Einführung in Operations Research

Flussprobleme (2)

Prof. Dr. Thomas Slawig

CAU Kiel
Institut für Informatik

Inhalt

Lernziele

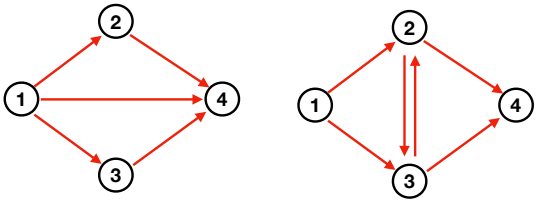
- Das Max-Flow-Problem als Lineares Programm schreiben können
- Den Aufwand der Lösung des Problems mit dem Simplex-Algorithmus angeben können
- Die Motivation für die speziellen Flussalgorithmen kennen
- Algorithmus von Edmonds-Karp erklären können
- Komplexität der Algorithmen von Ford-Fulkerson und Edmonds-Karp mit der Behandlung als LP vergleichen können

Inhalt

Wiederholung: Gerichteter Graph

Definition

Ein **gerichteter Graph/Digraph** (*directed graph*) ist ein Graph, bei dem für die Kantenmenge $E \subseteq V \times V$ gilt, d. h. E ist eine Relation.



Flüsse in Netzwerken

Ein **Netzwerk** (V, E, s, t) ist ein Digraph (V, E) mit zwei ausgewählten Knoten

- $s \in V$: Quelle, *engl.: source*,
- $t \in V$: Senke/Ziel, *engl.: target*.

In einem Netzwerk heißt eine Menge $f := (x_{ij})_{(i,j)\in E}$ von Kantenbewertungen $x_{ij} \in \mathbb{R}_{\geq 0}$ **(s-t)-Fluss**, wenn gilt:

- Flussbedingung: In jeden Knoten, außer Quelle/Senke, fließt so viel hinein wie hinaus, d. h.

$$\sum_{j\in V} x_{ji} - \sum_{j\in V} x_{ij} = 0 \quad \forall i \in V \setminus \{s, t\}.$$

- Ausgehender Fluss der Quelle $=$ eingehender Fluss der Senke $=:$ **Flusswert**, *engl.: value*:

$$\sum_{j\in V} x_{sj} - \sum_{j\in V} x_{js} = \sum_{j\in V} x_{jt} - \sum_{j\in V} x_{tj} =: \mathbf{v(f)}.$$

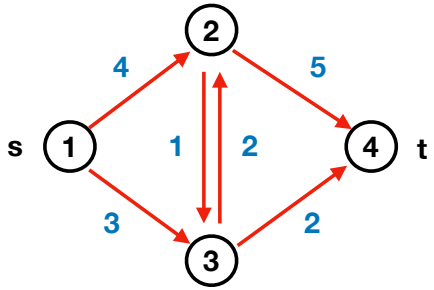
Wir setzen dabei $x_{ij} = 0$ für $(i, j) \notin E$ (Kante existiert nicht).

Beispiel: Fluss im Netzwerk

$V = \{1, 2, 3, 4\}, s = 1, t = 4$

$E = \{(1, 2), (1, 3), (2, 3), (2, 4), (3, 2), (3, 4)\}$

$x_{12} = 4, x_{13} = 3, x_{23} = 1, x_{24} = 5, x_{32} = 2, x_{34} = 2$



Flussbedingung:

$$\forall i \in V \setminus \{s, t\} : \sum_{j \in V} x_{ji} - \sum_{j \in V} x_{ij} = 0.$$

Test: $i = 2$: $\sum_{j \in V} x_{j2} = x_{12} + x_{32} = 4 + 2 = 6$, $\sum_{j \in V} x_{2j} = x_{23} + x_{24} = 1 + 5 = 6 \checkmark$

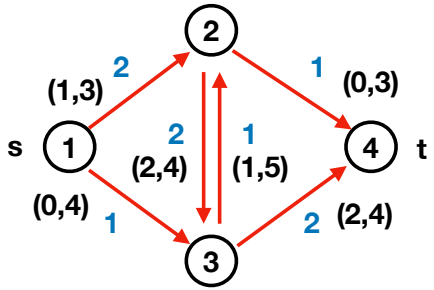
$i = 3$: $\sum_{j \in V} x_{j3} = x_{13} + x_{23} = 3 + 1 = 4$, $\sum_{j \in V} x_{3j} = x_{32} + x_{34} = 2 + 2 = 4 \checkmark$

Kapazitäten

- Zusätzlich gegeben sind oft **untere und obere Kapazitäten**

$$\ell := (\ell_{ij})_{(i,j)\in E}, c := (c_{ij})_{(i,j)\in E}$$
$$\text{mit } 0 \le \ell_{ij} \le c_{ij}.$$

- Wir sprechen dann vom Netzwerk (V, E, s, t, ℓ, c).



Definition

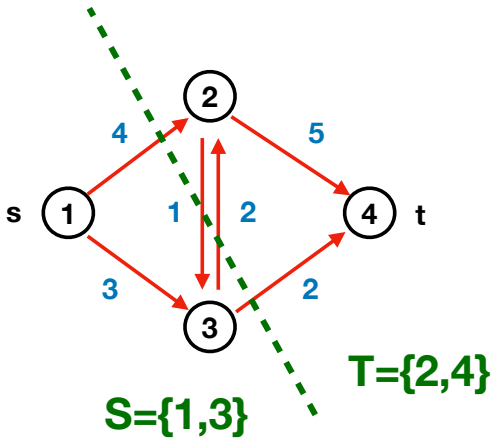
Ein Fluss $f = (x_{ij})_{(i,j)\in E}$, der $\ell_{ij} \le x_{ij} \le c_{ij}$ für alle $(i,j) \in E$ erfüllt, heißt **zulässig**.
Gilt $x_{ij} = c_{ij}$ auf einer Kante $(i,j) \in E$, dann sagen wir, dass der Fluss f die **Kante sättigt**.

Partition und Schnitt

Definition

Sei V eine Knotenmenge.

- Ein Paar (S, T) mit $S, T \subset V, S \cup T = V, S \cap T = \emptyset$ heißt **Partition von** V.
- Eine Partition (S, T) von V mit $s \in S, t \in T$ heißt **(s-t)-Schnitt von** V.



Berechnung des Flusswertes über $(s\text{-}t)$-Schnitte

Lemma (Flusswert bei $(s\text{-}t)$-Schnitt)

Es sei (V, E, s, t) ein Netzwerk und (S, T) ein (s-t)-Schnitt. Dann gilt für den Flusswert:

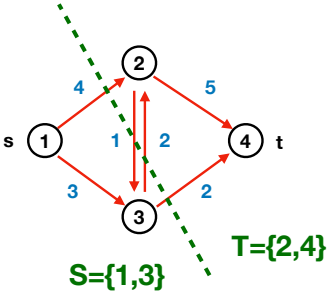
$$v(f) = f(S,T) - f(T,S) = \sum_{i \in S, j \in T} (x_{ij} - x_{ji}) .$$

Beispiel:

$$f(S,T) = \sum_{i=1,3} \sum_{j=2,4} x_{ij} = x_{12} + x_{14} + x_{32} + x_{34}$$
$$= 4 + 0 + 2 + 2 = 8$$

$$f(T,S) = \sum_{i=1,3} \sum_{j=2,4} x_{ji} = x_{21} + x_{23} + x_{41} + x_{43}$$
$$= 0 + 1 + 0 + 0 = 1$$

$$f(S,T) - f(T,S) = 7 = v(f) \quad \checkmark$$



Inhalt

Optimierungsprobleme in Netzwerken

Zusätzliche Parameter:

- Untere und obere Kapazitäten ℓ_{ij}, c_{ij} für Kante $(i,j) \in E$ mit $0 \leq \ell_{ij} \leq c_{ij}$. Ein Fluss f, der $\ell_{ij} \leq x_{ij} \leq c_{ij}$ für alle $(i,j) \in E$ erfüllt, heißt **zulässig**.
- Kosten a_{ij} für eine Flusseinheit durch Kante $(i,j) \in E$.

Optimierungsprobleme in Netzwerken:

1. Bestimme einen zulässigen Fluss, d. h. $f = (x_{ij})$ mit $\ell_{ij} \leq x_{ij} \leq c_{ij}$ für alle $(i,j) \in E$.
2. Max-Flow-Problem: Bestimme einen maximalen (bzw. minimalen) Fluss, d.h. mit maximalem (bzw. minimalem) Flusswert $v(f)$:

$$\max_f v(f) \text{ bei } \ell_{ij} \leq x_{ij} \leq c_{ij} \text{ für alle } (i,j) \in E.$$

3. Bestimme einen zulässigen Fluss der Stärke $\bar{v}$ mit minimalen Kosten:

$$\min_f \sum_{(i,j)\in E} a_{ij} x_{ij} \text{ bei } v(f) = \bar{v}, \ell_{ij} \leq x_{ij} \leq c_{ij} \text{ für alle } (i,j) \in E.$$

Inhalt

Max-Flow-Problem als LP

- Das Max-Flow-Problem

$$\max_f v(f) \text{ bei } \ell_{ij} \leq x_{ij} \leq c_{ij} \text{ für alle } (i,j) \in E$$

- kann als LP geschrieben werden: Wir führen den Flusswert v als zusätzliche Variable ein:

$$\max_{v, x_{ij} \in \mathbb{R}_{\geq 0}, (i,j) \in E} v \quad \text{bei} \begin{cases} \sum_{j \in V} x_{ji} - \sum_{j \in V} x_{ij} = 0 & \text{für alle } i \in V \setminus \{s,t\} \\ \sum_{j \in V} x_{js} - \sum_{j \in V} x_{sj} + v = 0 & \text{(Quelle)} \\ \sum_{j \in V} x_{jt} - \sum_{j \in V} x_{tj} - v = 0 & \text{(Senke)} \\ \ell_{ij} \leq x_{ij} \leq c_{ij} & \text{für alle } (i,j) \in E. \end{cases}$$

Max-Flow-Problem als LP

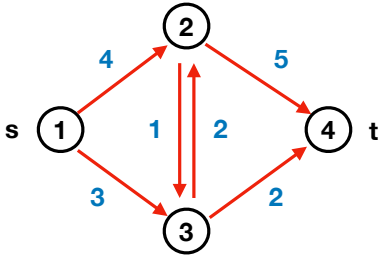
- Verwende Indizierung der Kanten: $e_k \in E, k = 1, \ldots, |E|$
- ... und der Flussvariablen $\hat{x}_k := x_{ij}$ für $e_k = (i, j)$.

Beispiel

$$e_1 = (1,2), e_2 = (1,3), e_3 = (2,3),$$
$$e_4 = (2,4), e_5 = (3,2), e_6 = (3,4)$$
$$\hat{x} = (x_{12}, x_{13}, x_{23}, x_{24}, x_{32}, x_{34})^\top = (4,3,1,5,2,2)^\top.$$

Flussbedingung:

$$x_{12} + x_{32} = x_{23} + x_{24},$$
$$x_{13} + x_{23} = x_{32} + x_{34},$$
$$x_{12} + x_{13} = v = x_{24} + x_{34}$$



Beschreibung des LP mit der Inzidenzmatrix

Definition (Inzidenzmatrix)

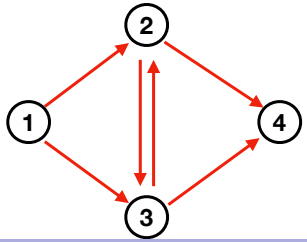
Ein Digraph ist eindeutig durch die **Inzidenzmatrix** $\mathcal{I} \in \{1, 0, -1\}^{|V| \times |E|}$ mit

$$\mathcal{I}_{ik} := \left\{ \begin{array}{rl} 1, & \text{Kante } e_k \text{ beginnt in Knoten } v_i \\ -1, & \text{Kante } e_k \text{ endet in Knoten } v_i, \\ 0, & \text{sonst.} \end{array} \right\}, i = 1, \ldots, |V|, k = 1, \ldots, |E|,$$

beschrieben. Also: Zeile gehört zum Knoten, Spalte zur Kante.

Beispiel

$e_1 = (1, 2), e_2 = (1, 3), e_3 = (2, 3),$
$e_4 = (2, 4), e_5 = (3, 2), e_6 = (3, 4).$
Wie lautet die Inzidenzmatrix für diesen Graphen?



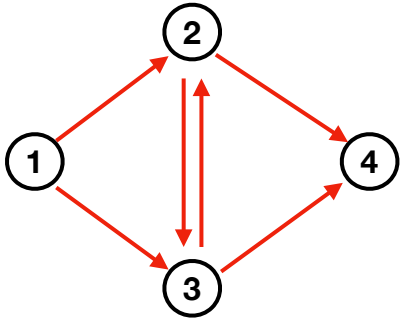
Inzidenzmatrix

Inzidenzmatrix:

$$\mathcal{I}_{ik} := \left\{ \begin{array}{rl} 1, & \text{Kante } e_k \text{ beginnt in Knoten } v_i \\ -1, & \text{Kante } e_k \text{ endet in Knoten } v_i, \\ 0, & \text{sonst.} \end{array} \right\}, i = 1, \ldots, |V|, k = 1, \ldots, |E|.$$

Beispiel

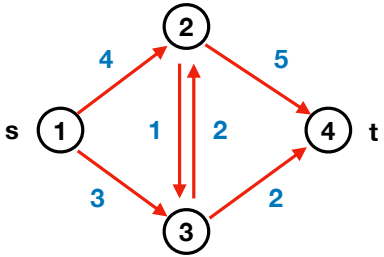
$$\mathcal{I} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 & 0 & -1 \end{bmatrix}$$



Max-Flow-Problem als LP

Beispiel: Flussbedingung:

$$x_{12} + x_{32} = x_{23} + x_{24}, \quad x_{13} + x_{23} = x_{32} + x_{34},$$
$$x_{12} + x_{13} = v = x_{24} + x_{34}$$



- Was ergibt die Multiplikation der Inzidenzmatrix mit dem so indizierten Vektor $\hat{x}$

$$\hat{x} = (x_{12}, x_{13}, x_{23}, x_{24}, x_{32}, x_{34})^\top = (4, 3, 1, 5, 2, 2)^\top$$

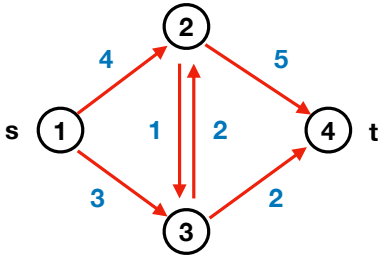
der Flussvariablen?

$$\mathcal{I}\hat{x} = ?$$

Max-Flow-Problem als LP

Beispiel: Flussbedingung:

$$x_{12} + x_{32} = x_{23} + x_{24}, \quad x_{13} + x_{23} = x_{32} + x_{34},$$
$$x_{12} + x_{13} = v = x_{24} + x_{34}$$



Multiplikation der Inzidenzmatrix mit dem Vektor der Flussvariablen:

$$\mathcal{I}\hat{x} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_{12} \\ x_{13} \\ x_{23} \\ x_{24} \\ x_{32} \\ x_{34} \end{bmatrix} = \begin{bmatrix} x_{12} + x_{13} \\ -x_{12} + x_{23} + x_{24} - x_{32} \\ -x_{13} - x_{23} + x_{32} + x_{34} \\ -x_{24} - x_{34} \end{bmatrix} = \begin{bmatrix} v \\ 0 \\ 0 \\ -v \end{bmatrix}$$

Max-Flow-Problem als LP

- Vertausche einfach nur die Reihenfolge der Summen im Max-Flow-Problem, um es mit der Inzidenzmatrix zu schreiben:

$$\max_{v, x_{ij} \in \mathbb{R}_{\geq 0}, (i,j) \in E} v \quad \text{bei} \quad \begin{cases} \sum_{j \in V} x_{ij} - \sum_{j \in V} x_{ji} = 0 & \text{für alle } i \in V \setminus \{s,t\} \\ \sum_{j \in V} x_{sj} - \sum_{j \in V} x_{js} - v = 0 & \text{(Quelle)} \\ \sum_{j \in V} x_{tj} - \sum_{j \in V} x_{jt} + v = 0 & \text{(Senke)} \\ \ell_{ij} \leq x_{ij} \leq c_{ij} & \text{für alle } (i,j) \in E. \end{cases}$$

- Wie lautet das Problem als LP?

Max-Flow-Problem als LP

- Max-Flow-Problem:

$$\max_{v, x_{ij} \in \mathbb{R}_{\geq 0}, (i,j) \in E} v \quad \text{bei} \begin{cases} \sum_{j \in V} x_{ij} - \sum_{j \in V} x_{ji} = 0 & \text{für alle } i \in V \setminus \{s,t\} \\ \sum_{j \in V} x_{sj} - \sum_{j \in V} x_{js} - v = 0 & \text{(Quelle)} \\ \sum_{j \in V} x_{tj} - \sum_{j \in V} x_{jt} + v = 0 & \text{(Senke)} \\ \ell_{ij} \leq x_{ij} \leq c_{ij} & \text{für alle } (i,j) \in E. \end{cases}$$

- Wir erhalten das LP:

$$\max_{(\hat{x}, v) \in \mathbb{R}^{|E|+1}} v \quad \text{bei} \left[\underbrace{\mathcal{I}}_{\in \mathbb{R}^{|V| \times |E|}}, \underbrace{e_t - e_s}_{\in \mathbb{R}^{|V| \times 1}} \right] \begin{bmatrix} \hat{x} \\ v \end{bmatrix} = 0, \quad \hat{\ell} \leq \hat{x} \leq \hat{c}$$

- mit Standardbasisvektoren $e_s, e_t \in \mathbb{R}^{|V|}$.

Max-Flow-Problem als LP

- Das so als LP umgeschriebene Max-Flow-Problem ist noch nicht in Standardform:

$$\max_{(\hat{x},v)\in\mathbb{R}^{|E|+1}} v \quad \text{bei } [\mathcal{I}, e_t - e_s] \begin{bmatrix} \hat{x} \\ v \end{bmatrix} = 0,\ \hat{\ell} \leq \hat{x} \leq \hat{c}.$$

- Wie lautet das Problem nach der Transformation auf Standardform?

Max-Flow-Problem als LP

- Max-Flow-Problem als LP

$$\max_{(\hat{x},v)\in\mathbb{R}^{|E|+1}} v \quad \text{bei} \quad [\mathcal{I}, e_t - e_s] \begin{bmatrix} \hat{x} \\ v \end{bmatrix} = 0, \ \hat{\ell} \le \hat{x} \le \hat{c}.$$

- Ungleichheitsnebenbedingungen

$$\hat{\ell} \le \hat{x} \iff -\hat{x} \le -\hat{\ell} \iff -\hat{x} + s = -\hat{\ell},$$
$$\hat{x} \le \hat{c} \iff \hat{x} + w = \hat{c}$$

mit Slackvariablen $s, w \in \mathbb{R}^{|E|}$. Ergibt Problem in Standardform

$$\min_{z\in\mathbb{R}^n}(-v) \quad \text{bei} \quad Az = b, z \ge 0$$

$$\text{mit} \quad z = (\hat{x}, v, s, w), A = \begin{bmatrix} \mathcal{I} & e_t - e_s & 0 & 0 \\ -I & 0 & I & 0 \\ I & 0 & 0 & I \end{bmatrix}, b = \begin{bmatrix} 0 \\ -\hat{\ell} \\ \hat{c} \end{bmatrix}.$$

Max-Flow-Problem als LP

- Max-Flow-Problem als LP in Standardform

$$\min_{z \in \mathbb{R}^n} (-v) \quad \text{bei} \quad Az = b, z \geq 0$$

$$\text{mit} \quad z = (\hat{x}, v, s, w), A = \begin{bmatrix} \mathcal{I} & e_t - e_s & 0 & 0 \\ -I & 0 & I & 0 \\ I & 0 & 0 & I \end{bmatrix}, b = \begin{bmatrix} 0 \\ -\hat{\ell} \\ \hat{c} \end{bmatrix}.$$

- Was ist die Dimension des so erhaltenen LP?

Max-Flow-Problem als LP

- Max-Flow-Problem als LP in Standardform

$$\min_{z \in \mathbb{R}^n} (-v) \quad \text{bei} \quad Az = b, z \geq 0$$

$$\text{mit} \quad z = (\hat{x}, v, s, w), A = \begin{bmatrix} \mathcal{I} & e_t - e_s & 0 & 0 \\ -I & 0 & I & 0 \\ I & 0 & 0 & I \end{bmatrix}, b = \begin{bmatrix} 0 \\ -\hat{\ell} \\ \hat{c} \end{bmatrix}.$$

- # Slackvariablen: $2|E|$.
- Anzahl der Unbekannten damit insgesamt: $n = 3|E| + 1$
- Anzahl der Gleichheitsnebenbedingungen in Standardform insgesamt: $m = |V| + 2|E|$.
- Es ist normalerweise $|E| \gg |V|$ (mehr Kanten als Knoten).
- Wieviele Schritte braucht der Simplex-Algorithmus im „Mittel“, im Worst-Case?

Max-Flow-Problem als LP

- Anzahl der Unbekannten insgesamt: $n = 3|E| + 1$
- Anzahl der Gleichheitsnebenbedingungen in Standardform insgesamt: $m = |V| + 2|E|$.
- Der Simplex-Algorithmus benötigt **im Mittel** $\mathcal{O}(m) = \mathcal{O}(|V| + 2|E|)$ Schritte.
- Erinnerung: **Worst-Case: exponentiell viele**.
- Jeder Schritt $\mathcal{O}((n - m)m)$ Operationen (Gauß-Algorithmus).
- Insgesamt also **im Mittel**

$$\mathcal{O}((n - m)m^2) = \mathcal{O}\left((|E| - |V|)(|V| + 2|E|)^2\right)$$

  Operationen.
- Motivation für andere Algorithmen.

Inhalt

Flusserhöhende Wege

Definition (Flusserhöhender Weg)

Gegeben seien ein Netzwerk (V, E, s, t, ℓ, c). Ein **flusserhöhender (s-t)-Weg** (für f) ist ein ungerichteter Weg W von s nach t mit

- $x_{ij} < c_{ij}$ auf allen Vorwärtskanten und
- $x_{ij} > \ell_{ij}$ auf allen Rückwärtskanten.

Die Größe

$$\begin{aligned}\delta_W \quad := \min \big(&\{c_{ij} - x_{ij} : (i,j) \text{ ist Vorwärtskante auf } W\} \\ &\cup \{x_{ij} - \ell_{ij} : (i,j) \text{ ist Rückwärtskante auf } W\}\big)\end{aligned} \tag{1}$$

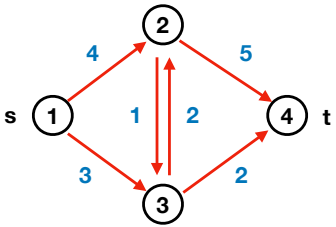
ist der Wert, um den sich der Fluss auf dem flusserhöhenden Weg W erhöhen lässt.

Beispiel: Flusserhöhende Wege

- Fluss mit $s = 1, t = 4$:

$$f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{32}, x_{34}) = (4, 3, 1, 5, 2, 2).$$

- Seien Kapazitäten $\ell_{ij} = 0, c_{ij} = 5$ f. a. $(i, j) \in E$ gegeben.
- Flusserhöhende (1-4)-Wege:



$$W = (1, 3, 4), \text{nur Vorwärtskanten} : x_{13} = 3 < 5, x_{34} = 2 < 5$$
$$\delta_W = \min\{c_{13} - x_{13}, c_{34} - x_{34}\} = \min\{2, 3\} = 2.$$
$$W = (1, 2, 3, 4) \text{ mit Vorwärtskante } (2, 3) : x_{12} = 4 < 5, x_{23} = 1 < 5, x_{34} = 2 < 5,$$
$$\delta_W = \min\{1, 4, 3\} = 1.$$
$$W = (1, 2, 3, 4) \text{ mit Rückwärtskante } (3, 2) : x_{12} = 4 < 5, x_{32} = 2 > 0, x_{34} = 2 < 5,$$
$$\delta_W = \min(\{1, 3\} \cup \{2\}) = 1.$$

Neuer zulässiger Fluss durch flusserhöhenden Weg

Lemma

Es sei $f = (x_{ij})_{(i,j)\in E}$ ein Fluss mit Flusswert v und W ein flusserhöhender $(s\text{-}t)$-Weg mit δ_W wie in (1), Seite 28. Dann ist der Fluss $f^W := (x_{ij}^W)_{(i,j)\in E}$, definiert durch

$$x_{ij}^W = \begin{cases} x_{ij} + \delta_W, & (i,j) \text{ Vorwärtskante auf } W \\ x_{ij} - \delta_W, & (i,j) \text{ Rückwärtskante auf } W \\ x_{ij}, & \text{sonst} \end{cases}$$

ein Fluss mit Flusswert $v + \delta_W$. Ist f zulässig, so ist auch f^W zulässig.

Lemma

Ein zulässiger Fluss f ist genau dann maximal, wenn es keinen flusserhöhenden $(s\text{-}t)$-Weg gibt. Dann gibt es einen $(s\text{-}t)$-Schnitt (S, T) mit

$$v(f) = c(S, T) - \ell(T, S).$$

Algorithmus von Ford-Fulkerson

Idee: Starte mit einem $(s\text{-}t)$-Fluss und erhöhe entlang flusserhöhender $(s\text{-}t)$-Wege solange, bis kein flusserhöhender $(s\text{-}t)$-Weg mehr existiert.

Algorithmus Ford-Fulkerson

1. Gegeben: $(V, E), s, t \in V, (c_{ij})_{(i,j)\in E}, (\ell_{ij})_{(i,j)\in E}$ und zulässiger Ausgangsfluss f.
2. Finde einen flusserhöhenden $(s\text{-}t)$-Weg W.
   - Bestimme δ_W.
   - Update f wie folgt:

$$x_{ij} := \begin{cases} x_{ij} + \delta_W, & (i,j) \text{ Vorwärtskante auf } W, \\ x_{ij} - \delta_W, & (i,j) \text{ Rückwärtskante auf } W, \\ x_{ij}, & \text{sonst.} \end{cases}$$

   - Gehe zu Schritt 2.

   Wenn es keinen flusserhöhenden $(s\text{-}t)$-Weg gibt: Breche ab.

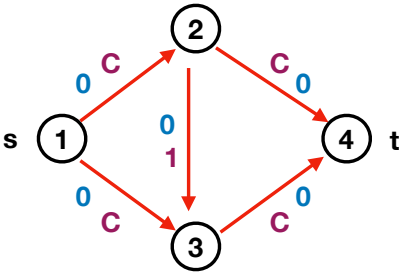
Algorithmus von Ford-Fulkerson: Beispiel (1)

Gegeben:

- $\ell_{ij} = 0, c_{ij} = C$ mit $C \in \mathbb{N}_{>0}$ fest
- $s = 1, t = 4$, Ausgangsfluss: $f \equiv 0$.

1. Iteration des Algorithmus:
   - Wähle $(s\text{-}t)$-Weg $W = (1, 2, 4)$, $\delta_W = C$
   - $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (C, 0, 0, C, 0), v(f) = C$
2. Iteration:
   - Wähle $(s\text{-}t)$-Weg $W = (1, 3, 4)$, $\delta_W = C$
   - $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (C, C, 0, C, C), v(f) = 2C$

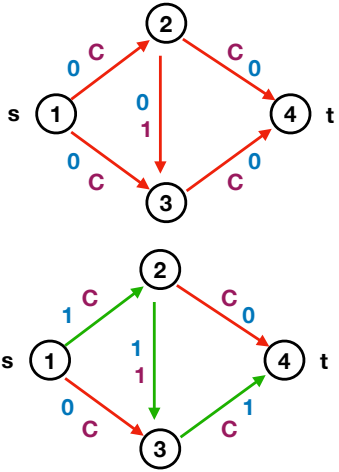
$\rightsquigarrow$ Terminiert nach 2 Iterationen.



Algorithmus von Ford-Fulkerson: Beispiel (2)

Aber:
Kann durch ungünstige Wahl auch anders ablaufen (vgl. letzte VL):

1. Iteration des Algorithmus:
   - Wähle z. B. $(s\text{-}t)$-Weg $W = (1, 2, 3, 4)$
   - $\delta_W = 1$ wegen $c_{23} = 1$
   - $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (1, 0, 1, 0, 1)$
   - $v(f) = 1$

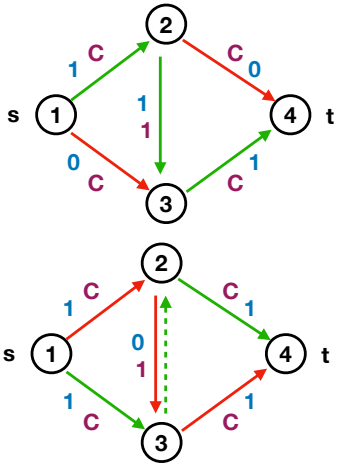


Algorithmus von Ford-Fulkerson: Beispiel (3)

2. Iteration:
   - Wähle $W = (1, 3, 2, 4)$ mit Rückwärtskante (2,3),
   - $\delta_W = 1$ wegen $\ell_{23} = 0$, Erinnerung:

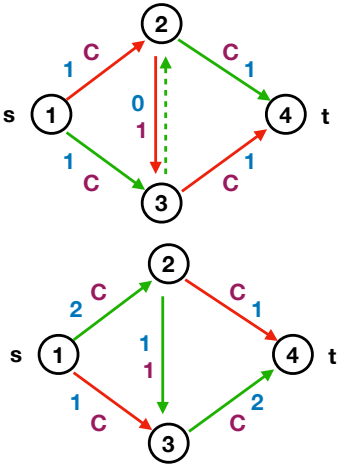
$$\delta_W := \min\big(\{c_{ij} - x_{ij} : (i,j) \text{ ist Vorwärtskante auf } W\} \cup \{x_{ij} - \ell_{ij} : (i,j) \text{ ist Rückwärtskante auf } W\}\big)$$

   - $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (1, 1, 0, 1, 1)$
   - $v(f) = 2$



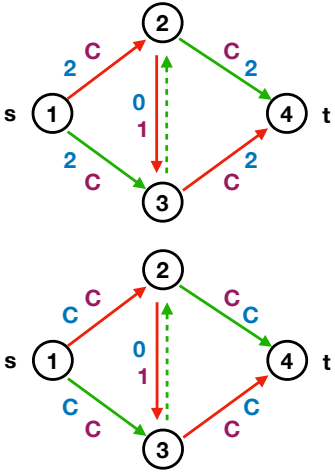
Algorithmus von Ford-Fulkerson: Beispiel (4)

3. Iteration:
   - Wähle wieder (wie in 1. Iteration) $W = (1, 2, 3, 4)$
   - $\delta_W = 1$ wegen $c_{23} = 1$
   - $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (2, 1, 1, 2, 1)$
   - $v(f) = 3$



Algorithmus von Ford-Fulkerson: Beispiel (5)

4. Iteration wie 2. Iteration:
   - Wähle wieder $W = (1, 3, 2, 4)$ mit Rückwärtskante (2,3),
   - $\delta_W = 1$ wegen $\ell_{23} = 0$
   - $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (2, 2, 0, 2, 2)$
   - $v(f) = 4$
5. Iteration wie 3. Iteration usw.
   - ... bis: Kapazität C erreicht
   - Am Ende: $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (C, C, 0, C, C)$
   - $\rightsquigarrow$ maximaler Flusswert $v(f) = 2C$
   - $2C$ Iterationen nötig in diesem Beispiel.



Aufwand Algorithmus von Ford-Fulkerson

Lemma

Die Laufzeit des Algorithmus bei ganzzahliger Eingabe ist $\mathcal{O}(|E| \max v)$.

Beweis.

- Aufwand für das Markieren aller in Frage kommenden Knoten in Schritt 2: $\mathcal{O}(|E|)$
- Bei ganzzahligen Werten x_{ij}: $\delta_W = 1$ möglich für den flusserhöhenden Weg in jedem Schritt
- $\rightsquigarrow$ Anzahl der Iterationen: $\mathcal{O}(\max v)$
- $\rightsquigarrow$ Gesamtaufwand $\mathcal{O}(|E| \max v)$.

□

Bemerkung

Für beliebige, nicht ganzzahlige Kapazitäten terminiert der Algorithmus im Allgemeinen nicht.

Inhalt

Der Algorithmus von Edmonds-Karp

Idee: Wähle im Algorithmus von Ford-Fulkerson einen flusserhöhenden Weg mit einer minimalen Anzahl von Kanten.

1. Gegeben: $(V, E), s, t \in V, (c_{ij})_{(i,j)\in E}, (\ell_{ij})_{(i,j)\in E}$ und zulässiger Ausgangsfluss f
2. Finde flusserhöhenden $(s\text{-}t)$-Weg W mit minimaler Anzahl von Kanten
   - Bestimme δ_W
   - Update f wie im Algorithmus von Ford-Fulkerson
   - Gehe zu Schritt 2

   Wenn es keinen flusserhöhenden $(s\text{-}t)$-Weg gibt: Breche ab.

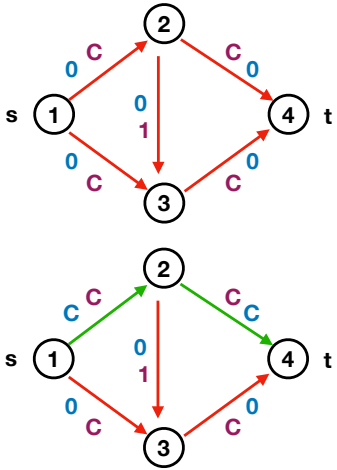
Algorithmus von Edmonds-Karp: Beispiel (s.o.)

Gegeben wie oben:

- $\ell_{ij} = 0$, $c_{ij} = C$ mit $C \in \mathbb{N}_{>0}$ fest
- $s = 1, t = 4$
- Ausgangsfluss: $f \equiv 0$.

1. Iteration des Algorithmus:

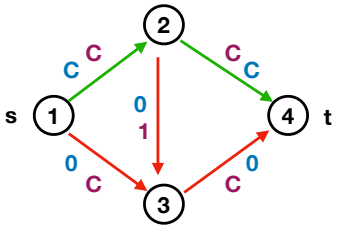
- Wähle flusserhöhenden Weg mit minimaler Kantenanzahl, z. B. $W = (1, 2, 4)$
  (statt: $W = (1, 2, 3, 4)$ wie in Ford-Fulkerson möglich).
- $\delta_W = C$
- $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (C, 0, 0, 0, C)$, $v(f) = C$



Algorithmus von Edmonds-Karp: Beispiel (2)

2. Iteration liefert bereits das Endergebnis:
   - Wähle flusserhöhenden Weg mit minimaler Kantenanzahl, einziger ist $W = (1, 3, 4)$
   - $\delta_W = C$
   - $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (C, C, 0, C, C)$
   - $v(f) = 2C$

Maximaler Flusswert erreicht. Es existieren keine flusserhöhenden Wege (von $s = 1$ ausgehend) mehr $\rightsquigarrow$ Abbruch.



Achtung: Dies ist nur ein Beispiel.

Aufwand des Algorithmus von Edmonds-Karp

Satz

Der Edmonds-Karp-Algorithmus benötigt maximal $2|E|\lceil|V|/4\rceil$ *Schritte (und damit flusserhöhende Wege). Er findet also einen maximalen Fluss mit Aufwand* $\mathcal{O}(|E|^2|V|)$.

Lernziele

- Das Max-Flow-Problem als Lineares Programm schreiben können
- Den Aufwand der Lösung des Problems mit dem Simplex-Algorithmus angeben können
- Die Motivation für die speziellen Flussalgorithmen kennen
- Algorithmus von Edmonds-Karp erklären können
- Komplexität der Algorithmen von Ford-Fulkerson und Edmonds-Karp mit der Behandlung als LP vergleichen können