Einführung in Operations Research

Flussprobleme (1)

Prof. Dr. Thomas Slawig

CAU Kiel
Institut für Informatik

Inhalt

Lernziele

- Den Begriff von Flüssen in Netzwerken erklären können.
- Den Begriff der Kapazitäten und der Zulässigkeit in Netzwerken erklären können.
- Erklären können, was das Max-Flow-Problem ist.
- Den Algorithmus von Ford-Fulkerson motivieren, ...
- ... anwenden ...
- ... und implementieren können.

Inhalt

Wiederholung: Ungerichteter Graph

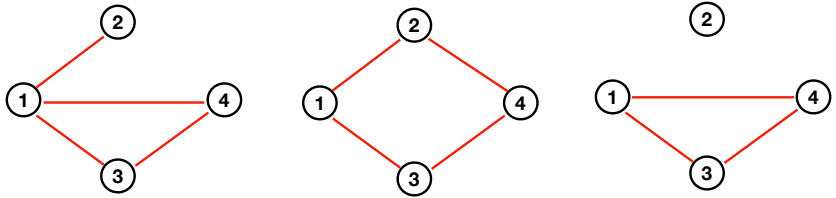
Definition

Ein **(ungerichteter) Graph** ist ein Paar (V, E) bestehend aus

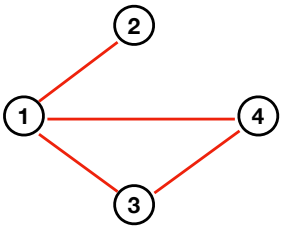
- einer endlichen **Knotenmenge** V (*vertices*)
- und einer **Kantenmenge** $E \subseteq \big\{\{u, v\} : u, v \in V, u \neq v\big\}$ (*edges*).

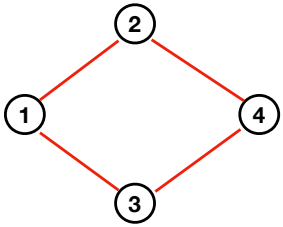
Bemerkung

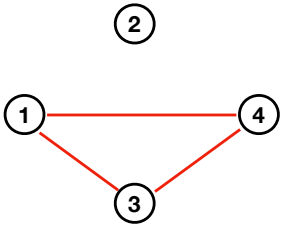
Oft bezeichnen wir die Knotenmenge kurz mit $V = \{1, \ldots, n\}$ *statt* $V = \{v_1, \ldots, v_n\}, n \in \mathbb{N}$.



Beispiele: Ungerichtete Graphen





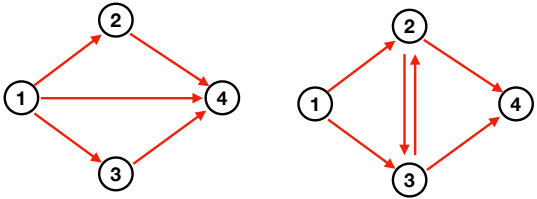


- $V = \{1, 2, 3, 4\}$
- links: $E = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{3, 4\}\} = \{\{2, 1\}, \{3, 1\}, \{4, 1\}, \{4, 3\}\}$
- Mitte: $E = \{\{1, 2\}, \{1, 3\}, \{2, 4\}, \{3, 4\}\} = \ldots$
- rechts: $E = \{\{1, 3\}, \{1, 4\}, \{3, 4\}\} = \ldots$
- E: Menge von Mengen mit je zwei Elementen.

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.

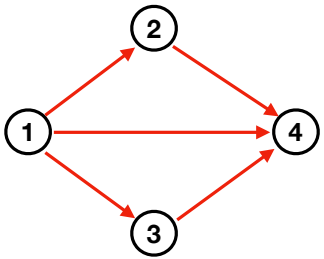


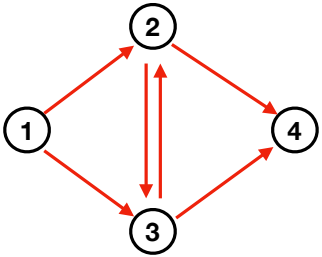
Bemerkung

Alternative Definition: Ungerichteter Graph als Spezialfall eines gerichteten Graphen mit

$$(u, v) \in E \iff (v, u) \in E \quad \forall u, v \in V.$$

Beispiele: Gerichtete Graphen





- $V = \{1, 2, 3, 4\}$
- links: $E = \{(1,2), (1,3), (1,4), (2,4), (3,4)\} \neq \{(2,1), (3,1), (4,1), (4,2), (4,3)\}$
- rechts: $E = \{(1,2), (1,3), (2,3), (2,4), (3,2), (3,4)\} \neq \ldots$
- E: Menge von geordneten Paaren.

Inhalt

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.

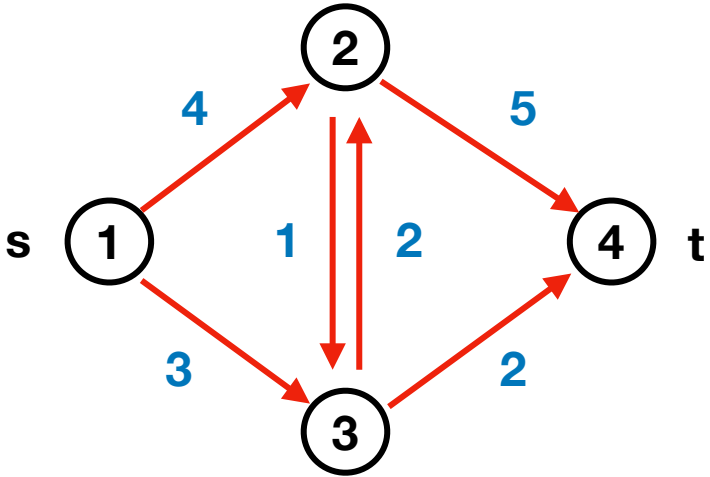
$$\sum_{j\in V} x_{ji} - \sum_{j\in V} x_{ij} = 0 \quad \forall i \in V \setminus \{s, t\}.$$

- 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

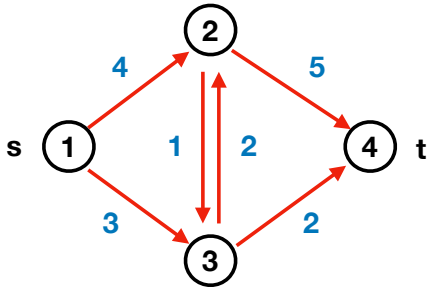


Beispiel: Fluss im Netzwerk

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

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

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



Flussbedingung:

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

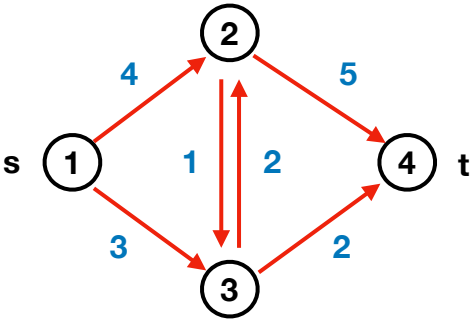
Test:

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

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

Fluss im Netzwerk: Motivation

- Übertragung von Energie/Daten durch ein Netzwerk von Quelle s zu Senke t.
- Jede Kante $(i, j) \in E$ im Netzwerk hat einen nichtnegativen Flusswert x_{ij}.
- Bilanz von ausgehenden und eingehenden Flusswerten an jedem Knoten (außer Quelle, Senke) ist ausgeglichen.
- Flusswert $v(f)$ des gesamten Netzwerkes ist bestimmt aus Überschuss der Quelle, entspricht dem Flusswert an der Senke.

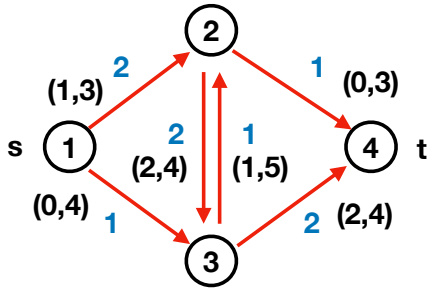


Kapazitäten

- 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**.

Inhalt

Teilflüsse

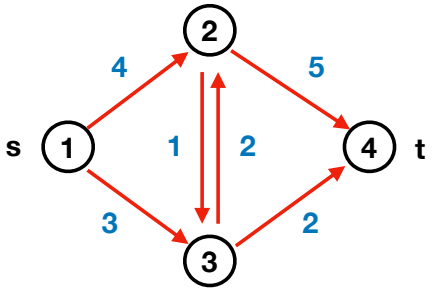
Definition

Für ein Netzwerk (V, E, s, t) und $i, j \in V, S, T \subset V$ definieren wir

$$f(i, T) := \sum_{j \in T} x_{ij}, \quad f(S, j) := \sum_{i \in S} x_{ij}, \quad f(S, T) := \sum_{i \in S, j \in T} x_{ij}.$$

Beispiel

$$f(1, \{2, 4\}) = x_{12} = 4,$$
$$f(\{2, 3\}, 4) = x_{24} + x_{34} = 5 + 2 = 7,$$
$$f(\{1, 2\}, \{3, 4\}) = x_{13} + x_{23} + x_{24} = 3 + 1 + 5 = 9,$$



Definition Fluss über Teilflüsse

Damit lauten die Bedingungen in der Definition eines Flusses:

- Flussbedingung:

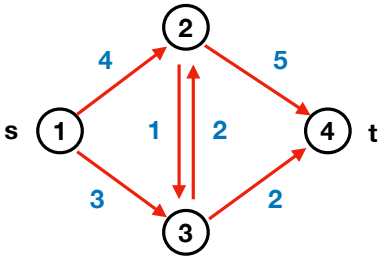
$$\sum_{j \in V} x_{ji} - \sum_{j \in V} x_{ij} = f(V, i) - f(i, V) = 0$$

$$\forall i \in V \setminus \{s, t\}.$$

- Bedingung an Quelle/Senke (Flusswert):

$$v(f) = \sum_{j \in V} x_{sj} - \sum_{j \in V} x_{js} = f(s, V) - f(V, s)$$

$$= \sum_{j \in V} x_{jt} - \sum_{j \in V} x_{tj} = f(V, t) - f(t, V).$$

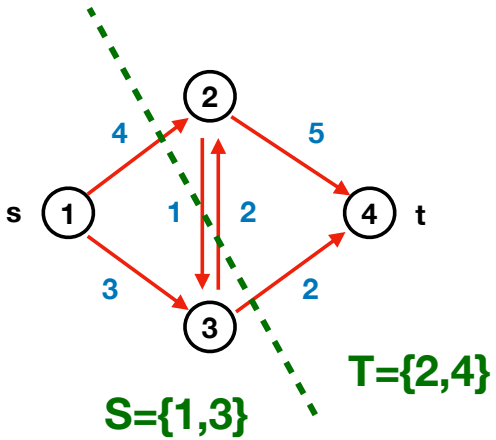


Partition und Schnitt

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:

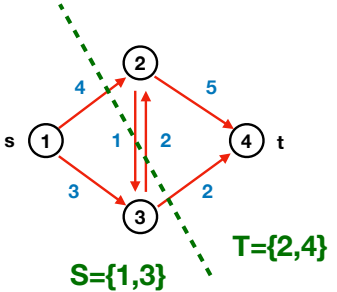
$$v(f) = f(S,T) - f(T,S) = \sum_{i \in S, j \in T} (x_{ij} - x_{ji}) .$$

Beispiel:

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

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

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



Beweis: Flusswert über $(s\text{-}t)$-Schnitte

- Nach Definition gilt

$$v(f) = f(s, V) - f(V, s) = \sum_{i \in S} \underbrace{(f(i, V) - f(V, i))}_{=0 \text{ für } i \neq s}.$$

- Wegen $V = S \cup T, S \cap T = \emptyset$ folgt

$$\begin{aligned} v(f) &= \sum_{i \in S} (f(i, S \cup T) - f(S \cup T, i)) \\ &= \sum_{i \in S} (f(i, S) + f(i, T) - f(S, i) - f(T, i)) \\ &= \sum_{i \in S} (f(i, S) - f(S, i)) + \sum_{i \in S} (f(i, T) - f(T, i)) \\ &= \underbrace{f(S, S) - f(S, S)}_{=0} + f(S, T) - f(T, S). \quad \square \end{aligned}$$

Kapazität eines Schnittes

Definition (Kapazität eines Schnittes)

Für ein Netzwerk (V, E, s, t, ℓ, c) und einen $(s\text{-}t)$-Schnitt (S, T) definieren wir

$$c(S,T) := \sum_{i \in S, j \in T} c_{ij}, \quad \ell(T,S) := \sum_{i \in S, j \in T} \ell_{ji}.$$

Die Größe

$$c(S,T) - \ell(T,S) = \sum_{i \in S, j \in T} (c_{ij} - \ell_{ji})$$

heißt **Kapazität** des Schnittes.

Kapazität eines Schnittes

Lemma

Es sei (V, E, s, t, ℓ, c) ein Netzwerk mit $(t, s) \notin E$. Dann gilt für jeden zulässigen Fluss f und jeden (s-t)-Schnitt

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

also

$$v(f) \leq \min \big\{ c(S, T) - \ell(T, S) : (S, T) \text{ ist } (s\text{-}t)\text{-Schnitt} \big\}.$$

Beweis.

Es ist $v = f(S, T) - f(T, S)$ (Lemma Seite 19). Außerdem gilt $\ell(T, S) \leq f(S, T) \leq c(S, T)$ wegen der Zulässigkeit von f. □

Inhalt

Optimierungsprobleme in Netzwerken

Typische Aufgabenstellungen:

- Bestimme einen zulässigen Fluss f.
- **Max-Flow-Problem**: Bestimme einen maximalen zulässigen Fluss:

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

- Analog: minimaler Fluss.

Inhalt

Wiederholung Graphen: Wege

Definition

In einem ungerichteten Graphen (V, E) heißt $(v_1, \ldots, v_n), v_i \in V$, **Weg** von v_1 zu v_n, wenn gilt:

$$\{v_i, v_{i+1}\} \in E \quad \text{für alle } i \in \{1, \ldots, n-1\}.$$

Definition

In einem Digraphen (V, E) heißt $W = (v_1, \ldots, v_n), v_i \in V$, **ungerichteter Weg** von v_1 zu v_n, wenn gilt:

$$(v_i, v_{i+1}) \in E \text{ oder } (v_{i+1}, v_i) \in E \quad \text{für alle } i \in \{1, \ldots, n-1\}.$$

Eine Kante $(v_i, v_{i+1}) \in E$ heißt **Vorwärtskante** in W, eine Kante $(v_{i+1}, v_i) \in E$ **Rückwärtskante**.

Beispiele

- $W = (1, 3, 2, 4)$
  Vorwärtskanten:

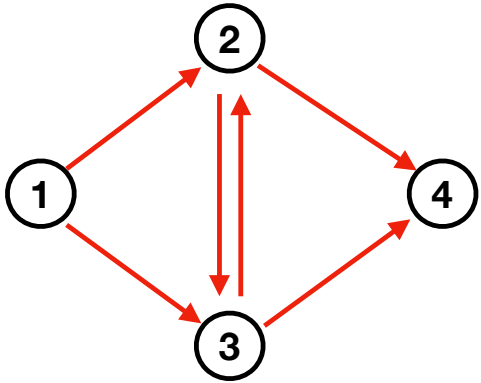
  $$(1, 3), (3, 2), (2, 4) \in E$$

  Rückwärtskante:

  $$(2, 3) \in E$$

- $W = (4, 2, 1)$
  Vorwärtskanten: keine
  Rückwärtskanten:

  $$(4, 2), (2, 1) \in E$$



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

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

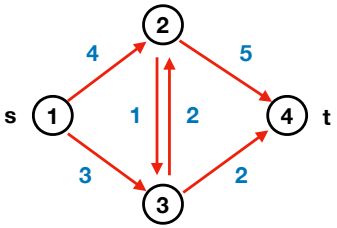
ist der Wert, um den sich der Fluss auf dem flusserhöhenden Weg W erhöhen lässt.

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:



$W = (1, 3, 4)$, nur Vorwärtskanten : $x_{13} = 3 < 5, x_{34} = 2 < 5$

$$\delta_W = \min\{c_{13} - x_{13}, c_{34} - x_{34}\} = \min\{2, 3\} = 2.$$

$W = (1, 2, 3, 4)$ mit Vorwärtskante $(2, 3)$: $x_{12} = 4 < 5, x_{23} = 1 < 5, x_{34} = 2 < 5,$

$$\delta_W = \min\{1, 4, 3\} = 1.$$

$W = (1, 2, 3, 4)$ mit Rückwärtskante $(3, 2)$: $x_{12} = 4 < 5, x_{32} = 2 > 0, x_{34} = 2 < 5,$

$$\delta_W = \min\left(\{1, 3\} \cup \{2\}\right) = 1.$$

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.

Beweis.

Siehe Übung. □

Flusserhöhende Wege und maximaler Fluss

Lemma

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

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

Beweis.

„⇒“: Existiert ein flusserhöhender Weg, dann ist der Fluss nicht maximal (Lemma Seite 30).
„⇐“: Sei $S := \{j \in V : \text{es existiert flusserhöhender Weg von } s \text{ nach } j\}$
$\Rightarrow s \in S, t \notin S$ da kein flusserhaltender Weg von s nach t existiert.
Setze $T := V \setminus S$. Dann gilt $f(S, T) = c(S, T)$ und $f(T, S) = \ell(T, S)$. Mit Lemma Seite 22:

$$v(f) = f(S, T) - f(T, S) = c(S, T) - \ell(T, S) = \text{Kapazität des Schnittes}$$

Für jeden beliebigen Fluss und Schnitt gilt: $v(f) \leq c(S, T) - \ell(T, S)$, also ist f maximal. □

Satz von Ford-Fulkerson

Satz (Max-Flow-Min-Cut-Theorem, Ford und Fulkerson)

Falls ein zulässiger Fluss existiert, so gilt:

$$\max_f v(f) = \min\left\{c(S,T) - \ell(T,S) : (S,T) \text{ ist } (s\text{-}t)\text{-Schnitt}\right\}.$$

Also: In einem Netzwerk ist der maximale Flusswert eines (s-t)-Flusses gleich der minimalen Kapazität aller (s-t)-Schnitte.

Beweis.

Das Lemma auf Seite 22 sagte aus: Für jeden zulässigen Fluss f gilt

$$v(f) \leq \min\left\{c(S,T) - \ell(T,S) : (S,T) \text{ ist } (s\text{-}t)\text{-Schnitt}\right\}.$$

Mit dem Lemma auf S. 31 folgt, dass das Minimum angenommen wird, also Gleichheit gilt. □

Inhalt

Algorithmus von Ford-Fulkerson

Idee: Starte mit einem (s-t)-Fluss und erhöhe entlang flusserhöhender (s-t)-Wege solange, bis kein flusserhöhender (s-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-t)-Weg W.
   - Bestimme δ_W.
   - Update f (wie in Lemma Seite 30) 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-t)-Weg gibt: Breche ab.

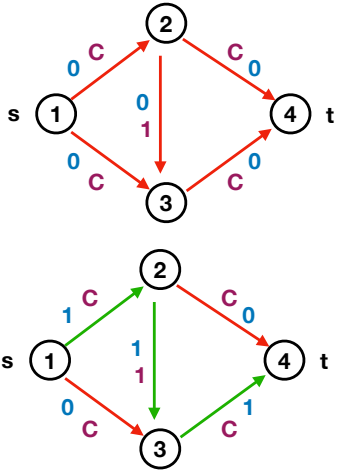
Algorithmus von Ford-Fulkerson: Beispiel (1)

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:

- Von $s = 1$ zu $t = 4$ existieren flusserhöhende Wege
- $\rightsquigarrow$ wähle z. B. $(s\text{-}t)$-Weg $W = (1, 2, 3, 4)$
- $\delta_W = 1$ wegen $c_{23} = 1$
- $f = (x_{12}, x_{13}, x_{23}, x_{24}, x_{34}) = (1, 0, 1, 0, 1)$
- $v(f) = 1$



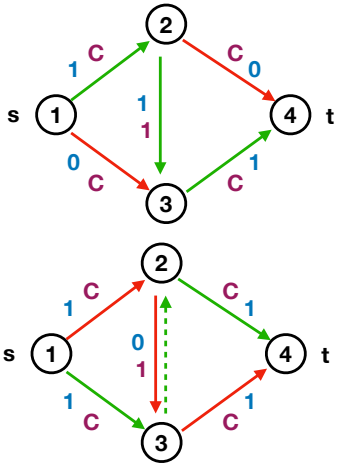
Algorithmus von Ford-Fulkerson: Beispiel (2)

2. Iteration:

- Von $s = 1$ zu $t = 4$ existieren flusserhöhende Wege
- $\rightsquigarrow$ wähle $W = (1, 3, 2, 4)$ mit Rückwärtskante (2,3),
- $\delta_W = 1$ wegen $\ell_{23} = 0$, Erinnerung:

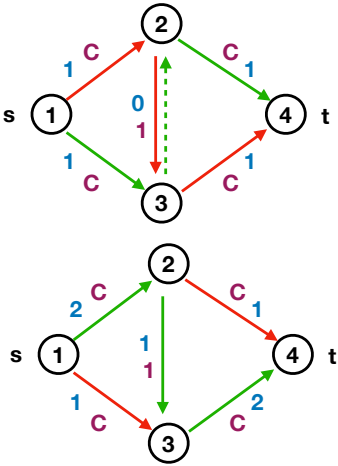
$$\delta_W := \min\big(\{c_{ij} - x_{ij} : (i,j) \text{ ist Vorwärtskante auf } W\} \cup \{x_{ij} - \ell_{ij} : (i,j) \text{ ist Rückwärtskante auf } W\}\big)$$

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



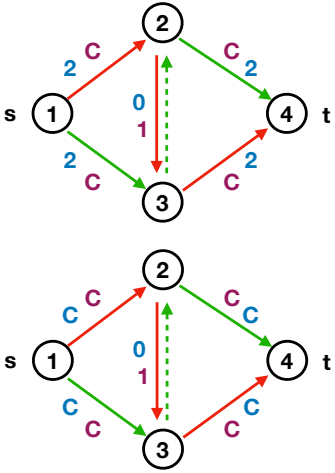
Algorithmus von Ford-Fulkerson: Beispiel (3)

3. Iteration:
   - Von $s = 1$ zu $t = 4$ existieren flusserhöhende Wege
   - $\rightsquigarrow$ 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 (4)

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.



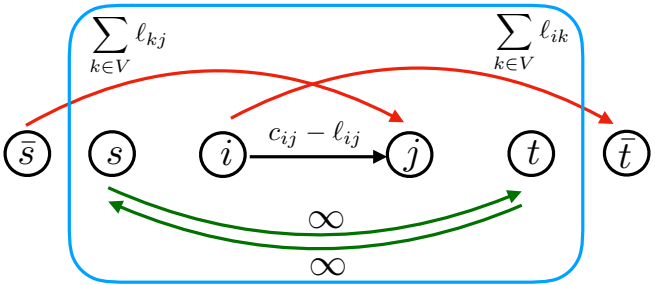
Inhalt

Bestimmung eines zulässigen Ausgangsflusses

Idee: Reduktion auf Bestimmung eines maximalen Flusses in Netzwerk mit $\ell_{ij} = 0$.
In diesem Netzwerk ist $\bar{f} \equiv 0$ ein zulässiger Fluss.
Sei (V, E, s, t, ℓ, c) gegeben. Konstruiere nun $(\bar{V}, \bar{E}, \bar{s}, \bar{t}, \bar{\ell} = 0, \bar{c})$ wie folgt:

- Führe künstliche Knoten $\bar{s}, \bar{t}$ ein: $\bar{V} := V \cup \{\bar{s}, \bar{t}\}$.
- Führe zusätzliche Kanten ein: $\bar{E} := E \cup \{(\bar{s}, j), (j, \bar{t}) : j \in V\} \cup \{(s, t), (t, s)\}$.
- Setze obere Kapazitäten:

$$\bar{c}_{ij} := \begin{cases} c_{ij} - \ell_{ij}, & (i,j) \in E, \\ \sum_{k \in V} \ell_{kj}, & i = \bar{s}, \\ \sum_{k \in V} \ell_{ik}, & j = \bar{t}, \\ \infty, & (i,j) = (s,t), \\ \infty, & (i,j) = (t,s). \end{cases}$$



Konstruktion eines zulässigen Flusses

Satz

Ein zulässiger Fluss in (V, E, s, t, ℓ, c) *existiert genau dann, wenn der maximale Fluss in* $(\bar{V}, \bar{E}, \bar{s}, \bar{t}, 0, \bar{c})$ *alle Kanten* $(\bar{s}, j), j \in V$, *sättigt.*
Ist $\bar{f}$ *ein maximaler Fluss in* $(\bar{V}, \bar{E}, \bar{s}, \bar{t}, \bar{\ell} = 0, \bar{c})$, *so ist* f, *definiert durch*

$$x_{ij} := \ell_{ij} + \bar{x}_{ij}, \quad (i,j) \in E,$$

ein zulässiger Fluss in (V, E, s, t, ℓ, c).

Beweis.

Wir beweisen nur die für den Algorithmus wichtige Richtung „$\Leftarrow$“.
Also Annahme: $\bar{f}$ sättige alle Kanten $(\bar{s}, j), j \in V$. Wir zeigen:
1. (Hilfsbehauptung): $\bar{f}$ sättigt auch alle Kanten $(j, \bar{t}), j \in V$.
2. Der konstruierte Fluss ist zulässig für (V, E, s, t, ℓ, c) und erfüllt die Flussbedingung. □

Beweis (1) Hilfsbehauptung: $\bar{f}$ sättigt auch alle Kanten $(j, \bar{t}), j \in V$

Es gilt:

$$\begin{aligned}
\sum_{j \in V} \bar{x}_{j\bar{t}} &= \sum_{j \in V} \bar{x}_{\bar{s}j} && \text{(Flussbedingung für } \bar{f}) \\
&= \sum_{j \in V} \bar{c}_{\bar{s}j} && (\bar{f} \text{ sättigt alle Kanten } (\bar{s}, j), j \in V) \\
&= \sum_{j,k \in V} \ell_{kj} && \text{(Definition } \bar{c}_{\bar{s}j}) \\
&= \sum_{j,k \in V} \ell_{jk} = \sum_{j \in V} \bar{c}_{j\bar{t}} && \text{(Definition } \bar{c}_{j\bar{t}})
\end{aligned}$$

Also sättigt $\bar{f}$ auch alle Kanten $(j, \bar{t}), j \in V$.

Beweis (2): Der durch $x_{ij} := \ell_{ij} + \bar{x}_{ij}$ konstruierte Fluss ist zulässig

Da $\bar{f}$ zulässig für $(\bar{V}, \bar{E}, \bar{s}, \bar{t}, \bar{\ell} = 0, \bar{c})$, gilt:

$$0 \leq \bar{x}_{ij} \leq \bar{c}_{ij} = c_{ij} - \ell_{ij} \quad \text{für alle } (i,j) \in E,$$

also

$$\ell_{ij} \leq \bar{x}_{ij} + \ell_{ij} =: x_{ij} \leq c_{ij} \quad \text{für alle } (i,j) \in E,$$

d. h. f ist zulässig für (V, E, s, t, ℓ, c).

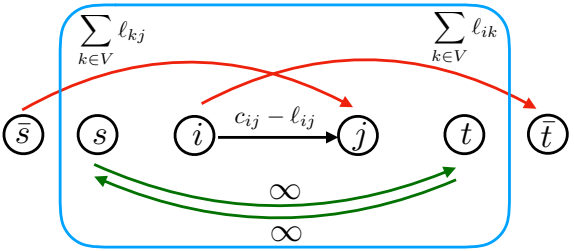
Beweis (3): Der durch $x_{ij} := \ell_{ij} + \bar{x}_{ij}$ konstruierte Fluss erfüllt die Flussbedingung

Für alle $j \in V \setminus \{s, t\}$ gilt für $\bar{f}$:

$$\underbrace{\bar{x}_{\bar{s}j} + \sum_{k \in V} \bar{x}_{kj}}_{\bar{f} \text{ sättigt } (\bar{s},j)} = \underbrace{\bar{x}_{j\bar{t}} + \sum_{k \in V} \bar{x}_{jk}}_{\bar{f} \text{ sättigt } (j,\bar{t})}$$

Einsetzen:

$$\sum_{k \in V} \ell_{kj} + \sum_{k \in V} \bar{x}_{kj} = \sum_{k \in V} \ell_{jk} + \sum_{k \in V} \bar{x}_{jk}$$



Nach Definition von x_{ij}:

$$\sum_{k \in V} x_{kj} = \sum_{k \in V} x_{jk}$$

□

Lernziele

- Den Begriff von Flüssen in Netzwerken erklären können.
- Den Begriff der Kapazitäten und der Zulässigkeit in Netzwerken erklären können.
- Erklären können, was das Max-Flow-Problem ist.
- Den Algorithmus von Ford-Fulkerson motivieren, ...
- ... anwenden ...
- ... und implementieren können.