25. Flüsse in Netzen

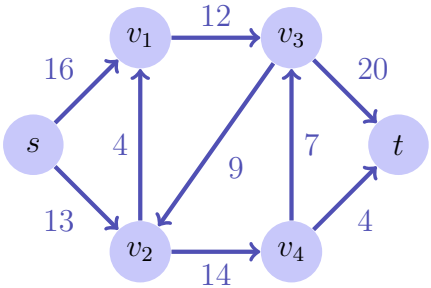
Flussnetzwerk, Maximaler Fluss, Schnitt, Restnetzwerk, Max-flow Min-cut Theorem, Ford-Fulkerson Methode, Edmonds-Karp Algorithmus, Maximales Bipartites Matching [Ottman/Widmayer, Kap. 9.7, 9.8.1], [Cormen et al, Kap. 26.1-26.3]

Motivation

Modelliere Fluss von Flüssigkeiten, Bauteile auf Fliessbändern, Strom in elektrischen Netwerken oder Information in Kommunikationsnetzwerken.

Flussnetzwerk

- *Flussnetzwerk* $G = (V, E, c)$: gerichteter Graph mit *Kapazitäten*
- Antiparallele Kanten verboten: $(u, v) \in E \Rightarrow (v, u) \notin E$.
- Fehlen einer Kante (u, v) auch modelliert durch $c(u, v) = 0$.
- *Quelle* s und *Senke* t: spezielle Knoten. Jeder Knoten v liegt auf einem Pfad zwischen s und t : $s \rightsquigarrow v \rightsquigarrow t$

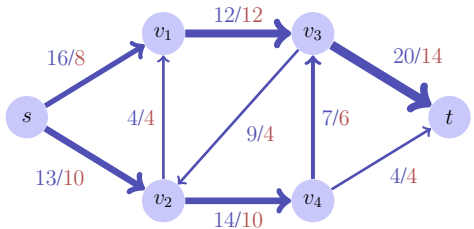


Fluss

Ein *Fluss* $f : V \times V \to \mathbb{R}$ erfüllt folgende Bedingungen:

- *Kapazitätsbeschränkung*:
  Für alle $u, v \in V$:
  $0 \leq f(u, v) \leq c(u, v)$.
- *Flusserhaltung*:
  Für alle $u \in V \setminus \{s, t\}$:

$$\sum_{v \in V} f(v, u) - \sum_{v \in V} f(u, v) = 0.$$



Wert w des Flusses:
$w(f) = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s)$.
Hier $w(f) = 18$.

Wie gross kann ein Fluss sein?

Begrenzende Faktoren: Schnitte

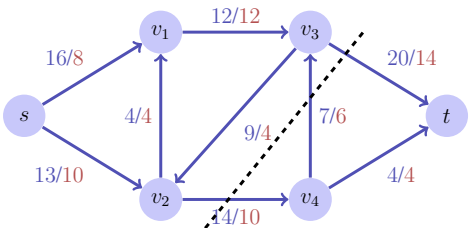
- *s von t trennender Schnitt*: Partitionierung von V in S und T mit $s \in S, t \in T$.
- *Kapazität* eines Schnittes: $c(S,T) = \sum_{v \in S, v' \in T} c(v,v')$
- *Minimaler Schnitt*: Schnitt mit minimaler Kapazität.
- *Fluss über Schnitt*:
  $f(S,T) = \sum_{v \in S, v' \in T} f(v,v') - \sum_{v \in S, v' \in T} f(v',v)$

Wie gross kann ein Fluss sein?

Es gilt für jeden Fluss und jeden Schnitt, dass $f(S,T) = w(f)$:

$$
\begin{aligned}
f(S,T) &= \sum_{v \in S, v' \in T} f(v,v') - \sum_{v \in S, v' \in T} f(v',v) \\
&= \sum_{v \in S, v' \in V} f(v,v') - \sum_{v \in S, v' \in S} f(v,v') - \sum_{v \in S, v' \in V} f(v',v) + \sum_{v \in S, v' \in S} f(v',v) \\
&= \sum_{v' \in V} f(s,v') - \sum_{v' \in V} f(v',s)
\end{aligned}
$$

Zweite Gleichheit: Ergänzung, letzte Gleichheit: Flusserhaltung.

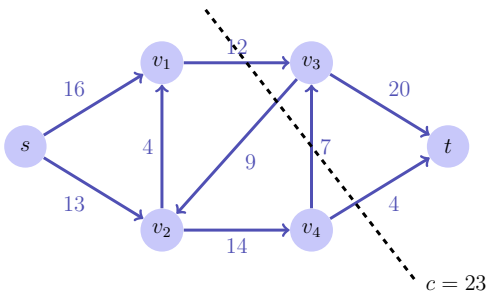


Maximaler Fluss ?

Es gilt insbesondere für alle Schnitte (S, T) von V.

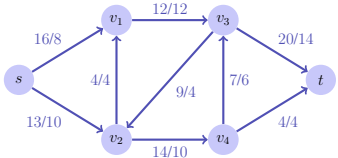
$$f(S,T) \leq \sum_{v \in S, v' \in T} c(v, v') = c(S,T)$$

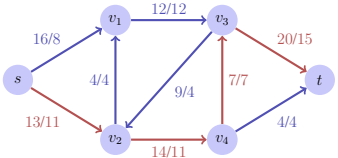
Werden sehen, dass Gleicheit gilt für $\min_{S,T} c(S,T)$.

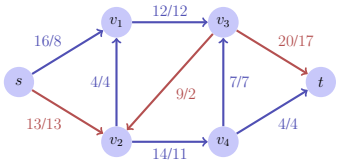


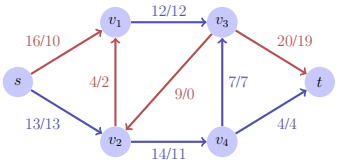
Maximaler Fluss ?

Naives Vorgehen:









Folgerung: Greedy Flusserhöhung löst das Problem nicht.

Die Ford-Fulkerson Methode

- Starte mit $f(u, v) = 0$ für alle $u, v \in V$
- Bestimme Restnetzwerk* G_f und Erweiterungspfad in G_f
- Erhöhe Fluss über den Erweiterungspfad*
- Wiederholung bis kein Erweiterungspfad mehr vorhanden.

*Wird nun erklärt

Flusserhöhung, negativ

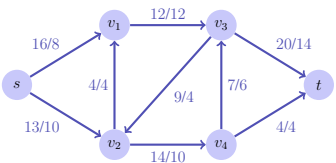
Sei ein Fluss f im Netzwerk gegeben.

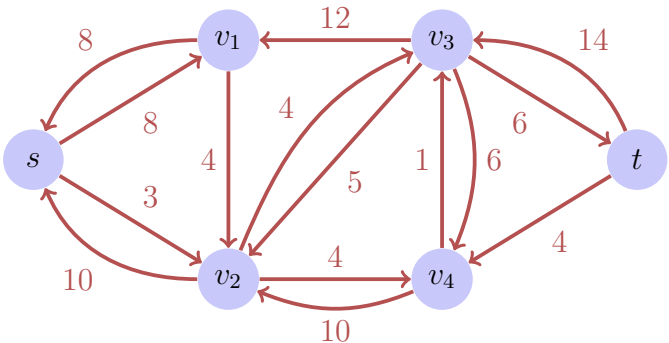
Erkenntnis:

- Flusserhöhung in Richtung einer Kante möglich, wenn Fluss entlang der Kante erhöht werden kann, also wenn $f(u, v) < c(u, v)$.
  Restkapazität $c_f(u, v) = c(u, v) - f(u, v)$.
- Flusserhöhung *entgegen* der Kantenrichtung möglich, wenn Fluss entlang der Kante verringert werden kann, also wenn $f(u, v) > 0$.
  Restkapazität $c_f(v, u) = f(u, v)$.

Restnetzwerk

Restnetzwerk G_f gegeben durch alle Kanten mit Restkapazität:





Restnetzwerke haben dieselben Eigenschaften wie Flussnetzwerke, ausser dass antiparallele Kanten zugelassen sind.

Beobachtung

Theorem

Sei $G = (V, E, c)$ ein Flussnetzwerk mit Quelle s und Senke t und f ein Fluss in G. Sei G_f das dazugehörige Restnetzwerk und sei f' ein Fluss in G_f. Dann definiert $f \oplus f'$ einen Fluss in G mit Wert $w(f) + w(f')$.

$$(f \oplus f')(u, v) = \begin{cases} f(u, v) + f'(u, v) - f'(v, u) & (u, v) \in E \\ 0 & (u, v) \notin E. \end{cases}$$

Beweis

Kapazitätsbeschränkung:

$$\begin{aligned}(f \oplus f')(u,v) &= f(u,v) + f'(u,v) - f'(v,u) \\ &\geq f(u,v) + f'(u,v) - f(u,v) = f'(u,v) \geq 0\end{aligned}$$

$$\begin{aligned}(f \oplus f')(u,v) &= f(u,v) + f'(u,v) - f'(v,u) \\ &\leq f(u,v) + f'(u,v) \\ &\leq f(u,v) + c_f(u,v) \\ &= f(u,v) + c(u,v) - f(u,v) = c(u,v).\end{aligned}$$

Beweis

Flusserhaltung

$$\begin{aligned}\sum_{u\in V}(f\oplus f')(u,v) &= \sum_{u\in V} f(u,v) + \sum_{u\in V} f'(u,v) - \sum_{u\in V} f'(v,u)\\ &\overset{\text{(Flusserhaltung von } f \text{ und } f')}{=} \sum_{u\in V} f(v,u) + \sum_{u\in V} f'(v,u) - \sum_{u\in V} f'(u,v)\\ &= \sum_{u\in V}(f\oplus f')(v,u)\end{aligned}$$

Beweis

Wert von $f \oplus f'$ (im Folgenden $N^+ := N^+(s)$, $N^- := N^-(s)$):

$$
\begin{aligned}
w(f \oplus f') &= \sum_{v \in N^+} (f \oplus f')(s,v) - \sum_{v \in N^-} (f \oplus f')(v,s) \\
&= \sum_{v \in N^+} f(s,v) + f'(s,v) - f'(v,s) - \sum_{v \in N^-} f(v,s) + f'(v,s) - f'(s,v) \\
&= \sum_{v \in N^+} f(s,v) - \sum_{v \in N^-} f(v,s) + \sum_{v \in N^+ \cup N^-} f'(s,v) + \sum_{v \in N^+ \cup N^-} f'(v,s) \\
&= \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s) + \sum_{v \in V} f'(s,v) + \sum_{v \in V} f'(v,s) \\
&= w(f) + w(f').
\end{aligned}
$$

■

Fluss in G_f

Erweiterungspfad p: Pfad von s nach t im Restnetzwerk G_f.

Restkapazität $c_f(p) = \min\{c_f(u,v) : (u,v) \text{ Kante in } p\}$

Theorem

Die Funktion $f_p : V \times V \to \mathbb{R}$*,*

$$f_p(u,v) = \begin{cases} c_f(p) & \text{wenn } (u,v) \text{ Kante in } p \\ 0 & \text{sonst} \end{cases}$$

ist ein Fluss in G_f *mit dem Wert* $w(f_p) = c_f(p) > 0$*.*

[Beweis: Übung]

Folgerung

Strategie für den Algorithmus:

Mit einem Erweiterungspfad p in G_f definiert $f \oplus f_p$ einen neuen Fluss mit Wert $w(f \oplus f_p) = w(f) + w(f_p) > w(f)$

Max-Flow Min-Cut Theorem

Theorem

Wenn f ein Fluss in einem Flussnetzwerk $G = (V, E, c)$ mit Quelle s und Senke t is, dann sind folgende Aussagen äquivalent:

1. *f ist ein maximaler Fluss in G*
2. *Das Restnetzwerk G_f enthält keine Erweiterungspfade*
3. *Es gilt $w(f) = c(S, T)$ für einen Schnitt (S, T) von G.*

Beweis

- $(3) \Rightarrow (1)$:
  Es gilt $w(f) \leq c(S,T)$ für alle Schnitte S, T. Aus $w(f) = c(S,T)$ folgt also $w(f)$ maximal.
- $(1) \Rightarrow (2)$:
  f maximaler Fluss in G. Annahme: G_f habe einen Erweiterungsfad. Dann gilt $w(f \oplus f_p) = w(f) + w(f_p) > w(f)$. Widerspruch.

Beweis $(2) \Rightarrow (3)$

Annahme: G_f habe keinen Erweiterungsfad. Definiere $S = \{v \in V :$ es existiert Pfad $s \rightsquigarrow v$ in $G_f\}$. $(S, T) := (S, V \setminus S)$ ist ein Schnitt: $s \in S, t \notin S$. Sei $u \in S$ und $v \in T$.

- Wenn $(u, v) \in E$, dann $f(u, v) = c(u, v)$, sonst wäre $(u, v) \in E_f$.
- Wenn $(v, u) \in E$, dann $f(v, u) = 0$, sonst wäre $c_f(u, v) = f(v, u) > 0$ und $(u, v) \in E_f$
- Wenn $(u, v) \notin E$ und $(v, u) \notin E$, dann $f(u, v) = f(v, u) = 0$.

Also

$$\begin{aligned} w(f) = f(S, T) &= \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{v \in T} \sum_{u \in s} f(v, u) \\ &= \sum_{u \in S} \sum_{v \in T} c(u, v) - \sum_{v \in T} \sum_{u \in s} 0 = \sum_{u \in S} \sum_{v \in T} c(u, v) = c(S, T). \end{aligned}$$

Algorithmus Ford-Fulkerson(G, s, t)

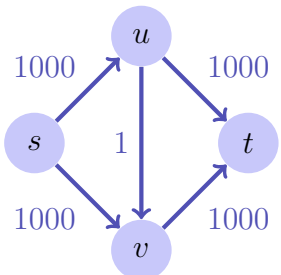
Input : Flussnetzwerk $G = (V, E, c)$
Output : Maximaler Fluss f.

for $(u, v) \in E$ **do**
    $f(u, v) \leftarrow 0$

while Existiert Pfad $p : s \rightsquigarrow t$ im Restnetzwerk G_f **do**
    $c_f(p) \leftarrow \min\{c_f(u, v) : (u, v) \in p\}$
    foreach $(u, v) \in p$ **do**
        if $(u, v) \in E$ **then**
            $f(u, v) \leftarrow f(u, v) + c_f(p)$
        else
            $f(v, u) \leftarrow f(u, v) - c_f(p)$

Analyse

- Der Ford-Fulkerson Algorithmus muss für irrationale Kapazitäten nicht einmal terminieren! Für ganze oder rationale Zahlen terminiert der Algorithmus.
- Für ganzzahligen Fluss benötigt der Algorithmus maximal $w(f_{\max})$ Durchläufe der While-Schleife. Suche einzelner zunehmender Weg (z.B. mit Tiefensuche oder Breitensuche $\mathcal{O}(|E|)$). Also $\mathcal{O}(f_{\max}|E|)$.



Bei schlecht gewählter Strategie benötigt der Algorithmus hier bis zu 2000 Iterationen.

Edmonds-Karp Algorithmus

Wähle in der Ford-Fulkerson-Methode zum Finden eines Pfades in G_f jeweils einen Erweiterungspfad kürzester Länge (z.B. durch Breitensuche).

Edmonds-Karp Algorithmus

Theorem

Wenn der Edmonds-Karp Algorithmus auf ein ganzzahliges Flussnetzwerk $G = (V, E)$ mit Quelle s und Senke t angewendet wird, dann ist die Gesamtanzahl der durch den Algorithmus angewendete Flusserhöhungen in $\mathcal{O}(|V| \cdot |E|)$

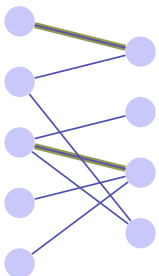
[Ohne Beweis]

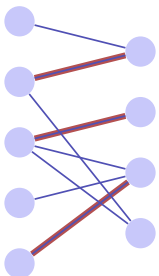
Anwendung: Maximales bipartites Matching

Gegeben: bipartiter ungerichteter Graph $G = (V, E)$.

Matching M: $M \subseteq E$ so dass $|\{m \in M : v \in m\}| \leq 1$ für alle $v \in V$.

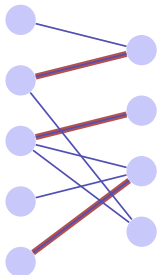
Maximales Matching M: Matching M, so dass $|M| \geq |M'|$ für jedes Matching M'.

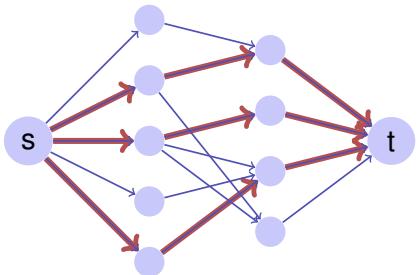




Korrespondierendes Flussnetzwerk

Konstruiere zur einer Partition L, R eines bipartiten Graphen ein korrespondierendes Flussnetzwerk mit Quelle s und Senke t, mit gerichteten Kanten von s nach L, von L nach R und von R nach t. Jede Kante bekommt Kapazität 1.





Ganzzahligkeitstheorem

Theorem

Wenn die Kapazitäten eines Flussnetzwerks nur ganzzahlige Werte anehmen, dann hat der durch Ford-Fulkerson erzeugte maximale Fluss die Eigenschaft, dass der Wert von $f(u, v)$ für alle $u, v \in V$ eine ganze Zahl ist.

[ohne Beweis]

Folgerung: Ford Fulkerson erzeugt beim zum bipartiten Graph gehörenden Flussnetzwerk ein maximales Matching $M = \{(u, v) : f(u, v) = 1\}$.