



UNIVERSITÄT
LEIPZIG

Algorithmen und Datenstrukturen II

Vorlesung *fyr4*

Leipzig, 04.11.2026

Peter F. Stadler & Thomas Gatter

FLÜSSE UND NETZWERKE



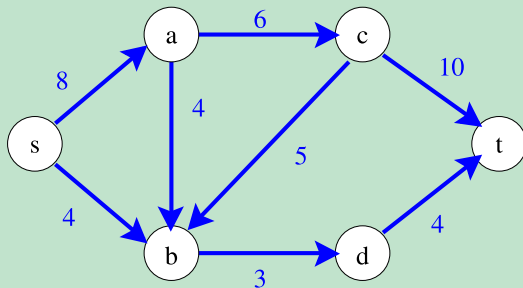
Flüsse in Netzwerken I

Anwendungsprobleme

- Wieviel Autos können durch ein Straßennetz fahren?
- Wieviel Abwasser fasst ein Kanalnetz?
- Wieviel Strom kann durch ein Leitungsnetz fließen?

Flüsse in Netzwerken II

Beispiel



Kantenwert: Kapazität $c(e)$

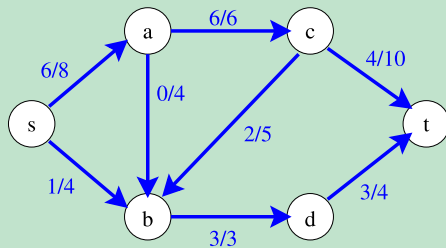
Flüsse in Netzwerken III

Definition

- Ein (*Fluss-*) *Netzwerk* ist ein gerichteter Graph $G = (V, E, c)$ mit ausgezeichneten Knoten s (Quelle, Source) und t (Senke, Drain), sowie einer Kapazitätsfunktion $c : \rightarrow \mathbb{R}^+$.
- Ein *Fluss* für das Netzwerk ist eine Funktion $f : E \rightarrow \mathbb{R}_0^+$, so dass gilt:
 - Kapazitätsbeschränkung: $\forall e \in E : 0 \leq f(e) \leq c(e)$.
 - Symmetriebedingung: $f(u, v) = -f(v, u)$
 - Flusserhaltung/Kirchhoffsches Gesetz:
$$\forall v \in V \setminus \{s, t\} : \sum_{(v', v) \in E} f(v', v) = \sum_{(v, v') \in E} f(v, v')$$
 - Der Wert von f , $w(f)$, ist die Summe der Flusswerte der die Quelle s verlassenden Kanten: $\sum_{(s, v) \in E} f(s, v)$

Flüsse in Netzwerken IV

Beispiel



Kantenbeschriftung: Fluss $f(e)$ / Kapazität $c(e)$

Flüsse in Netzwerken V

Gesucht: Fluss mit maximalem Wert

- begrenzt durch Summe der aus s wegführenden bzw. in t eingehenden Kapazitäten
- jeder weitere “Schnitt” durch den Graphen, der s und t trennt, begrenzt maximalen Fluss

Wikipedia: *Flüsse und Schnitte* ([Link](#))

Schnitte

Definition

Es sei G ein gerichteter Graph mit Knotenmenge V und Kantenmenge E . Ein **Schnitt** (A, B) in G ist eine Partition von V in zwei nichtleere Teilmengen A und B . Zu jedem Schnitt gehört eine Menge von Schnittkanten

$$E(A, B) = \{e \in E \mid e = (u, v), u \in A, v \in B\}$$

(A, B) ist ein Schnitt in G wenn $A \cap B = \emptyset$, $A \neq \emptyset$, $B \neq \emptyset$, and $A \cup B = V$.

Schnitte

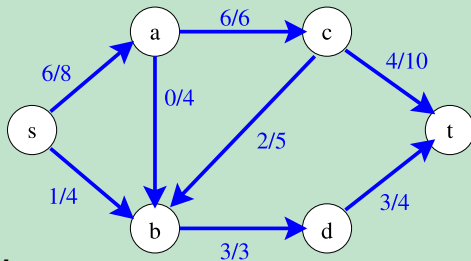
Definition

Für jeden Schnitt mit $s \in A$ und $t \in B$ sei die Kapazität

$$c(A, B) = \sum_{e \in E(A, B)} c(e)$$

Schnitte

Beispiel



In unserem Beispiel:

$$c(\{s, a, b\}, \{c, d, t\}) = c(a, c) + c(b, d) = 6 + 3 = 9$$

$$c(\{s, a\}, \{b, c, d, t\}) = c(a, c) + c(a, b) + c(s, b) = 6 + 4 + 4 = 14$$

Hinweis: es wird hier über die Kapazität summiert und nur Kanten von A nach B berücksichtigt

Restgraph

Definition

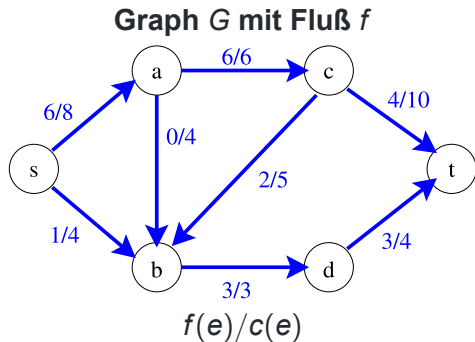
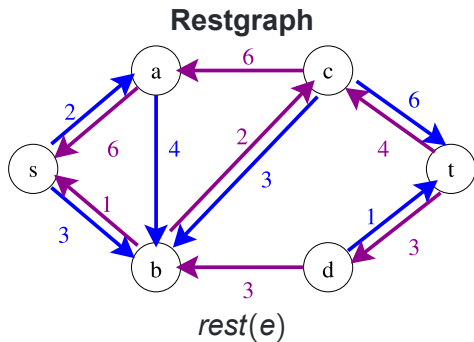
Sei f ein zulässiger Fluss für $G = (V, E, c)$:

- Für jede Kante $e = (u, v) \in E$ gibt es eine Kante $\bar{e} = (v, u)$ in die umgekehrte Richtung (Rückwärtskante).
- Die Menge der Vorwärts und Rückwärtskanten ist daher $E' = \{(v, w) \mid (v, w) \in E \vee (w, v) \in E\}$
- Wir definieren die **Restkapazität** einer Kante $e = (v, w) \in E'$ als

$$\text{rest}(e) := \begin{cases} c(e) - f(e) & \text{falls } e \in E \\ f(e) & \text{falls } \bar{e} \in E \end{cases}$$

- Der Restgraph G_f von f (bzgl. G) besteht aus den Kanten $e \in E'$ mit Kapazitäten $\text{rest}(e)$. Kanten mit $\text{rest}(e) == 0$ können ausgelassen werden.

Restgraph



Erweiterungspfad I

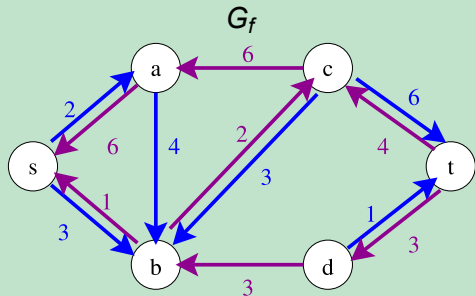
Definition

Ein **Erweiterungspfad** ist ein (zyklenfreier) Pfad p in G_f von s nach t . Die **Restkapazität** eines Erweiterungspfades p ist

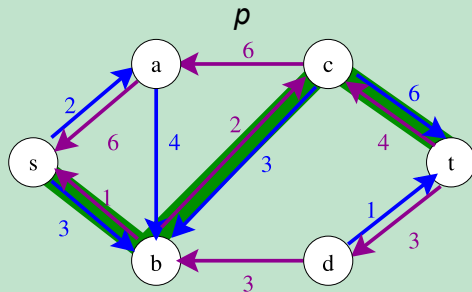
$$c_f(p) = \min\{(\text{rest}(u, v) \mid (u, v) \text{ liegt auf } p)\}$$

Erweiterungspfad II

Beispiel



$$c_f(p) = \min\{3, 2, 6\} = 2$$



Erweiterungspfad III

Beobachtung:

Sei G ein Netzwerk, c eine Kapazität und f ein Fluss. Sei G_f das zugehörige Restnetzwerk und $c_f = \text{rest}$ die zugehörige Restkapazität. Sei ferner g ein zulässiger Fluss auf G_f . Dann gilt: $(f + g)$ ist ein zulässiger Fluss auf G mit $|(f + g)| = |f| + |g|$

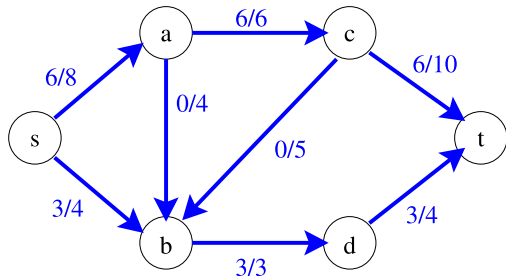
Beobachtung:

Sei p ein Erweiterungspfad in G_f . Dann ist g mit $g(e) = c_f(p)$ für alle Kanten auf p und $g(e) = 0$ ein zulässiger Fluss auf dem Restgraphen.

Erweiterungspfad IV

Idee: Optimierung des Flusses für G :

- entlang eines Erweiterungspfades p
- Verbesserung um $c_f(p)$ für alle Kanten e auf dem Weg p



Ford-Fulkerson-Algorithmus I

Optimierung eines bestehenden Flusses f in einem Netzwerk G mit Kapazität c :

Ablauf des Algorithmus

- bestimme Restnetzwerk G_f
- suche Erweiterungspfad
- bestimme $c_f(p)$
- erhöhe den Fluss um Minimum der verfügbaren Restkapazität der einzelnen Kanten des Erweiterungspfades

Ford-Fulkerson-Algorithmus II

Algorithmus

input : Restgraph G_f zu Graph G mit initialem Fluss f

output: Optimaler Fluss f

while *es gibt einen zunehmenden Weg p in Restgraph G* **do**

$r = \min \{ \text{rest}(e) \mid e \text{ liegt auf } p \}$

for *alle $e = (v, w)$ auf p* **do**

if *e in E* **then**

$f(e) = f(e) + r$

else

$f(w, v) = f(w, v) - r$

Ford-Fulkerson-Algorithmus - Laufzeit

- $O(|E|)$ für Konstruktion des Restgraphen
- $O(|E|)$ für Konstruktion eines Erweiterungspfades: Graphtraversierung und Bestimmung des Flusses entlang des Wegs
- in jedem Schritt verbessert sich der Fluss um $r = \min\{rest(e) | e \text{ liegt auf } p\}$

Für *ganzzahlige* Kapazitäten: $O(E) \times \max |f|$ (jeder Durchlauf erhöht den Fluss um mindestens eins, also gibt es bis zu $|f^*|$ Durchläufe (f^* maximaler Fluss))

⇒ Für *allgemeine* Kapazitäten: Konvergenz nicht garantiert!

Min-Cut I

Wenn ein Erweiterungspfad gefunden werden kann, bringt der neue Fluss ($f + g$) eine Verbesserung.

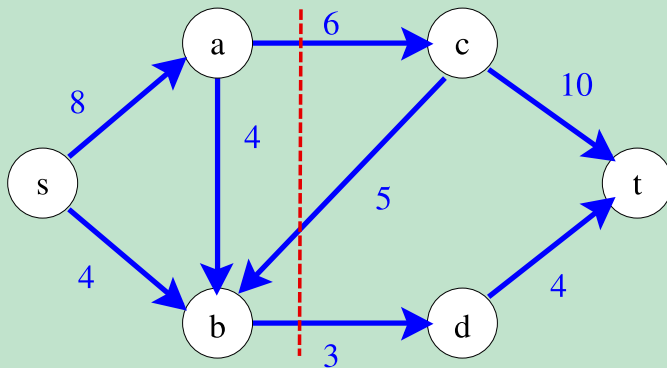


Aber kann immer ein Erweiterungspfad gefunden werden, wenn der bisherige Fluss f nicht maximal ist?

Beobachtung: Im Graph mit optimalem Fluss existiert im Restgraph kein Weg von Quelle zu Senke!

Min-Cut II

Beispiel



Min-Cut-Max-Flow-Theorem

Theorem (Min-Cut-Max-Flow-Theorem)

Sei f ein zulässiger Fluss für G . Dann sind folgende Aussagen äquivalent:

1. f ist maximaler Fluss in G .
2. Der Restgraph von f enthält keinen zunehmenden Weg.
3. $w(f) = c(A, B)$ für einen Schnitt (A, B) von G .

Min-Cut-Max-Flow-Theorem -- Beweis

Beweis I

(1) Sei (A, B) ein Schnitt mit $s \in A$ und $t \in B$. Flusserhaltung impliziert, dass $\sum_{e \in (A, B)} f(e) = w(f)$. Dies gilt für jeden Schnitt, der s und t trennt. Da $f(e) \leq c(e)$ erfüllt sein muss, gilt ausserdem $w(f) \leq c(A, B)$ für alle Schnitte die s und t trennen.

(2) ((1) $\Rightarrow$ (2)) Sei f ein maximaler Fluss. Würde der zugehörige Restgraph G_f einen zunehmenden Weg enthalten, könnte der Fluss weiter erhöht werden. Also gibt es keinen zunehmenden Weg.

Min-Cut-Max-Flow-Theorem -- Beweis

Beweis II

(3) ((2) $\Rightarrow$ (3)) Sei A die Menge der Knoten, die in G_f von s aus erreicht werden können, und sei $B = V \setminus A$. Also gilt $s \in A$ und insbesondere $t \in B$. Zudem ist (A, B) ein Schnitt, der s und t trennt.

Es gibt keinen Weg in G_f durch Kanten von $c(A, B)$. (Wenn $e = (u, v) \in (A, B)$ existieren würde, wäre ja nicht nur u von s erreichbar sondern auch v , was der Definition von B widersprechen würde.) Daher gilt für die Restkapazität entlang jeder Kante von $e \in c(A, B)$, dass $\text{rest}(e) = c(e) - f(e) = 0$ verschwindet. Also ist $f(e) = c(e)$ für alle $e \in c(A, B)$.

Wegen (1) gilt $w(f) = \sum_{e \in (A, B)} f(e) = \sum_{e \in (A, B)} c(e) = c(A, B)$.

(4)((3) $\Rightarrow$ (1)) Für alle zulässigen Flüsse f und Schnitte (A, B) , die s und t trennen, gilt nach (1) $w(f) \leq c(A, B)$. Wenn also $w(f) = c(A, B)$ ist, dann ist $w(f)$ notwendigerweise maximal. □

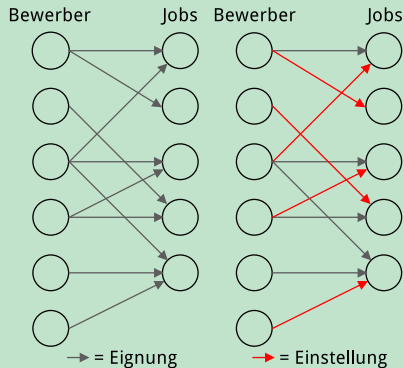
Maximales Bipartites Matching

Beispiel (leicht verallgemeinerbar):

- Eine Gruppe von M Personen bewirbt sich auf N Jobangebote
- Jede Person hat eine Prioritätenliste von präferierten Jobs
- Jeder Arbeitgeber hat eine Prioritätenliste von präferierten Kandidaten
- Wieviele Personen können maximal auf die verfügbaren Jobs verteilt werden?

Maximales Bipartites Matching - Beispiel

Beispiel



Verteilung von Bewerbern auf Jobangebote

Maximales Bipartites Matching

Definition

Ein **Matching (Zuordnung)** M für einen ungerichteten Graphen $G = (V, E)$ ist eine Teilmenge der Kanten, so dass jeder Knoten in V in höchstens einer Kante vorkommt.

- $|M|$ = Größe der Zuordnung
- Perfektes Matching: kein Knoten bleibt “allein” (unmatched), d.h. jeder Knoten ist in einer Kante von M vertreten

Matching M ist **maximal**, wenn es kein Matching M' mit $|M| < |M'|$ gibt

Maximales Bipartites Matching

Definition

Ein **bipartiter Graph** ist ein Graph, dessen Knotenmenge V in zwei disjunkte Teilmengen V_1 und V_2 aufgeteilt ist, und dessen Kanten jeweils einen Knoten aus V_1 mit einem aus V_2 verbinden (also keine Kanten innerhalb von V_1 oder V_2).

Bipartites Matching und maximaler Fluss

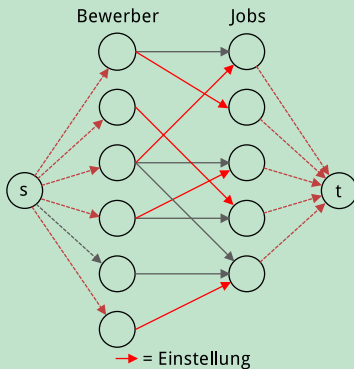
Maximales Matching kann auf maximalen Fluss zurückgeführt werden:

Algorithmus - Ablauf

- Quelle und Senke hinzufügen.
 - Kanten von V_1 nach V_2 richten.
 - Jeder Knoten in V_1 erhält eingehende Kante von der Quelle.
 - Jeder Knoten in V_2 erhält ausgehende Kante zur Senke.
 - Alle Kanten erhalten Kapazität $c(e) = 1$
- Anwendung des Ford-Fulkerson-Algorithmus

Maximales Bipartites Matching - Beispiel

Beispiel



Verteilung von Bewerbern auf Jobangebote