Datenstrukturen & Algorithmen

Peppo Brambilla
Universität Bern
Frühling 2018

Graphenalgorithmen

Maximaler Fluss

- Einleitung
- Flussnetzwerke
- Ford-Fulkerson Methode
- Maximales bipartites Matching

Einleitung

- Gerichtete Graphen zur Modellierung von Flussnetzwerken
- Modell: Material fliesst durch Röhrensystem von Quelle zu Senke
  - Quelle produziert konstantes Materialvolumen pro Zeit
  - Senke konsumiert konstantes Volumen pro Zeit
  - «Fluss» in Röhre ist Volumen pro Zeit, das verschoben wird
- Beispiele
  - Flüssigkeit durch Röhren
  - Teile auf Fliessbändern
  - Strom in elektrischen Netzwerken
  - Information in Kommunikationsnetzwerken

Flussnetzwerke

- Modellierung mit gerichteten Graphen
- Kante: «Röhre»
  - Gewisse maximale Kapazität von Einheiten pro Zeit, die fliessen können
- Knoten: Verbindungspunkte
  - Material darf sich nicht «stauen»
  - «Fluss hinein» muss gleich «Fluss hinaus» sein
- Problem des maximalen Fluss
  - Grösstmöglicher Fluss von Quelle zu Senke, so dass keine maximale Kapazität irgendeiner Kante verletzt wird

Graphenalgorithmen

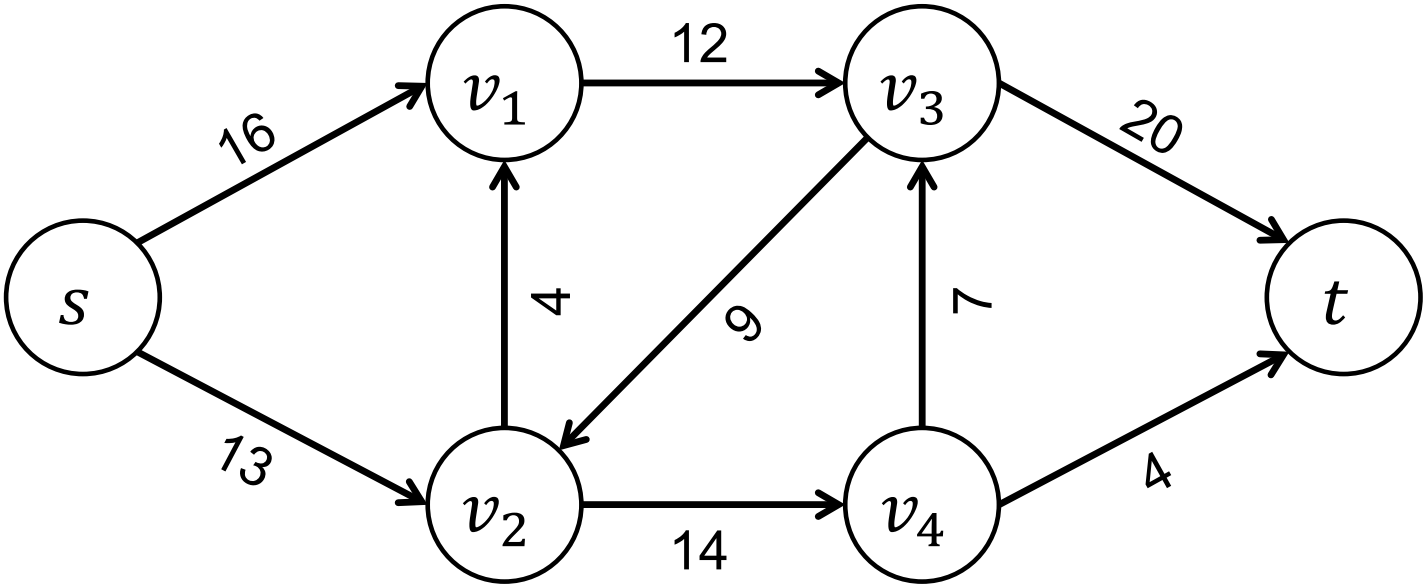
Maximaler Fluss

- Einleitung
- Flussnetzwerke
- Ford-Fulkerson Methode
- Maximales bipartites Matching

Flussnetzwerke

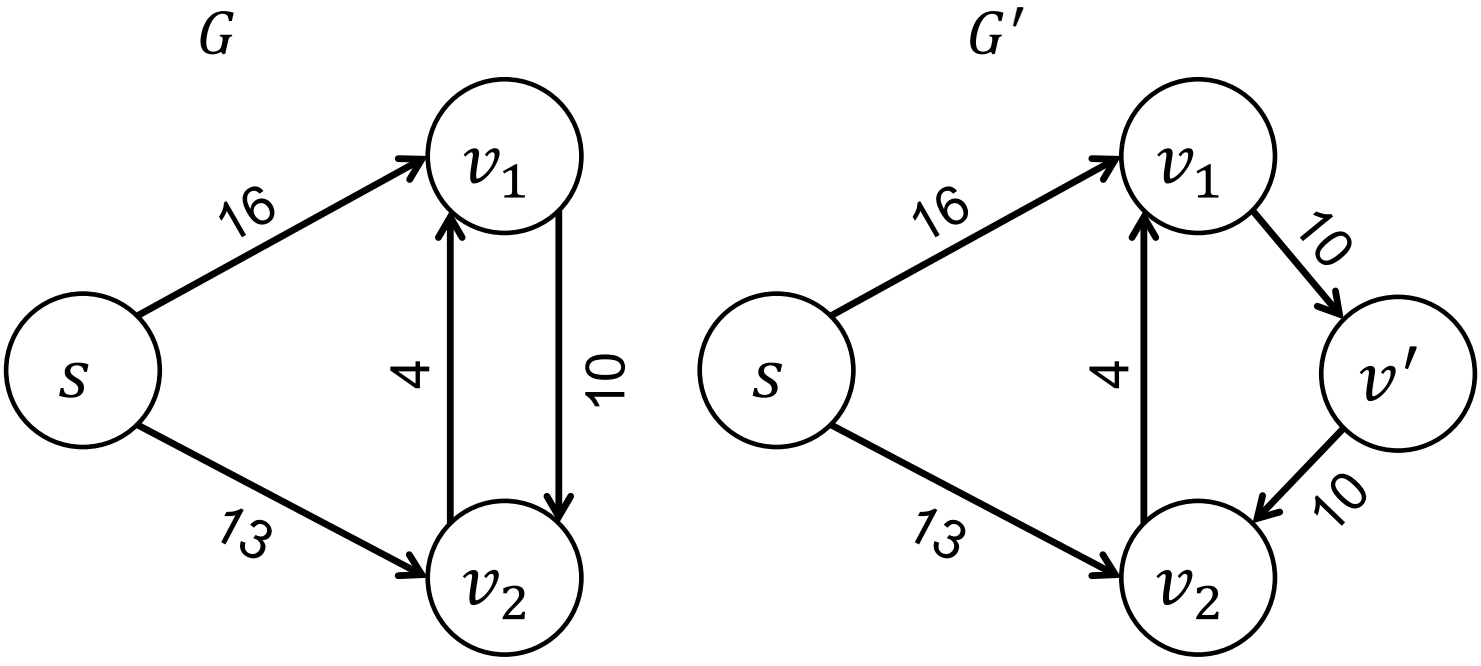
- Gerichteter Graph $G = (V, E)$
  - $(u, v) \in E \rightarrow (v, u) \notin E$
    (keine antiparallelen Kanten)
  - für alle $u \in V$: $(u, u) \notin E$ (keine Schlingen)
- Jede Kante (u, v) hat Kapazität $c(u, v) \geq 0$
  - Konvention: $(u, v) \notin E \rightarrow c(u, v) = 0$
- 2 Ausgezeichnete Knoten
  - Quelle $s \in V$
  - Senke $t \in V$
- Annahme: für jeden Knoten v gibt es einen Pfad $s \rightsquigarrow v \rightsquigarrow t$ von s über v nach t

Flussnetzwerke



Jede Kante (u, v) hat eine Kapazität $c(u, v)$
Nicht eingezeichnete Kanten haben implizit $c(u, v) = 0$

Antiparallele Kanten



Graph mit antiparallelen Kanten (G) kann durch hinzufügen von zusätzlichen Knoten und Kanten zu Flussnetzwerk (G') transformiert werden.

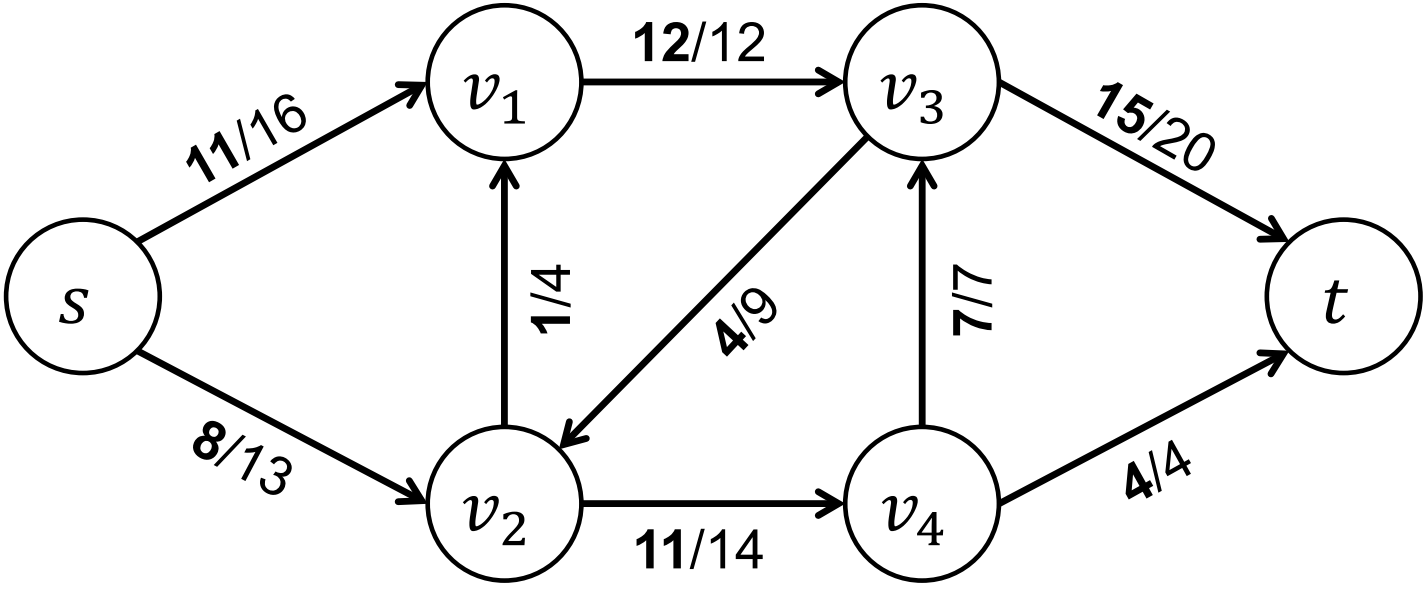
Fluss in Flussnetzwerk

- Fluss in Flussnetzwerk $G = (V, E)$: Funktion $f: V \times V \to \mathbb{R}$, die folgende Bedingungen erfüllt
  - Kapazitätsbedingung
    Für alle $(u, v) \in V \times V$: $0 \leq f(u, v) \leq c(u, v)$
    Wenn $(u, v) \notin E$, dann $f(u, v) = 0$
  - Flusserhaltung
    Für alle $u \in V - \{s, t\}$ gilt:

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

«Eingehender Fluss» = «Ausgehender Fluss»

Fluss in Flussnetzwerk



Jede Kante ist annotiert mit
Fluss/Kapazität, d.h. $\boldsymbol{f(u,v)}/c(u,v)$

Wert eines Flusses

- Wert $|f|$ des Flusses f ist definiert als

$$|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s)$$

«Fluss aus Quelle s» − «Fluss in Quelle s»

- Kann zeigen

$$|f| = \sum_{v \in V} f(v, t) - \sum_{v \in V} f(t, v)$$

- Wert des Beispiel-Flusses: 19

Graphenalgorithmen

Maximaler Fluss

- Einleitung
- Flussnetzwerke
- Ford-Fulkerson Methode
- Maximales bipartites Matching

Ford-Fulkerson Methode

- Löst das Problem des maximalen Flusses
  - Gegeben: Graph G, Quelle s, Senke t, Kapazitäten c
  - Gesucht: Fluss f mit grösstem Wert
- Basiert auf drei Ideen
  - Restnetzwerke
  - Erweiterungspfade (in Restnetzwerken)
  - Schnitte

Ford-Fulkerson Methode

- Startet mit $f(u, v) = 0$ für alle $u, v \in V$
- Iteration
  - Suche Erweiterungspfad p im Restnetzwerk G_f
  - Erhöhe Wert von f anhand von p
  - Iteriere bis $|f|$ maximal ist,
    d.h. bis kein Erweiterungspfad mehr existiert.

FORD-FULKERSON-METHOD(G, s, t)

```
1  initialize flow f to 0
2  while there exists an augmenting path p in the residual network G_f
3        augment flow f along p
4  return f
```

Restnetzwerke

- Gegeben Fluss f in Netzwerk $G = (V, E)$
- Restkapazität $c_f(u, v)$ zwischen $u, v \in V$

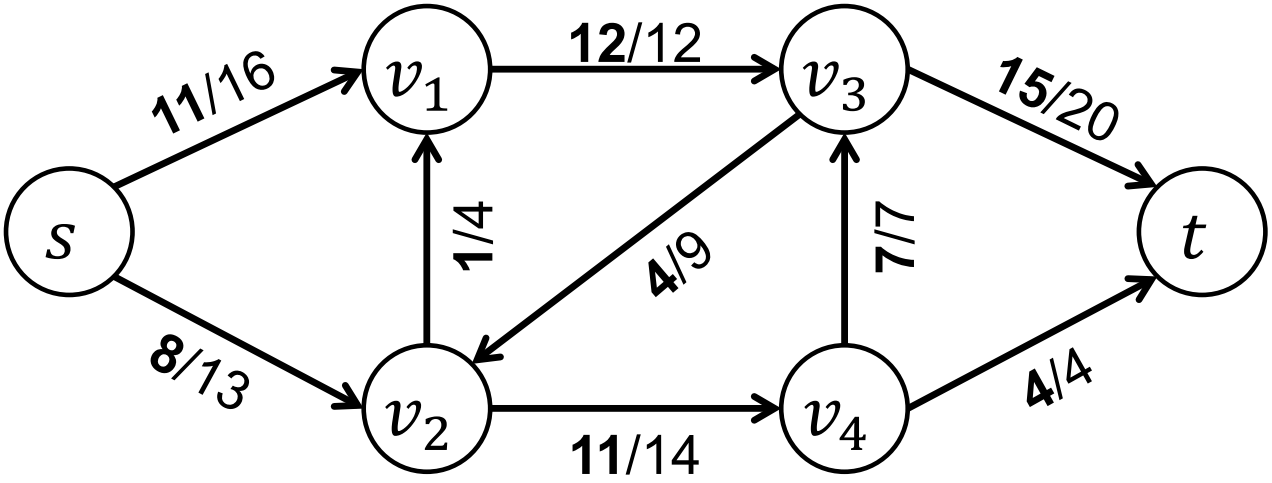
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{falls } (u,v) \in E, \\ f(v,u) & \text{falls } (v,u) \in E, \\ 0 & \text{andernfalls.} \end{cases}$$

- Restnetzwerk $G_f = (V, E_f)$

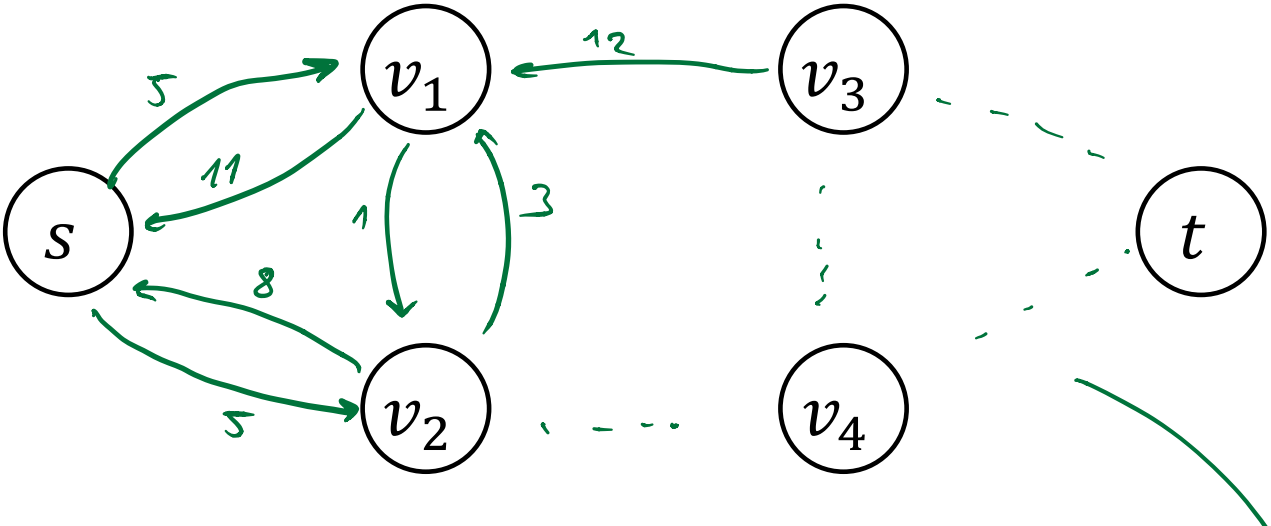
$$E_f = \{ (u,v) \in V \times V : c_f(u,v) > 0 \}$$

  - Menge der Kanten, die Restkapazität haben, d.h. wo Kapazität erhöht oder verringert werden kann.
  - Kann antiparallele Kanten haben

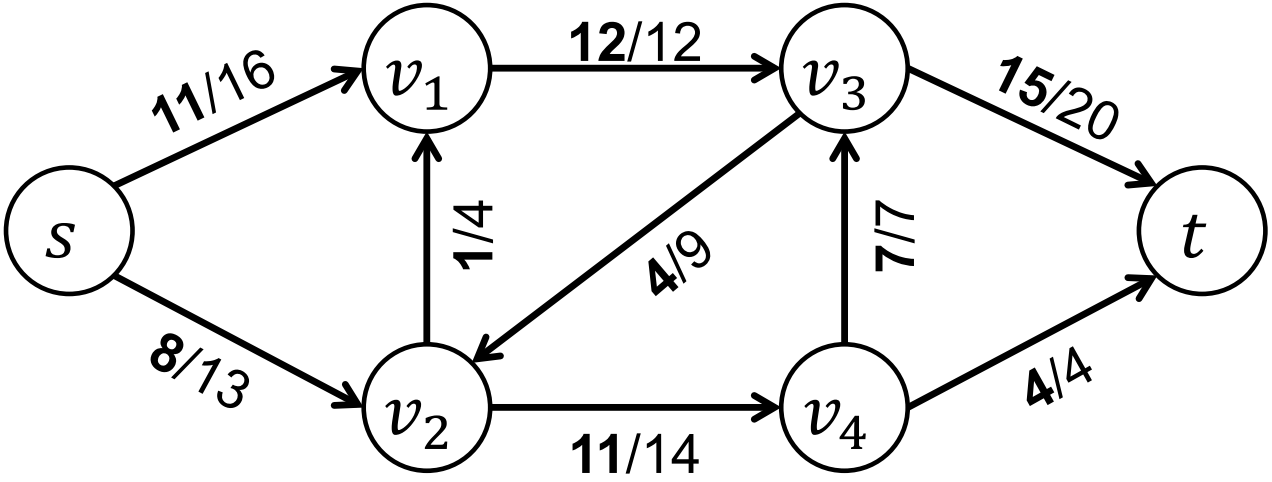
Restnetzwerke



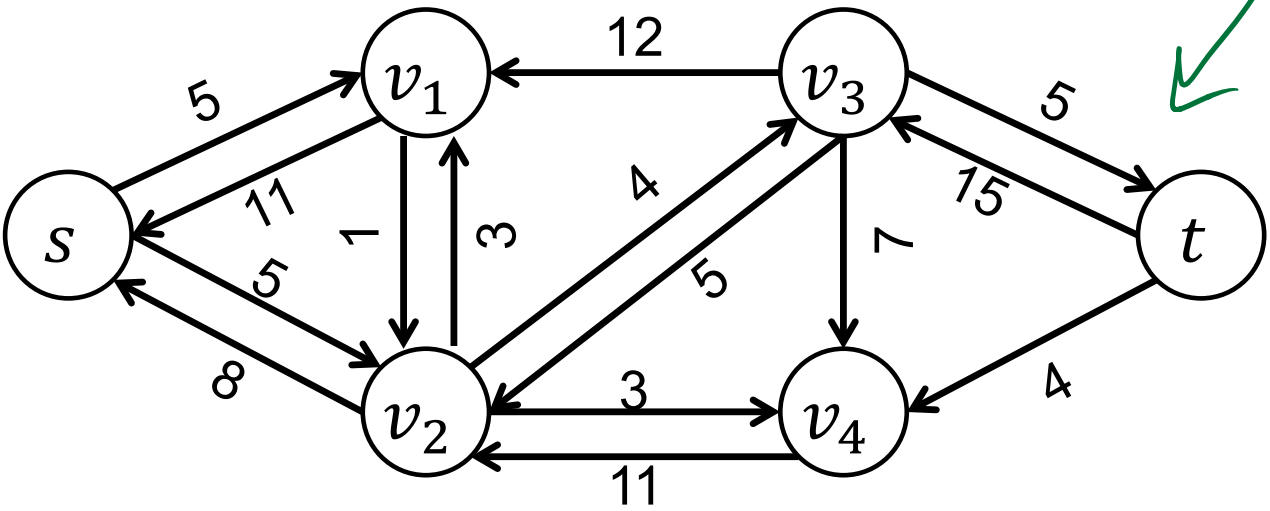
G_f



Restnetzwerke



G_f



Fluss im Restnetzwerk

- $G_f = (V, E_f)$ hat bis auf antiparallele Kanten die gleichen Eigenschaften wie Flussnetzwerk
- Können Fluss f' auf G_f definieren, der Definition eines Flusses bezüglich den Kapazitäten c_f im Netzwerk G_f genügt.
  - $f' : V \times V \to \mathbb{R}$
  - f' erfüllt Kapazitätsbedingung
  - f' erfüllt Flusserhaltung

Erhöhung eines Flusses

- Gegeben Fluss f in G und Fluss f' in G_f
- Erhöhung $f \uparrow f'$ von f um f' ist

$$(f \uparrow f')(u,v) = \begin{cases} f(u,v) + f'(u,v) - f'(v,u) & \text{falls } (u,v) \in E, \\ 0 & \text{sonst} \end{cases}$$

- Lemma: Gegeben
  - Flussnetzwerk G, Fluss f in G
  - Restnetzwerk G_f, Fluss f' in G_f
  - Dann definiert $f \uparrow f'$ einen Fluss in G mit Wert

$$|f \uparrow f'| = |f| + |f'|$$

Erweiterungspfad

- Ein einfacher Pfad $s \rightsquigarrow t$ in G_f
  - Erlaubt mehr Fluss entlang jeder Kannte
  - Eine Folge von Röhren von Quelle zu Senke, durch die mehr Material fliessen kann
- Wieviel mehr Fluss möglich entlang Pfad p?
- Restkapazität des Pfades p

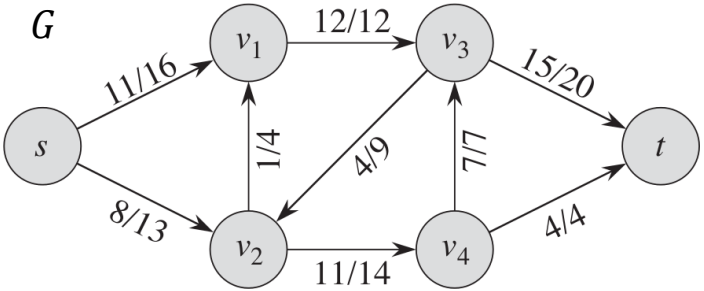
$$c_f(p) = \min\{c_f(u,v) : (u,v) \text{ ist in } p\}$$

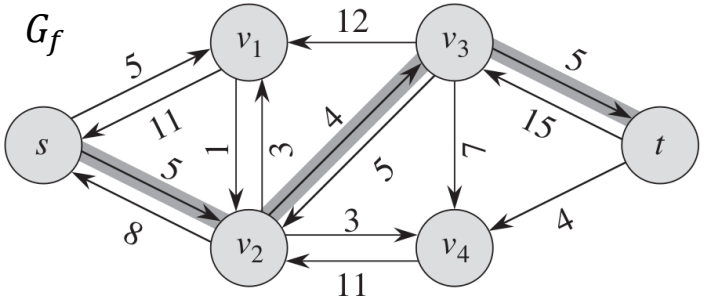
$=$ «maximaler Zusatzfluss durch Pfad p in G_f»

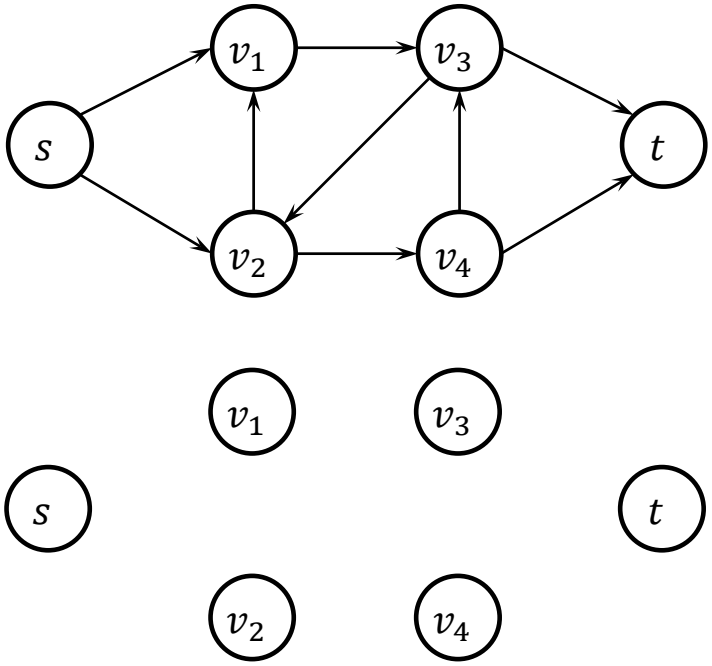
- Im Beispiel: $p = (s, v_2, v_3, t) \quad c_f(p) = 4$

Erweiterungspfad

Erhöhung des Flusses entlang $p = (s, v_2, v_3, t)$ um 4

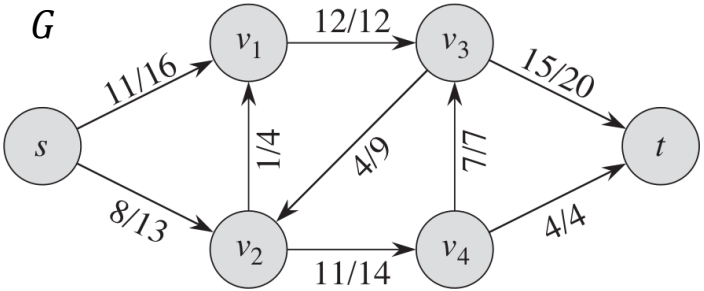


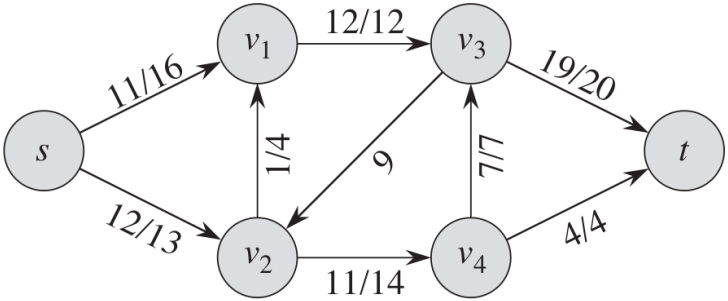


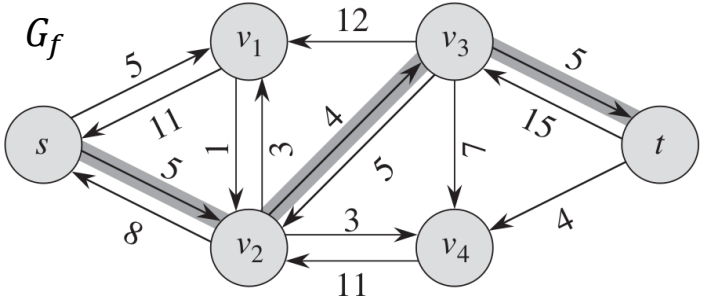


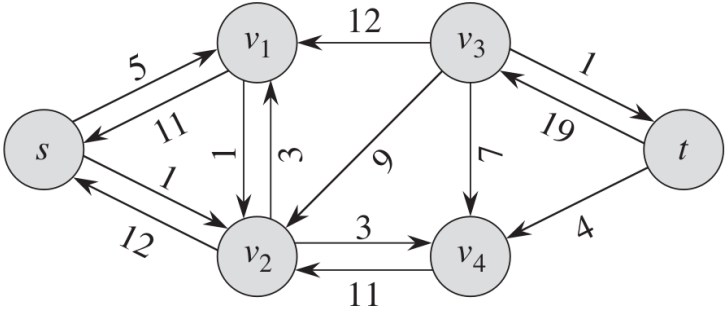
Erweiterungspfad

Erhöhung des Flusses entlang $p = (s, v_2, v_3, t)$ um 4









Kein Erweiterungspfad mehr!
Behauptung: Fluss ist maximal

Erweiterungspfad

- Lemma
  - Sei G ein Flussnetzwerk, f ein Fluss in G und p ein Erweiterungspfad in G_f
  - Definiere $f_p: V \times V \to \mathbb{R}$ mit

$$f_p(u,v) = \begin{cases} c_f(p) & \text{falls } (u,v) \text{ zu } p \text{ gehört,} \\ 0 & \text{sonst.} \end{cases}$$

  - Dann ist f_p ein Fluss in G_f mit Wert $|f_p| = c_f(p) > 0$
- Korollar
  - Definiere f_p wie oben
  - Dann ist $f \uparrow f_p$ ein Fluss in G mit Wert

$$|f \uparrow f_p| = |f| + |f_p| > |f|$$

Schnitte

- Vorgehen um maximalen Fluss zu finden:
  - Suche Erweiterungspfad p in G_f
  - Erhöhe f entlang p
  - Fertig, wenn kein Erweiterungspfad mehr in G_f existiert
- Behauptung: Fluss f ist maximal, wenn kein Erweiterungspfad in G_f existiert
- Zeigen Behauptung mithilfe von Schnitten

Schnitte

- Schnitt (S, T) eines Flussnetzwerks $G = (V, E)$ ist

  Partitionierung von V in S und $T = V - S$, so dass Quelle $s \in S$ und Senke $t \in T$

- Nettofluss $f(S, T)$ über Schnitt (S, T)

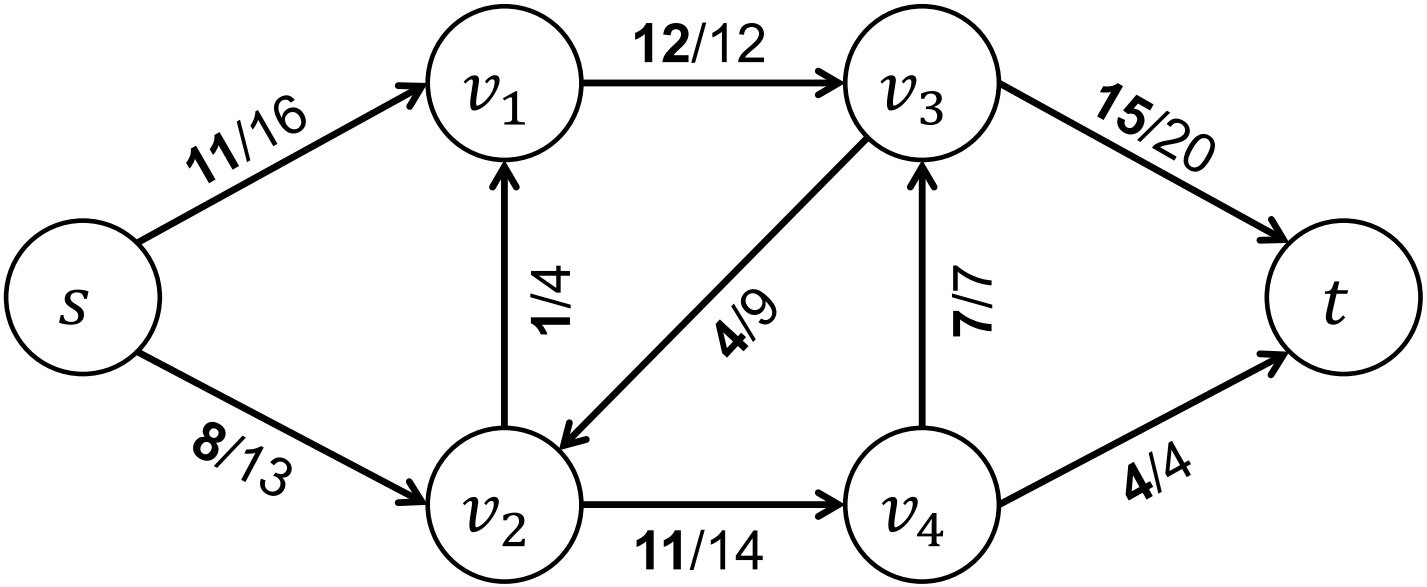
$$f(S,T) := \sum_{u \in S} \sum_{v \in T} f(u,v) - \sum_{u \in S} \sum_{v \in T} f(v,u)$$

- Kapazität $c(S, T)$ des Schnittes (S, T)

$$c(S,T) := \sum_{u \in S} \sum_{v \in \mathrm{T}} c(u,v)$$

- Minimaler Schnitt ist Schnitt mit minimaler Kapazität über alle Schnitte von G

Schnitte



Schnitte

- Beispiel: Schnitt $S = \{s, v_1, v_2\}, T = \{v_3, v_4, t\}$

$$\begin{aligned} f(S,T) &= f(v_1, v_3) + f(v_2, v_4) - f(v_3, v_2) \\ &= 12 + 11 - 4 \\ &= 19 \end{aligned}$$

$$\begin{aligned} c(S,T) &= c(v_1, v_3) + c(v_2, v_4) \\ &= 12 + 14 \\ &= 26 \end{aligned}$$

- Beachte Unterschied zwischen Nettofluss und Kapazität
  - Nettofluss: subtrahiert Fluss von T nach S
  - Kapazität: ignoriert Kapazitäten von T nach S

Schnitte

- Beispiel: Schnitt $S = \{s, v_1, v_2, v_3\}$, $T = \{v_4, t\}$

$$\begin{aligned} f(S,T) &= f(v_2, v_4) + f(v_3, t) - f(v_4, v_3) \\ &= 11 + 15 - 7 \\ &= 19 \end{aligned}$$

$$\begin{aligned} c(S,T) &= c(v_2, v_4) + c(v_3, t) \\ &= 14 + 20 \\ &= 34 \end{aligned}$$

- Gleicher Nettofluss wie vorher, aber höhere Kapazität

Schnitte

- Lemma
  Für jeden Schnitt (S, T) gilt: $f(S, T) = |f|$,
  d.h. «Wert des Flusses» =
  «Nettofluss über Schnitt»

- Korollar
  «Wert eines beliebigen Flusses» $\leq$
  «Kapazität eines beliebigen Schnitts»

- Folgerung:
  «Maximaler Fluss» $\leq$
  «Kapazität des minimalen Schnitts»

Max-flow min-cut Theorem

- Theorem:
  Sei f ein Fluss in einem Flussnetzwerk G,
  dann sind folgende Bedingungen äquivalent

  1. f ist maximaler Fluss in G.
  2. G_f enthält keine Erweiterungspfade.
  3. $|f| = c(S,T)$ für einen Schnitt (S,T) von G.

Beweis

- (1) → (2): Falls G_f Erweiterungspfad p hätte, dann könnten wir Fluss mit Wert $|f| + |f_p| > |f|$ erhalten; Widerspruch zu Annahme (1)
- (3) → (1): Korollar $|f| \leq c(S,T)$ für jeden Schnitt.
  Darum: $|f| = c(S,T)$ → f ist ein maximaler Fluss

Beweis

- (2) → (3): Nehmen an, dass G_f keinen Erweiterungspfad hat. Definieren
  - $S = \{v \in V: \text{ existiert Pfad } s \to v \text{ in } G_f\}$, $T = V - S$
    - t muss in T sein, sonst gibt es Erweiterungspfad
    - Also ist (S, T) ein Schnitt
  - Betrachte $u \in S$, $v \in T$
    - $(u, v) \in E$, dann $f(u, v) = c(u, v)$, sonst wäre $(u, v) \in E_f$ und damit $v \in S$
    - $(v, u) \in E$, dann $f(v, u) = 0$, sonst wäre $c_f(u, v) = f(v, u)$ positiv, d.h. $(u, v) \in E_f$, also $v \in S$
    - $(u, v) \notin E$ und $(v, u) \notin E$, dann $f(u, v) = f(v, u) = 0$

$$f(S,T) = \sum_{u \in S} \sum_{v \in T} f(u,v) - \sum_{v \in T} \sum_{u \in S} f(v,u)$$
$$= \sum_{u \in S} \sum_{v \in T} c(u,v) - \sum_{v \in T} \sum_{u \in S} 0 = c(S,T)$$

  - Mit vorherigem Lemma folgt $|f| = f(S,T) = c(S,T)$

Ford-Fulkerson Algorithmus

FORD-FULKERSON(G, s, t)

```
1  for each edge (u, v) ∈ G.E
2      (u, v).f = 0
3  while there is a path p from s to t in the residual network G_f
4      c_f(p) = min{c_f(u, v) : (u, v) is in p}
5      for each edge (u, v) in p
6          if (u, v) ∈ E
7              (u, v).f = (u, v).f + c_f(p)
8          else (v, u).f = (v, u).f − c_f(p)
```

- Analyse
  - Falls ganzzahlige Kapazitäten, dann vergrössert jeder Schritt $|f|$ um ≥ 1
  - Falls maximaler Fluss Wert $|f^*|$ hat, dann $\leq |f^*|$ Iterationen
  - Komplexität $O(E\,|f^*|)$

Edmonds-Karp Algorithmus

- Finde erweiternde Pfade mit Breitensuche in Restnetzwerk
- Aufwand $O(VE^2)$
  - Beweis siehe Buch

Graphenalgorithmen

Maximaler Fluss

- Einleitung
- Flussnetzwerke
- Ford-Fulkerson Methode
- Maximales bipartites Matching

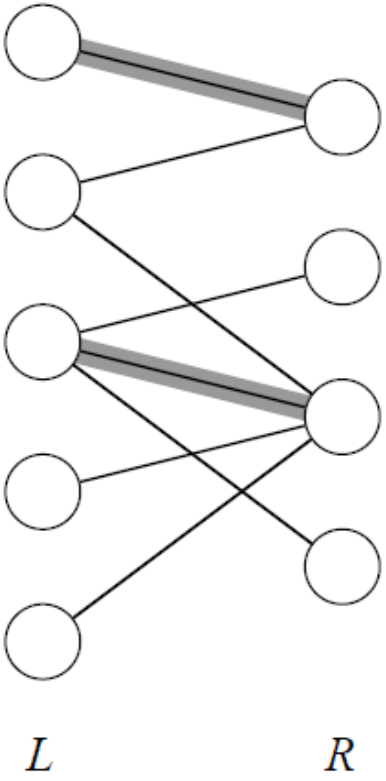
Maximales bipartites Matching

- Eines von vielen Problemen, das gelöst werden kann, indem man es als Flussproblem formuliert
- Ungerichteter Graph $G = (V, E)$ heisst bipartit falls V in $V = L \cup R$ partitioniert werden kann, so dass alle Kanten zwischen L und R sind

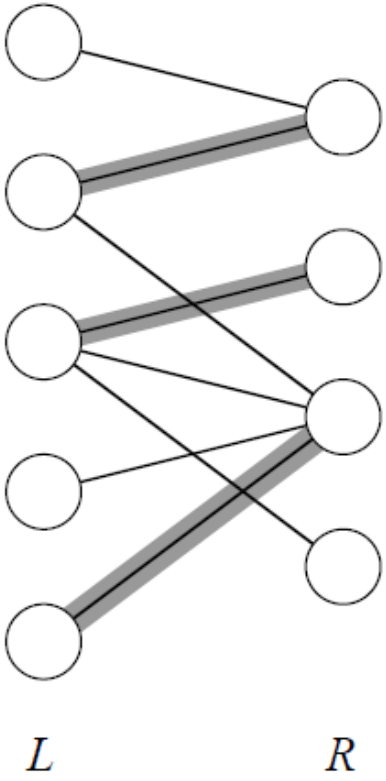
Maximales bipartites Matching

- Matching: Eine Teilmenge $M \subseteq E$, so dass für alle $v \in V$, ≤ 1 Kante von M auf v inzident ist
  - Knoten heisst matched, falls Kante inzident auf ihn, sonst unmatched
- Maximales Matching: ein Matching mit grösster Kardinalität
  - M ist maximal falls $|M| \geq |M'|$ für alle M'
- Problem: gegeben bipartiter Graph (mit Partitionierung), finde maximales Matching

Maximales bipartites Matching



Kein maximales Matching



Maximales Matching

Maximales bipartites Matching

- Anwendungsbeispiel: Flugzeuge einer Menge von Routen zuweisen
  - L: Menge von Flugzeugen
  - R: Menge von Routen
  - $(u, v) \in E$ falls Flugzeug u Route v fliegen kann
- Wollen grösstmögliche Anzahl Routen, die von den Flugzeugen geflogen werden können

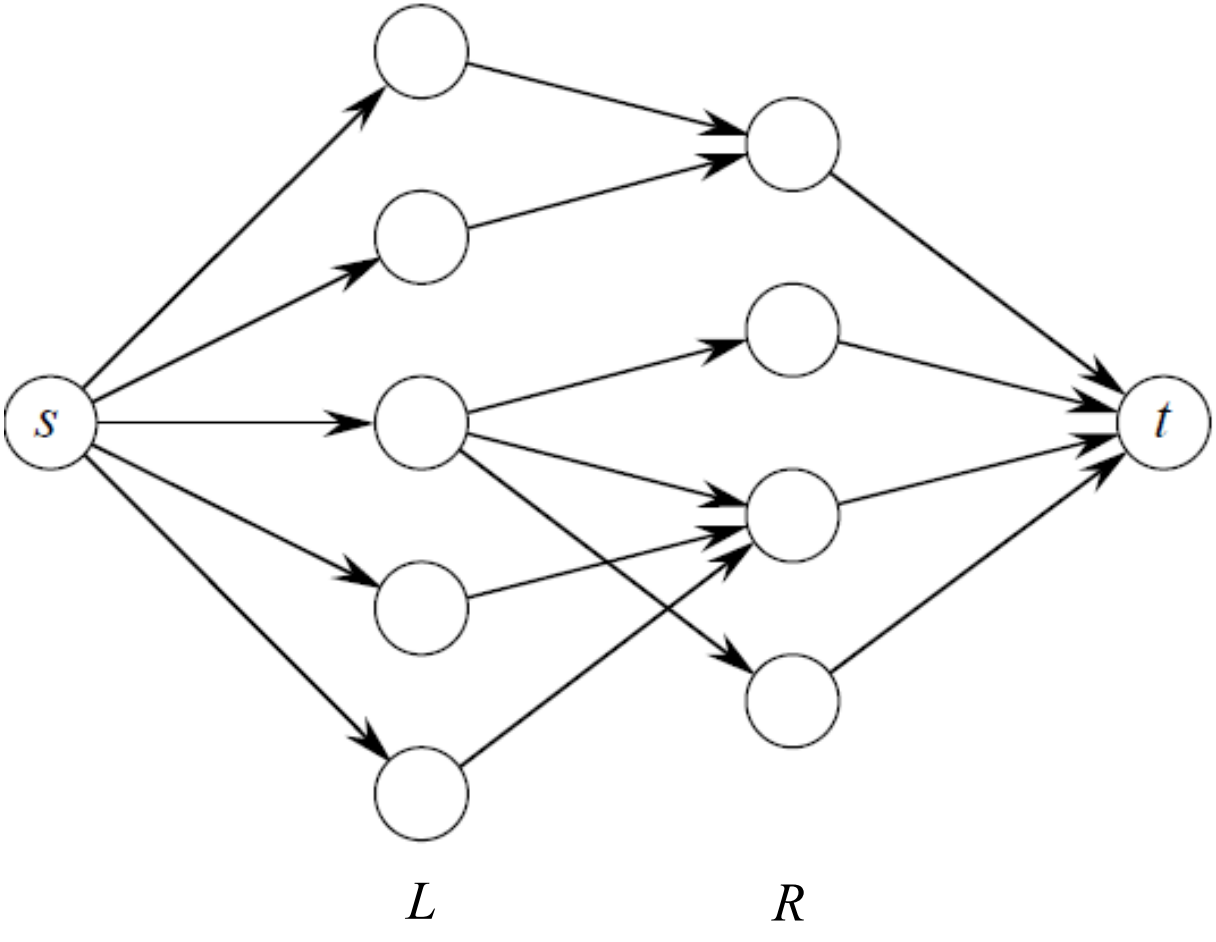
Maximales bipartites Matching

- Gegeben G, definiere Flussnetzwerk

$$G' = (V', E')$$

  - Knoten $V' = V \cup \{s, t\}$
  - Kanten $E' = \{(s, u) : u \in L\}$ // Alle Knoten links Verbindung mit s
    $\cup \{(u, v) : u \in L, v \in R, (u, v) \in E\}$
    $\cup \{(v, t) : v \in R\}$
  - Kapazität $c(u, v) = 1$ für alle $(u, v) \in E'$

Formulierung als Flussproblem



Formulierung als Flussproblem

Lemma

Sei $G = (V, E)$ bipartiter Graph mit $V = L \cup R$, $G' = (V', E')$ entsprechendes Flussnetzwerk.

Wenn M Matching in G, dann gibt es Fluss f in G' mit Wert $|f| = |M|$.

Wenn f Fluss in G, dann gibt es Matching M in G mit Kardinalität $|M| = |f|$.

Finde maximalen Fluss in G'

- Benutze Kanten mit Fluss 1 im Matching
- Beweise siehe Buch

Analyse

- Jeder Knoten in V hat ≥ 1 inzidente Kanten → $|E| \geq \frac{|V|}{2}$ → $|V| \leq 2|E|$
- Deshalb $|E| \leq |E'| = |E| + |V| \leq 3|E|$
- Deshalb $|E'| = \Theta(E)$
- Wert des maximalen Fluss $O(V)$
- Ford-Fulkerson hat Komplexität $O(E\,|f^*|) = O(E\,V)$ wobei $|f^*|$ Kapazität des maximalen Fluss

Anwendung von Max-flow min-cut

- Viele Optimierungsprobleme in der Bildverarbeitung
  - Z.B. Automatische Bildsegmentierung

http://research.microsoft.com/pubs/67890/siggraph04-grabcut.pdf

Nächstes Mal

- Repetition