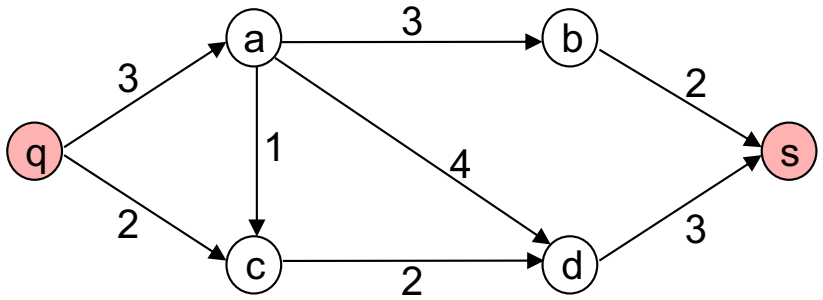
11. Flüsse in Graphen

- Netzwerke
- Flüsse
- Residualgraphen
- Algorithmus von Ford-Fulkerson

Netzwerk

- Ein Netzwerk ist ein gewichteter und gerichteter Graph mit zwei speziellen Knoten:
  - Quelle (engl. source): Knoten ohne eingehende Kanten
  - Senke (engl. sink): Knoten ohne ausgehende Kanten.
- Die Gewichte c(v,w) werden auch als Kapazitäten bezeichnet.

Beispiel:



Netzwerk mit Quelle q, Senke s und Kapazitäten.

Fluss

- Ein Fluss f in einem Netzwerk G ordnet jeder Kante (v,w) eine Zahl f(v,w) mit folgenden Eigenschaften zu:

  (1) Kapazitätsbeschränkung: $0 \leq f(v,w) \leq c(v,w)$.

  (2) Für jeden Knoten (außer Quelle und Senke) gilt die Erhaltungseigenschaft: Die Summe der eingehenden Flüsse ist gleich der Summe der ausgehenden Flüsse.

- Folgerung: Die Summe der Flüsse, die die Quelle verlassen, muss gleich der Summe der Flüsse sein, die an der Senke ankommen.

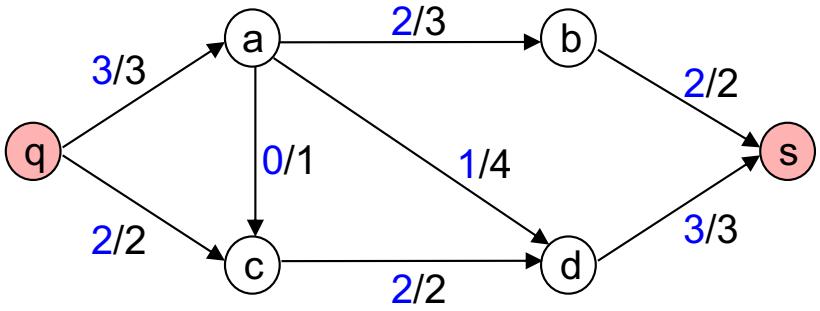
- Der Wert eines Flusses ist gleich der Summe der Flüsse, die die Quelle verlassen.

- Ziel: bestimme einen maximalen Fluss (d.h. Fluss mit maximalen Wert). Maximaler Fluss muss nicht eindeutig sein.

Typische Anwendungen

- Straßenverkehrsnetz
  - Kapazitäten geben maximal mögliche Verkehrsdichte je Straße an.
  - Problemstellung:
    Welchen Verkehrsfluss verkraftet eine Stadt, für die ein Straßenverkehrsnetz gegeben ist?
- Kanalisationsysytem
  - Kapazitäten geben maximale Wassermenge an, die je Zeiteinheit durch ein Rohr fließen kann.
  - Problemstellung:
    Welche Wassermenge lässt sich durch eine Kanalisation höchstens abtransportieren.

Beispiel für Netzwerk mit maximalem Fluss



Fluss / Kapazität

Maximaler Fluss hat den Wert 5.

Idee des Algorithmus

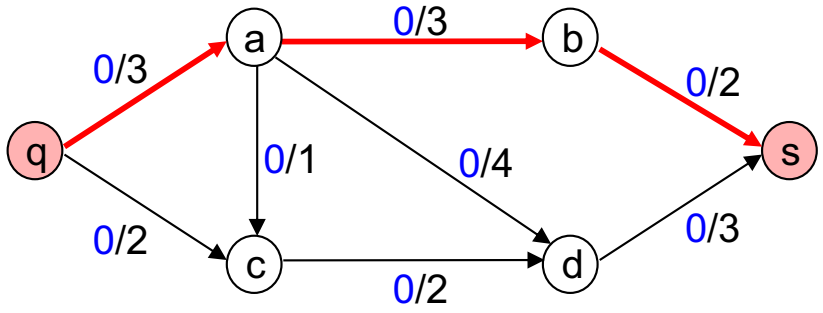
Starte mit Null-Fluss, d.h. f(v,w) = 0 für alle Kanten (v,w);

do {

(1) // Erweiterungsweg suchen:
Suche einen Weg p von q nach s, bei dem jede Kante um einen Fluss Δf vergrößert werden kann;

(2) // Flusserweiterung:
vergrößere für jede Kante (v,w) im Weg p den Fluss f(v,w) um Δf;

} **while** (Fluss konnte erweitert werden);

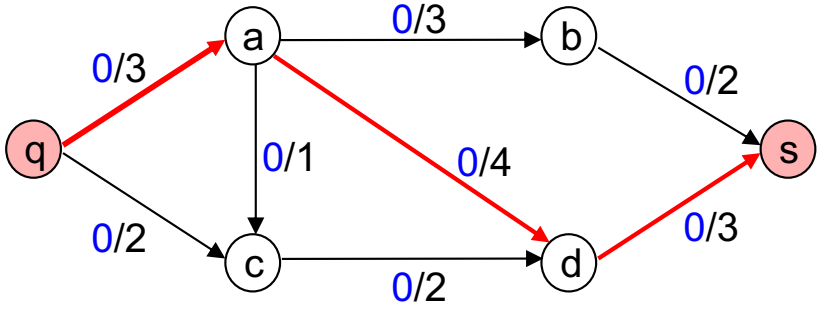


Null-Fluss.

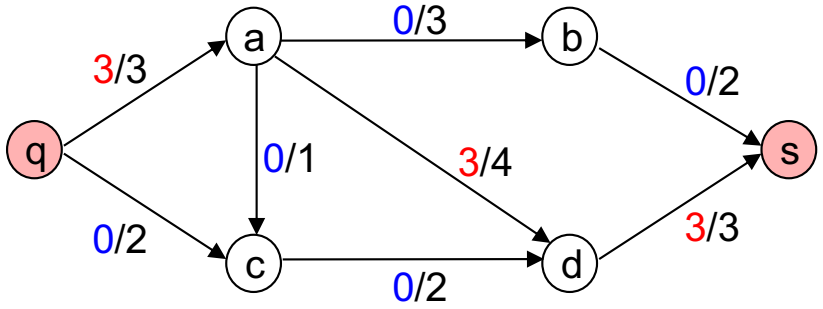
q, a, b, s ist ein möglicher Erweiterungsweg:
jede Kante kann um den Fluss $\Delta f = 2$ vergrößert werden.

Problem

- Die Wahl eines Erweiterungsweges kann in eine Sackgasse führen (suboptimale Lösung).
- Beispiel



Erweiterungsweg,
bei dem jede Kante um $\Delta f = 3$
vergrößert werden kann.



Nach Flusserweiterung:
Fluss hat nun den Wert 3.

- Es gibt nun keinen weiteren Erweiterungsweg, obwohl es eine Lösung mit einem Flusswert von 5 gibt (siehe S. 11-4)

Residualgraphen

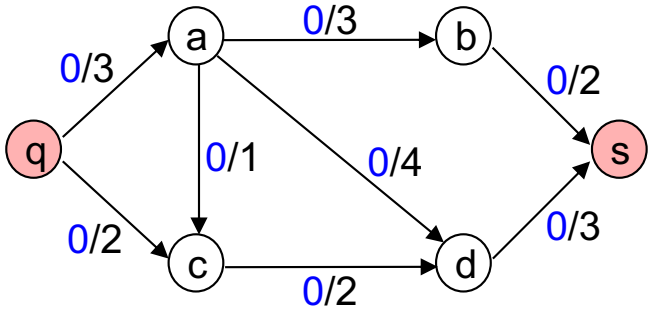
- Man erlaubt nach der Vergrößerung auch wieder eine Verkleinerung von Kantenflüssen.
- Verwalte zusätzlich einen sogenannten Residualgraphen G_r (residual = als Rest zurückbleibend).
- Im Residualgraph wird gespeichert, um welchen Wert der Fluss jeder Kante noch verändert werden darf:

  (1) Hat eine Kante (v,w) den Fluss f(v,w) > 0,
  dann darf der Fluss verkleinert werden.
  Im Residualgraphen G_r wird die Kante (w,v) mit dem Residualfluss f(v,w) abgespeichert. Beachte: Kante in G_r verläuft in umgekehrter Richtung.
  (Flusserhöhung in umgekehrter Richtung verkleinert den Fluss.)

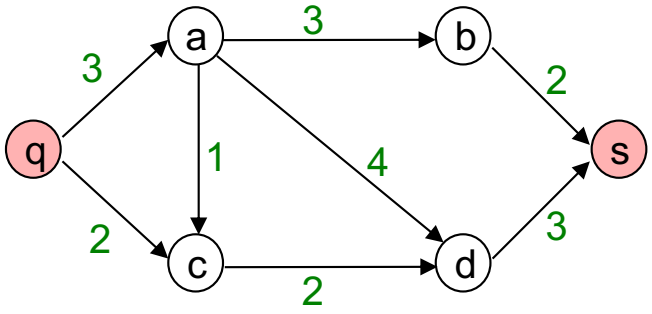
  (2) Hat eine Kante (v,w) den Fluss f(v,w) < c(v,w),
  dann darf der Fluss vergrößert werden.
  Im Residualgraphen G_r wird die Kante (v,w) mit dem Residualfluss c(v,w) - f(v,w) abgespeichert.

Beispiel

Residualgraph für Nullfluss



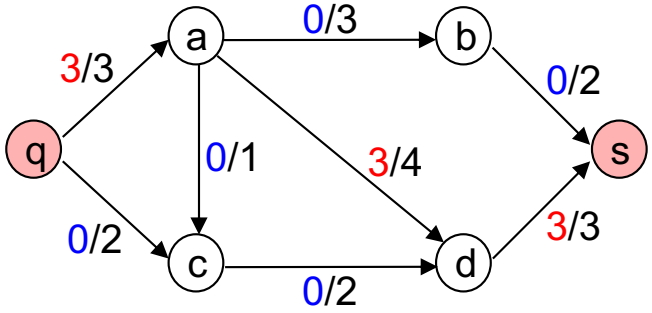
Graph G mit Nullfluss.



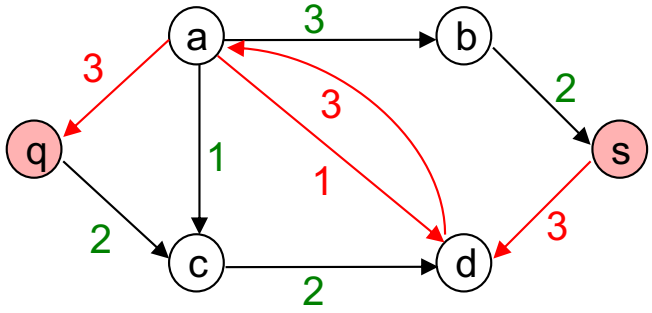
Residualgraph G_r mit Residualflüssen

Residualgraph nach Flusserweiterung

q, a, d, s wird als Erweiterungsweg gewählt. Fluss lässt sich um $\Delta f = 3$ vergrößern.



Graph G mit Flusswert 3.



Residualgraph G_r mit Residualflüssen

Fluss von q nach a darf um bis zu 3 verringert werden.

Fluss von a nach d darf um bis zu 1 erhöht und bis zu 3 verringert werden.

Fluss von d nach s darf um bis zu 3 verringert werden.

Algorithmus zur Berechnung eines maximalen Flusses

Initialisiere Graph G mit Null-Fluss, d.h. f(v,w) = 0 für alle Kanten (v,w);
Initialisiere Residualgraph G_r;

do {
(1) // Erweiterungsweg suchen:
Suche im Residualgraphen G_r einen Weg p von q nach s;
Δf = Minimum aller Residual-Flüsse im Weg p;

(2) // Fluss vergrößern:
vergrößere für jede Kante (v,w) im Weg p den Fluss f(v,w) um Δf
(beachte dabei, dass umgekehrte Residualflüsse den Fluss verkleinern);
führe entsprechende Änderungen im Residualgraphen durch;

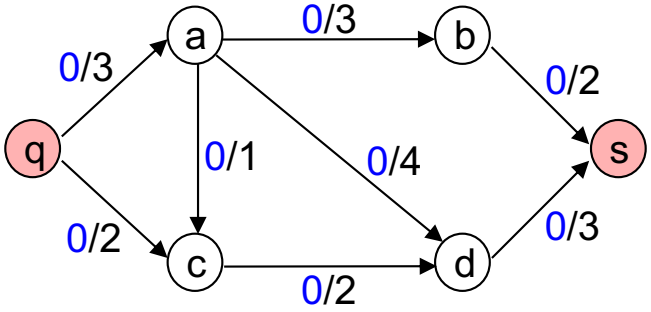
} **while** (Fluss konnte vergrößert werden);

2 Ansätze, um einen Erweiterungsweg von q nach s im Residualgraphen zu suchen:

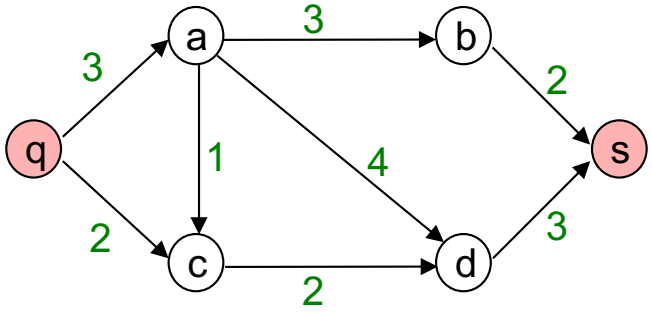
- suche den Weg mit dem größten Δf
  (wie bei kürzeste Wege in Distanzgraphen; siehe Algorithmus von Dijkstra)
- suche Weg mit kleinster Kantenzahl
  (wie kürzeste Wege in ungewichteten Graphen durch erweiterte Breitensuche)

Beispiel (1)

Starte mit Nullfluss



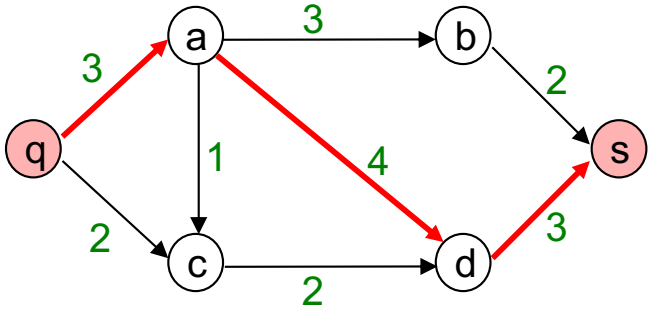
Graph G mit Nullfluss



Residualgraph G_r mit Residualflüssen

Beispiel (2)

Suche Weg mit größten Δf im Residualgraphen:

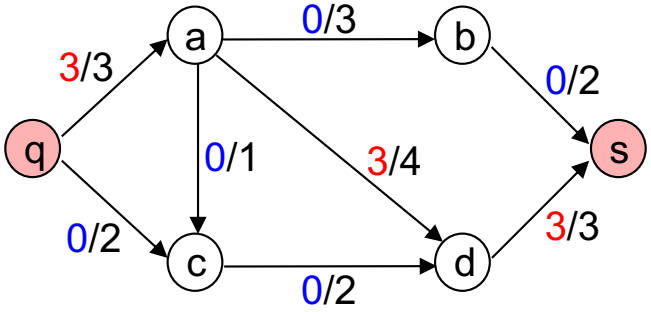


Residualgraph G_r

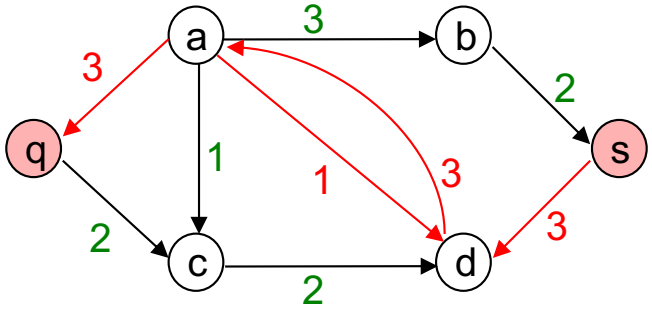
q, a, d, s ist Weg mit größtem Δf.

Δf = Minimum der Residual-Flüsse im Weg = 3.

Flusserweiterung um Δf:



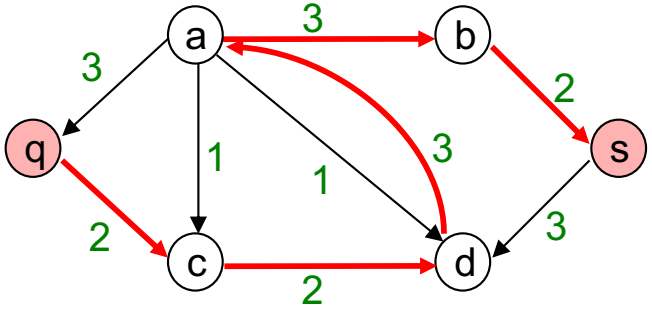
Graph G mit Flusswert 3



Residualgraph G_r

Beispiel (3)

Suche Weg mit größten Δf im Residualgraphen:

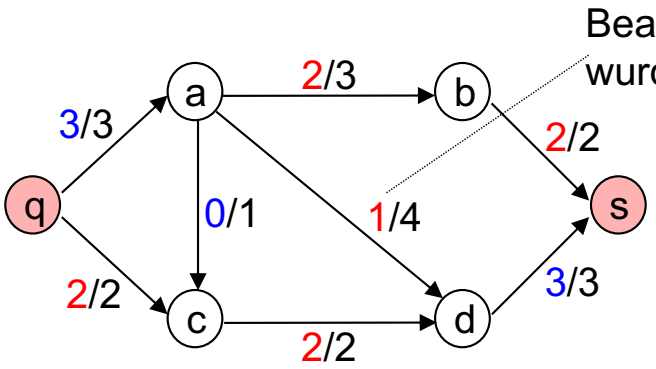


Residualgraph G_r

q, c, d, a, b, s ist Weg mit größtem Δf.

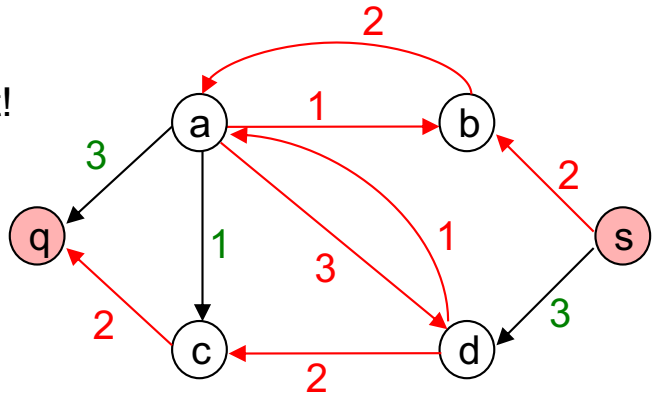
Δf = Minimum der Residual-Flüsse im Weg = 2.

Flusserweiterung um Δf:



Beachte: Fluss wurde verkleinert!

Graph G mit Flusswert 5



Residualgraph G_r

Ende:

Es gibt im Residualgraphen keine Wege mehr von q nach s. Maximaler Fluss gefunden!

Analyse und Bemerkungen

- Analyse ergibt:

  $T = O(|E|^2 \log|V|)$

  (Beweis siehe [Turau 2004]).

- Bemerkungen:
  - Der hier beschriebene Algorithmus geht zurück auf Ford und Fulkerson (1956). Analyse von Karp und Edmonds (1972).
  - Man beachte, dass bei einem dichten Graphen (d.h. $|E| = O(|V|^2)$ die Laufzeit anwächst auf $T = O(|V|^4 \log|V|)$
  - Es gibt inzwischen wesentlich schnellere Algorithmen:
    Preflow-Push-Algorithmus von Goldberg, 1985: $T = O(|V|^3)$.
    Mittels spezieller Datenstrukturen erreichte Goldberg 1988 sogar:
    $T = O(|E| \, |V| \log(|V|^2/|E|)$.