overline{e} = (w, v)$ gilt:

$$\begin{aligned} l'(\overline{e}) &= l(\overline{e}) + p(w) - p(v) \\ & -l(e) + p(w) - p(v) = -l'(e) \end{aligned}$$

- Potentiale ändern die Kostensumme auf Kreisen C nicht: $l(C) = l'(C)$.
- Potentiale ändern damit auch nicht den Ablauf des Verfahrens.
- Wir zeigen nun, dass es Potentiale gibt, die die Annahme erzwingen.

Laufzeit (Erzwinge die Annahme durch Potential)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Existenz von p)

In $G_f = (V, E_f)$ gelte $\bar{l}(C) \geqslant -\mu$ für jeden Kreis C.
Dann gibt es $p : V \mapsto \mathbb{Z}$ mit:
$l'(e) = l(e) + p(w) - p(v) \geqslant -\mu$ für jede Kante $e \in E_f$.

Beweis:

Laufzeit (Erzwinge die Annahme durch Potential)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Existenz von p)

In $G_f = (V, E_f)$ gelte $\bar{l}(C) \geqslant -\mu$ für jeden Kreis C.
Dann gibt es $p : V \mapsto \mathbb{Z}$ mit:
$l'(e) = l(e) + p(w) - p(v) \geqslant -\mu$ für jede Kante $e \in E_f$.

Beweis:

- Für $e \in E_f$ setze: $l_\mu(e) = l(e) + \mu$

Laufzeit (Erzwinge die Annahme durch Potential)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Existenz von p)

*In $G_f = (V, E_f)$ gelte $\bar{l}(C) \geqslant -\mu$ für jeden Kreis C.
Dann gibt es $p : V \mapsto \mathbb{Z}$ mit:
$l'(e) = l(e) + p(w) - p(v) \geqslant -\mu$ für jede Kante $e \in E_f$.*

Beweis:

- Für $e \in E_f$ setze: $l_\mu(e) = l(e) + \mu$
- Für $v \in V$ bestimme Kantenzug P in G_f von beliebigen Knoten w zu v mit minimalen Gewicht. Setze dann $p(v) = l_\mu(P)$.

Laufzeit (Erzwinge die Annahme durch Potential)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Existenz von p)

In $G_f = (V, E_f)$ gelte $\bar{l}(C) \geqslant -\mu$ für jeden Kreis C.
Dann gibt es $p : V \mapsto \mathbb{Z}$ mit:
$l'(e) = l(e) + p(w) - p(v) \geqslant -\mu$ für jede Kante $e \in E_f$.

Beweis:

- Für $e \in E_f$ setze: $l_\mu(e) = l(e) + \mu$
- Für $v \in V$ bestimme Kantenzug P in G_f von beliebigen Knoten w zu v mit minimalen Gewicht. Setze dann $p(v) = l_\mu(P)$.
- Beachte: Das ist wohldefinert, denn aus $\bar{l}(C) \geqslant -\mu$ folgt $l_\mu(C) \geqslant 0$ für beliebigen Kreis C.

Laufzeit (Erzwinge die Annahme durch Potential)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Existenz von p)

*In $G_f = (V, E_f)$ gelte $\bar{l}(C) \geqslant -\mu$ für jeden Kreis C.
Dann gibt es $p : V \mapsto \mathbb{Z}$ mit:
$l'(e) = l(e) + p(w) - p(v) \geqslant -\mu$ für jede Kante $e \in E_f$.*

Beweis:

- Für $e \in E_f$ setze: $l_\mu(e) = l(e) + \mu$
- Für $v \in V$ bestimme Kantenzug P in G_f von beliebigen Knoten w zu v mit minimalen Gewicht. Setze dann $p(v) = l_\mu(P)$.
- Beachte: Das ist wohldefinert, denn aus $\bar{l}(C) \geqslant -\mu$ folgt $l_\mu(C) \geqslant 0$ für beliebigen Kreis C.
- Damit gilt für jede Kante $e = (v, w) \in E_f : p(w) \leqslant p(v) + l_\mu(e)$.

Laufzeit (Erzwinge die Annahme durch Potential)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, **Annahme**: $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Existenz von p)

In $G_f = (V, E_f)$ gelte $\bar{l}(C) \geqslant -\mu$ für jeden Kreis C.
Dann gibt es $p : V \mapsto \mathbb{Z}$ mit:
$l'(e) = l(e) + p(w) - p(v) \geqslant -\mu$ für jede Kante $e \in E_f$.

Beweis:

- Für $e \in E_f$ setze: $l_\mu(e) = l(e) + \mu$
- Für $v \in V$ bestimme Kantenzug P in G_f von beliebigen Knoten w zu v mit minimalen Gewicht. Setze dann $p(v) = l_\mu(P)$.
- Beachte: Das ist wohldefinert, denn aus $\bar{l}(C) \geqslant -\mu$ folgt $l_\mu(C) \geqslant 0$ für beliebigen Kreis C.
- Damit gilt für jede Kante $e = (v, w) \in E_f : p(w) \leqslant p(v) + l_\mu(e)$.
- Es folgt:

$$\begin{aligned} l'(e) &= l(e) + p(v) - p(w) \\ &= l_\mu(e) + p(v) - p(w) - \mu \geqslant -\mu \end{aligned}$$

Laufzeit pro Iteration

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \;\; \mu(f) = -\bar{l}(C), \;\; \forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Karp, 1978)

Ein Min-Mean-Cycle kann in G_f in Zeit $O(nm)$ gefunden werden.

Beweis:

Laufzeit pro Iteration

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \quad \mu(f) = -\bar{l}(C), \quad \forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Karp, 1978)

Ein Min-Mean-Cycle kann in G_f in Zeit $O(nm)$ gefunden werden.

Beweis:

- Setze für $v \in V$ und $k \in \{0, \ldots, n\}$:
  $d_k(v)$ seien die Kosten des günstigsten Kantenzugs nach v mit genau k Kanten.

Laufzeit pro Iteration

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Karp, 1978)

Ein Min-Mean-Cycle kann in G_f in Zeit $O(nm)$ gefunden werden.

Beweis:

- Setze für $v \in V$ und $k \in \{0, \ldots, n\}$:
  $d_k(v)$ seien die Kosten des günstigsten Kantenzugs nach v mit genau k Kanten.
- Es gilt $d_0(v) = 0$ und

$$d_{k+1}(v) = \min_{e=(w,v) \in E_f} (d_k(w) + l(e)).$$

Laufzeit pro Iteration

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$, $\mu(f) = -\bar{l}(C)$, $\forall e \in E(G_f) : l(e) \geqslant -\mu(f)$

Lemma (Karp, 1978)

Ein Min-Mean-Cycle kann in G_f in Zeit $O(nm)$ gefunden werden.

Beweis:

- Setze für $v \in V$ und $k \in \{0, \ldots, n\}$:
  $d_k(v)$ seien die Kosten des günstigsten Kantenzugs nach v mit genau k Kanten.
- Es gilt $d_0(v) = 0$ und

$$d_{k+1}(v) = \min_{e=(w,v) \in E_f} (d_k(w) + l(e)).$$

- Alle $d_k(v)$ Werte können durch dynamische Programmierung in Zeit $O(nm)$ bestimmt werden.

Wert des Min-Mean-Kreises

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \ \mu(f) = -\bar{l}(C)$

Lemma

Der Wert $\bar{l}(C)$ des Min-Mean-Kreises ist:

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Beweis:

Wert des Min-Mean-Kreises

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \quad \mu(f) = -\bar{l}(C)$

Lemma

Der Wert $\bar{l}(C)$ des Min-Mean-Kreises ist:

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Beweis:

Wert des Min-Mean-Kreises

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e\in C} l(e)}{|C|}, \quad \mu(f) = -\bar{l}(C)$

Lemma

Der Wert $\bar{l}(C)$ des Min-Mean-Kreises ist:

$$\alpha = \min_{v\in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Beweis:

- Sei C ein Min-Mean-Kreis.

Wert des Min-Mean-Kreises

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \quad \mu(f) = -\bar{l}(C)$

Lemma

Der Wert $\bar{l}(C)$ des Min-Mean-Kreises ist:

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Beweis:

- Sei C ein Min-Mean-Kreis.
- Wenn alle Kantenkosten um δ erhöht werden, bleibt C Min-Mean-Kreis.

Wert des Min-Mean-Kreises

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \quad \mu(f) = -\bar{l}(C)$

Lemma

Der Wert $\bar{l}(C)$ des Min-Mean-Kreises ist:

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Beweis:

- Sei C ein Min-Mean-Kreis.
- Wenn alle Kantenkosten um δ erhöht werden, bleibt C Min-Mean-Kreis.
- Der Wert von C erhöht sich auch um δ.

Wert des Min-Mean-Kreises

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \quad \mu(f) = -\bar{l}(C)$

Lemma

Der Wert $\bar{l}(C)$ des Min-Mean-Kreises ist:

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Beweis:

- Sei C ein Min-Mean-Kreis.
- Wenn alle Kantenkosten um δ erhöht werden, bleibt C Min-Mean-Kreis.
- Der Wert von C erhöht sich auch um δ.
- Also können wir alle Kantenkosten um δ verschieben bis $\bar{l}(C) = 0$ gilt.

Wert des Min-Mean-Kreises

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}, \quad \mu(f) = -\bar{l}(C)$

Lemma

Der Wert $\bar{l}(C)$ des Min-Mean-Kreises ist:

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Beweis:

- Sei C ein Min-Mean-Kreis.
- Wenn alle Kantenkosten um δ erhöht werden, bleibt C Min-Mean-Kreis.
- Der Wert von C erhöht sich auch um δ.
- Also können wir alle Kantenkosten um δ verschieben bis $\bar{l}(C) = 0$ gilt.
- Zeige nun: $\alpha = 0$ gilt damit auch.

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.
- P muss Kreis C beinhalten.

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.
- P muss Kreis C beinhalten.
- Sei P' der verbleibende Kantenzug mit j Kanten ohne C.

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.
- P muss Kreis C beinhalten.
- Sei P' der verbleibende Kantenzug mit j Kanten ohne C.
- Dann hat C $n-j$ Kanten.

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.
- P muss Kreis C beinhalten.
- Sei P' der verbleibende Kantenzug mit j Kanten ohne C.
- Dann hat C $n-j$ Kanten.
- Wegen $l(C) \geqslant 0$ gilt:

$$d_n(v) = l(P) = l(C) + l(P') \geqslant l(P') \geqslant d_j(v).$$

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j}\right)$$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.
- P muss Kreis C beinhalten.
- Sei P' der verbleibende Kantenzug mit j Kanten ohne C.
- Dann hat C $n-j$ Kanten.
- Wegen $l(C) \geqslant 0$ gilt:

$$d_n(v) = l(P) = l(C) + l(P') \geqslant l(P') \geqslant d_j(v).$$

- Für jeden Knoten v gibt es $j \in \{1, \ldots, n-1\}$ mit $d_n(v) \geqslant d_j(v)$.

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.
- P muss Kreis C beinhalten.
- Sei P' der verbleibende Kantenzug mit j Kanten ohne C.
- Dann hat C $n-j$ Kanten.
- Wegen $l(C) \geqslant 0$ gilt:

$$d_n(v) = l(P) = l(C) + l(P') \geqslant l(P') \geqslant d_j(v).$$

- Für jeden Knoten v gibt es $j \in \{1, \ldots, n-1\}$ mit $d_n(v) \geqslant d_j(v)$.
- Damit $\alpha \geqslant 0$.

Wert des Min-Mean-Kreises (Zeige $\alpha \geqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Sei v beliebig und P Kantenzug, der bei v endet und Kosten $d_n(v)$ hat.
- P muss Kreis C beinhalten.
- Sei P' der verbleibende Kantenzug mit j Kanten ohne C.
- Dann hat C $n-j$ Kanten.
- Wegen $l(C) \geqslant 0$ gilt:

$$d_n(v) = l(P) = l(C) + l(P') \geqslant l(P') \geqslant d_j(v).$$

- Für jeden Knoten v gibt es $j \in \{1, \ldots, n-1\}$ mit $d_n(v) \geqslant d_j(v)$.
- Damit $\alpha \geqslant 0$.
- Im Folgenden zeigen wir nun noch $\alpha \leqslant 0$.

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

$$\mu(f) = -\bar{l}(C)$$

$$\alpha = \min_{v \in V} \max_{j=\mathbf{0}}^{n-\mathbf{1}} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

$$\mu(f) = -\bar{l}(C)$$

$$\alpha = \min_{v \in V} \max_{j=\mathbf{0}}^{n-\mathbf{1}} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Zeige: Es gibt Knoten w mit: $d_n(w) \leqslant d_j(w)$ für alle $j \in \{0, \ldots, n-1\}$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Zeige: Es gibt Knoten w mit: $d_n(w) \leqslant d_j(w)$ für alle $j \in \{0, \ldots, n-1\}$
- Sei v Knoten des Min-Mean-Kreises C.

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Zeige: Es gibt Knoten w mit: $d_n(w) \leqslant d_j(w)$ für alle $j \in \{0, \ldots, n-1\}$
- Sei v Knoten des Min-Mean-Kreises C.
- Sei P kürzester Kantenzug, der bei v endet. O.B.d.A. enthält P keinen Kreis. Sei $p < n$ die Anzahl der Kanten in P

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Zeige: Es gibt Knoten w mit: $d_n(w) \leqslant d_j(w)$ für alle $j \in \{0, \ldots, n-1\}$
- Sei v Knoten des Min-Mean-Kreises C.
- Sei P kürzester Kantenzug, der bei v endet. O.B.d.A. enthält P keinen Kreis. Sei $p < n$ die Anzahl der Kanten in P
- Sei w der Knoten, der auf C von v aus nach $n-p$ Kanten liegt. Diesen Kantenzug nennen wir Q.

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Zeige: Es gibt Knoten w mit: $d_n(w) \leqslant d_j(w)$ für alle $j \in \{0, \ldots, n-1\}$
- Sei v Knoten des Min-Mean-Kreises C.
- Sei P kürzester Kantenzug, der bei v endet. O.B.d.A. enthält P keinen Kreis. Sei $p < n$ die Anzahl der Kanten in P
- Sei w der Knoten, der auf C von v aus nach $n - p$ Kanten liegt. Diesen Kantenzug nennen wir Q.
- Sei R der Weg von w zu v auf C mit r Kanten.

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Zeige: Es gibt Knoten w mit: $d_n(w) \leqslant d_j(w)$ für alle $j \in \{0, \ldots, n-1\}$
- Sei v Knoten des Min-Mean-Kreises C.
- Sei P kürzester Kantenzug, der bei v endet. O.B.d.A. enthält P keinen Kreis. Sei $p < n$ die Anzahl der Kanten in P
- Sei w der Knoten, der auf C von v aus nach $n - p$ Kanten liegt. Diesen Kantenzug nennen wir Q.
- Sei R der Weg von w zu v auf C mit r Kanten.
- Sei $j \in \{0, \ldots, n-1\}$ und S ein Kantenzug minimaler Länge aus j Kanten, der an w endet.

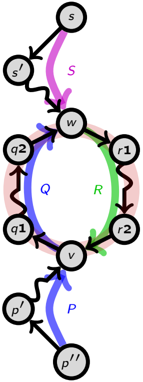
Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Zeige: Es gibt Knoten w mit: $d_n(w) \leqslant d_j(w)$ für alle $j \in \{0, \ldots, n-1\}$
- Sei v Knoten des Min-Mean-Kreises C.
- Sei P kürzester Kantenzug, der bei v endet. O.B.d.A. enthält P keinen Kreis. Sei $p < n$ die Anzahl der Kanten in P
- Sei w der Knoten, der auf C von v aus nach $n - p$ Kanten liegt. Diesen Kantenzug nennen wir Q.
- Sei R der Weg von w zu v auf C mit r Kanten.
- Sei $j \in \{0, \ldots, n-1\}$ und S ein Kantenzug minimaler Länge aus j Kanten, der an w endet.



Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

$$\mu(f) = -\bar{l}(C)$$

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e\in C} l(e)}{|C|}$$

$$\mu(f) = -\bar{l}(C)$$

$$\alpha = \min_{v\in V} \max_{j=0}^{n-1} \left(\frac{d_n(v)-d_j(v)}{n-j}\right)$$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Dann gilt:

$$d_n(w) \leqslant l(P) + l(Q) = d_p(v) + l(Q).$$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$

$$\mu(f) = -\bar{l}(C)$$

$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

- Dann gilt:

$$d_n(w) \leqslant l(P) + l(Q) = d_p(v) + l(Q).$$

- und

$$d_p(v) \leqslant d_{j+r}(v) \leqslant d_j(w) + l(R).$$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$$
$$\mu(f) = -\bar{l}(C)$$
$$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$$

- Dann gilt:

$$d_n(w) \leqslant l(P) + l(Q) = d_p(v) + l(Q).$$

- und

$$d_p(v) \leqslant d_{j+r}(v) \leqslant d_j(w) + l(R).$$

- es folgt:

$$d_n(w) \leqslant d_j(w) + l(R) + l(Q).$$

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Dann gilt:

$$d_n(w) \leqslant l(P) + l(Q) = d_p(v) + l(Q).$$

- und

$$d_p(v) \leqslant d_{j+r}(v) \leqslant d_j(w) + l(R).$$

- es folgt:

$$d_n(w) \leqslant d_j(w) + l(R) + l(Q).$$

- Der Kantenzug $Q \odot R$ hat Kosten 0 (wegen $l(C) = 0$).

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$
$\mu(f) = -\bar{l}(C)$
$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Dann gilt:

$$d_n(w) \leqslant l(P) + l(Q) = d_p(v) + l(Q).$$

- und

$$d_p(v) \leqslant d_{j+r}(v) \leqslant d_j(w) + l(R).$$

- es folgt:

$$d_n(w) \leqslant d_j(w) + l(R) + l(Q).$$

- Der Kantenzug $Q \odot R$ hat Kosten 0 (wegen $l(C) = 0$).
- Damit gilt: $d_n(w) \leqslant d_j(w)$.

Wert des Min-Mean-Kreises (Zeige $\alpha \leqslant 0$)

$\bar{l}(C) = \frac{l(C)}{|C|} = \frac{\sum_{e \in C} l(e)}{|C|}$

$\mu(f) = -\bar{l}(C)$

$\alpha = \min_{v \in V} \max_{j=0}^{n-1} \left(\frac{d_n(v) - d_j(v)}{n-j} \right)$

- Dann gilt:

$$d_n(w) \leqslant l(P) + l(Q) = d_p(v) + l(Q).$$

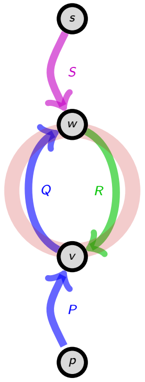
- und

$$d_p(v) \leqslant d_{j+r}(v) \leqslant d_j(w) + l(R).$$

- es folgt:

$$d_n(w) \leqslant d_j(w) + l(R) + l(Q).$$

- Der Kantenzug $Q \odot R$ hat Kosten 0 (wegen $l(C) = 0$).
- Damit gilt: $d_n(w) \leqslant d_j(w)$.



Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.
   - Damit ist der Wert des Min-Mean-Kreises 0.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.
   - Damit ist der Wert des Min-Mean-Kreises 0.
3. Transformiere die Kantenkosten um den Wert α wie in Lemma "Existenz von p" beschrieben.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.
   - Damit ist der Wert des Min-Mean-Kreises 0.
3. Transformiere die Kantenkosten um den Wert α wie in Lemma "Existenz von p" beschrieben.
   - Die Potentiale ergeben sich aus den Werten $d_k(v)$.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.
   - Damit ist der Wert des Min-Mean-Kreises 0.
3. Transformiere die Kantenkosten um den Wert α wie in Lemma "Existenz von p" beschrieben.
   - Die Potentiale ergeben sich aus den Werten $d_k(v)$.
   - Alle Kantenkosten sind nun nicht negativ.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.
   - Damit ist der Wert des Min-Mean-Kreises 0.
3. Transformiere die Kantenkosten um den Wert α wie in Lemma "Existenz von p" beschrieben.
   - Die Potentiale ergeben sich aus den Werten $d_k(v)$.
   - Alle Kantenkosten sind nun nicht negativ.
   - Die Kanten des Min-Mean-Cycle haben Wert 0.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.
   - Damit ist der Wert des Min-Mean-Kreises 0.
3. Transformiere die Kantenkosten um den Wert α wie in Lemma “Existenz von p” beschrieben.
   - Die Potentiale ergeben sich aus den Werten $d_k(v)$.
   - Alle Kantenkosten sind nun nicht negativ.
   - Die Kanten des Min-Mean-Cycle haben Wert 0.
4. Lösche alle Kanten mit Wert größer 0.

Gesamtverfahren

1. Berechne die $d_k(v)$ Werte und bestimme α.
2. Addiere zu allen Kantenkosten um α.
   - Damit ist der Wert des Min-Mean-Kreises 0.
3. Transformiere die Kantenkosten um den Wert α wie in Lemma "Existenz von p" beschrieben.
   - Die Potentiale ergeben sich aus den Werten $d_k(v)$.
   - Alle Kantenkosten sind nun nicht negativ.
   - Die Kanten des Min-Mean-Cycle haben Wert 0.
4. Lösche alle Kanten mit Wert größer 0.
5. Suche mit Tiefensuche Kreis in den verbleibenden Kanten.

Gesamtlaufzeit

Theorem

Der Mean-Cycle-Algorithmus hat eine Laufzeit von $O(m^2 \cdot n^2 \cdot \log(nL))$.

Beweis, siehe obige Überlegungen:

Gesamtlaufzeit

Theorem

Der Mean-Cycle-Algorithmus hat eine Laufzeit von $O(m^2 \cdot n^2 \cdot \log(nL))$.

Beweis, siehe obige Überlegungen:

- $n \log(nL) + 1$ Phasen (L Maximum der absoluten Kantenkosten).

Gesamtlaufzeit

Theorem

Der Mean-Cycle-Algorithmus hat eine Laufzeit von $O(m^2 \cdot n^2 \cdot \log(nL))$.

Beweis, siehe obige Überlegungen:

- $n\log(nL)+1$ Phasen (L Maximum der absoluten Kantenkosten).
- m Iterationen pro Phase.

Gesamtlaufzeit

Theorem

Der Mean-Cycle-Algorithmus hat eine Laufzeit von $O(m^2 \cdot n^2 \cdot \log(nL))$.

Beweis, siehe obige Überlegungen:

- $n \log(nL) + 1$ Phasen (L Maximum der absoluten Kantenkosten).
- m Iterationen pro Phase.
- $O(nm)$ Laufzeit pro Iteration.

Gesamtlaufzeit

Theorem

Der Mean-Cycle-Algorithmus hat eine Laufzeit von $O(m^2 \cdot n^2 \cdot \log(nL))$.

Beweis, siehe obige Überlegungen:

- $n \log(nL) + 1$ Phasen (L Maximum der absoluten Kantenkosten).
- m Iterationen pro Phase.
- $O(nm)$ Laufzeit pro Iteration.

Gesamtlaufzeit

Theorem

Der Mean-Cycle-Algorithmus hat eine Laufzeit von $O(m^2 \cdot n^2 \cdot \log(nL))$.

Beweis, siehe obige Überlegungen:

- $n \log(nL) + 1$ Phasen (L Maximum der absoluten Kantenkosten).
- m Iterationen pro Phase.
- $O(nm)$ Laufzeit pro Iteration.

Theorem

Der Mean-Cycle-Algorithmus hat eine Laufzeit von $O(m^3 \cdot n^2 \cdot \log n)$.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.
- Cormen, Leiserson, Rives: Introduction to Algorithms, First Edition, MIT Press, 1990.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.
- Cormen, Leiserson, Rives: Introduction to Algorithms, First Edition, MIT Press, 1990.
- Cormen, Leiserson, Rives: Introduction to Algorithms, Second Edition, MIT Press, 2001.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.
- Cormen, Leiserson, Rives: Introduction to Algorithms, First Edition, MIT Press, 1990.
- Cormen, Leiserson, Rives: Introduction to Algorithms, Second Edition, MIT Press, 2001.
- Ottmann, Widmayer: Algorithmen und Datenstrukturen. BI-Wiss.-Verl. 1990.

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?
- Was ist die Laufzeit der Ford-Fulkerson Methode mit Breitensuche?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?
- Was ist die Laufzeit der Ford-Fulkerson Methode mit Breitensuche?
- Wie ist die Idee des Min-Cut-Max-Flow Theorems?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?
- Was ist die Laufzeit der Ford-Fulkerson Methode mit Breitensuche?
- Wie ist die Idee des Min-Cut-Max-Flow Theorems?
- Wie wird der Schnitt zu dem maximalen Fluss gefunden?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?
- Was ist die Laufzeit der Ford-Fulkerson Methode mit Breitensuche?
- Wie ist die Idee des Min-Cut-Max-Flow Theorems?
- Wie wird der Schnitt zu dem maximalen Fluss gefunden?
- Was ist die Idee des Algorithmus von Dinitz?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?
- Was ist die Laufzeit der Ford-Fulkerson Methode mit Breitensuche?
- Wie ist die Idee des Min-Cut-Max-Flow Theorems?
- Wie wird der Schnitt zu dem maximalen Fluss gefunden?
- Was ist die Idee des Algorithmus von Dinitz?
- Wie funktioniert die Forward-Propagation?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?
- Was ist die Laufzeit der Ford-Fulkerson Methode mit Breitensuche?
- Wie ist die Idee des Min-Cut-Max-Flow Theorems?
- Wie wird der Schnitt zu dem maximalen Fluss gefunden?
- Was ist die Idee des Algorithmus von Dinitz?
- Wie funktioniert die Forward-Propagation?
- Wie ist die Laufzeit vom Algorithmus von Dinitz?

Fragen(Flüsse)

- Was ist die Laufzeit der Ford-Fulkerson Methode?
- Wann hat die Ford-Fulkerson Methode die maximale Laufzeit?
- Warum liefert die Ford-Fulkerson Methode den maximalen Fluss?
- Was ist die Laufzeit der Ford-Fulkerson Methode mit Breitensuche?
- Wie ist die Idee des Min-Cut-Max-Flow Theorems?
- Wie wird der Schnitt zu dem maximalen Fluss gefunden?
- Was ist die Idee des Algorithmus von Dinitz?
- Wie funktioniert die Forward-Propagation?
- Wie ist die Laufzeit vom Algorithmus von Dinitz?
- Wie ist die Begründung zur Laufzeit vom Algorithmus von Dinitz?

Fragen(Flüsse)

- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten?

Fragen(Flüsse)

- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten?
- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten, wo aber auch ein leerer Fluss erlaubt ist?

Fragen(Flüsse)

- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten?
- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten, wo aber auch ein leerer Fluss erlaubt ist?
- Wie ist die Idee der Algorithmen zu kostenminimalen Flüssen?

Fragen(Flüsse)

- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten?
- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten, wo aber auch ein leerer Fluss erlaubt ist?
- Wie ist die Idee der Algorithmen zu kostenminimalen Flüssen?
- Was ist die Laufzeit der Algorithmen zu kostenminimalen Flüssen?

Fragen(Flüsse)

- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten?
- Wie bestimmt man Flüsse mit einem Mindestfluss auf den Kanten, wo aber auch ein leerer Fluss erlaubt ist?
- Wie ist die Idee der Algorithmen zu kostenminimalen Flüssen?
- Was ist die Laufzeit der Algorithmen zu kostenminimalen Flüssen?
- Gebe die Idee zum Beweis der Laufzeit der Algorithmen zu kostenminimalen Flüssen?