Propädeutikum Diskrete Mathematik

Flüsse und Algorithmus von Ford und Fulkerson

Prof. Dr. R. Hemmecke
B.Sc. W.F. Riedl   Dipl-Math. M. Silbernagl

Technische Universität München

WS 2013/14

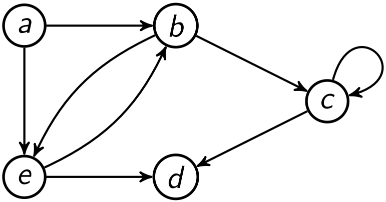
Gerichteter Graph

$G = (V, A)$ ist ein gerichteter Graph, wenn $A \subset V \times V$.

[Ungerichteter Graph: $G = (V, E)$ mit $E \subset \binom{V}{2}$]

Ein- und ausgehende Kanten von $x \in V$:

- $N^-(x) := \{y \in V : (y, x) \in A\}$
- $N^+(x) := \{y \in V : (x, y) \in A\}$



$$A = \{(a, b), (a, e), (b, c), (b, e), (c, c), (c, d), (e, b), (e, d)\}$$

$N^+(a) = \{b, e\}$
$N^-(d) = \{c, e\}$

Netzwerk

Ein Netzwerk ist ein Tupel $N = (V, A, s, t, c)$ mit

- ger. Graph (V, A, c) mit *Kapazitätsfunktion* $c : A \to \mathbb{R}_{\geq 0}$ und
- zwei Knoten $s \in V$ (*Quelle*) und $t \in V$ (*Senke*).

Netzwerk

Ein Netzwerk ist ein Tupel $N = (V, A, s, t, c)$ mit

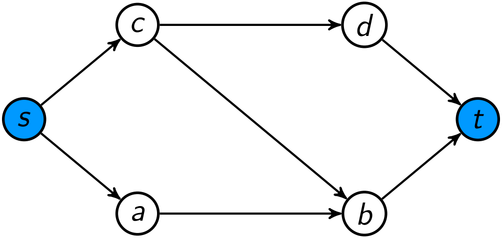
- ger. Graph (V, A, c) mit *Kapazitätsfunktion* $c : A \to \mathbb{R}_{\geq 0}$ und
- zwei Knoten $s \in V$ (*Quelle*) und $t \in V$ (*Senke*).

Fluss

Sei $N = (V, A, s, t, c)$ ein Netzwerk. Eine Abbildung $f : A \to \mathbb{R}_{\geq 0}$ heißt s, t-*Fluss* (flow) in N, wenn

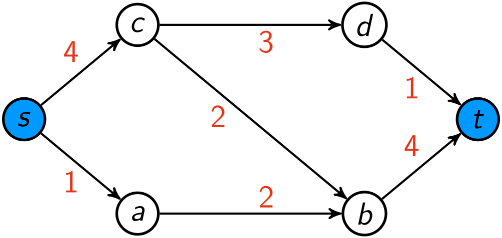
1. Flusserhaltung
$$\sum_{y \in N^-(x)} f((y, x)) = \sum_{y \in N^+(x)} f((x, y)) \qquad \forall\, x \in V \setminus \{s, t\}$$
2. Zulässigkeit
$$f((x, y)) \leq c((x, y)) \qquad \forall\, (x, y) \in A.$$

Beispiel: Netzwerke und Flüsse



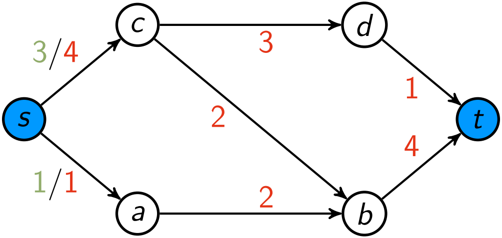
- Ger. Graph mit Quelle und Senke

Beispiel: Netzwerke und Flüsse



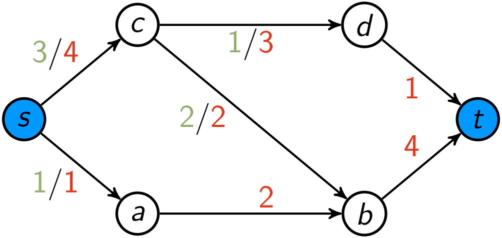
- Ger. Graph mit Quelle und Senke
- Kapazitätsgrenze $c : A \to \mathbb{R}_{\geq 0}$

Beispiel: Netzwerke und Flüsse



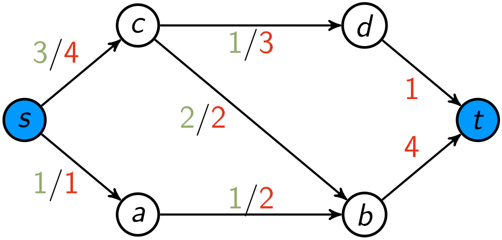
- Ger. Graph mit Quelle und Senke
- Kapazitätsgrenze $c : A \to \mathbb{R}_{\geq 0}$
- s, t-Fluss $f : A \to \mathbb{R}_{\geq 0}$

Beispiel: Netzwerke und Flüsse



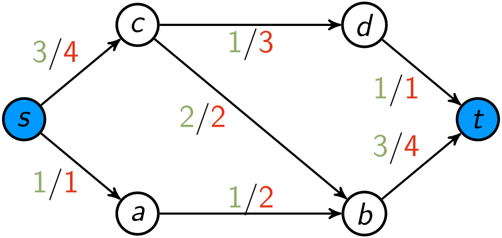
- Ger. Graph mit Quelle und Senke
- Kapazitätsgrenze $c : A \to \mathbb{R}_{\geq 0}$
- s, t-Fluss $f : A \to \mathbb{R}_{\geq 0}$

Beispiel: Netzwerke und Flüsse



- Ger. Graph mit Quelle und Senke
- Kapazitätsgrenze $c : A \to \mathbb{R}_{\geq 0}$
- s, t-Fluss $f : A \to \mathbb{R}_{\geq 0}$

Beispiel: Netzwerke und Flüsse



- Ger. Graph mit Quelle und Senke
- Kapazitätsgrenze $c : A \to \mathbb{R}_{\geq 0}$
- s, t-Fluss $f : A \to \mathbb{R}_{\geq 0}$

Vereinbarung

Schreibe $f(x, y)$ statt $f((x, y))$ und $c(x, y)$ statt $c((x, y))$.

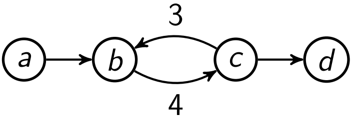
Vereinbarung

Schreibe $f(x, y)$ statt $f((x, y))$ und $c(x, y)$ statt $c((x, y))$.

Bemerkung

In Zukunft gehen wir davon aus, dass für alle $x, y \in V$ gilt

$$f(x, y) > 0 \quad \Rightarrow \quad f(y, x) = 0.$$



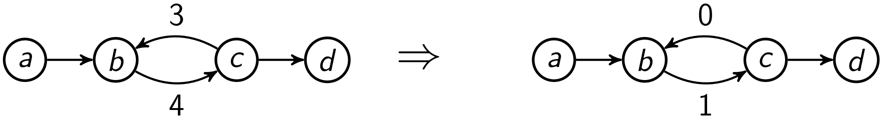
Vereinbarung

Schreibe $f(x,y)$ statt $f((x,y))$ und $c(x,y)$ statt $c((x,y))$.

Bemerkung

In Zukunft gehen wir davon aus, dass für alle $x, y \in V$ gilt

$$f(x,y) > 0 \quad \Rightarrow \quad f(y,x) = 0.$$



Wert eines Flusses

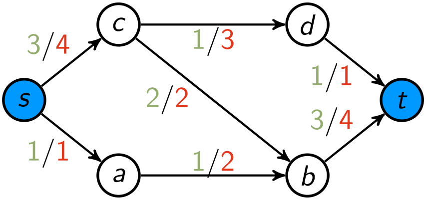
Der *Wert* eines Flusses f ist definiert als

$$\operatorname{val}(f) := \sum_{y \in N^+(s)} f(s,y) - \sum_{y \in N^-(s)} f(y,s).$$

Wert eines Flusses

Der *Wert* eines Flusses f ist definiert als

$$\mathrm{val}(f) := \sum_{y \in N^+(s)} f(s,y) \;-\; \sum_{y \in N^-(s)} f(y,s).$$



$$\mathrm{val}(f) = (3+1) - 0 = 4$$

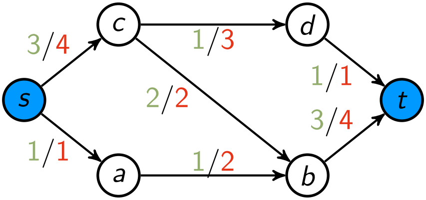
Wert eines Flusses

Der *Wert* eines Flusses f ist definiert als

$$\mathrm{val}(f) := \sum_{y \in N^+(s)} f(s,y) - \sum_{y \in N^-(s)} f(y,s).$$

Ein Fluss f hat *maximalen Wert* im Netzwerk N, wenn

$$\mathrm{val}(f) \geq \mathrm{val}(f') \qquad \text{für alle Flüsse } f' \text{ in } N.$$



$\mathrm{val}(f) = (3+1) - 0 = 4$

Proposition

Jedes Netzwerk besitzt einen Fluss maximalen Werts.

Restnetzwerk

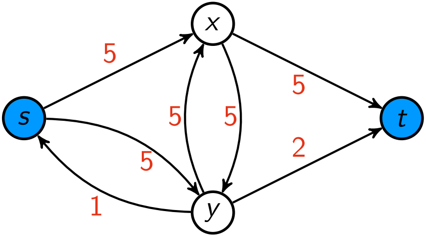
Sei $N = (V, A, s, t, c)$ ein Netzwerk und f ein Fluss in N. Wir erweitern zunächst die Kapazitätsfunktion zu $c : V \times V \to \mathbb{R}_{\geq 0}$ durch $c(x, y) = 0$ für alle $(x, y) \notin A$.

Das Restnetzwerk $N_f = (V, A_f, s, t, c_f)$ ist definiert durch $A_f = \{(x, y) : c_f(x, y) > 0\}$ und

$$c_f(x,y) = \begin{cases} c(x,y) - f(x,y), & \text{falls } f(x,y) > 0 \\ c(x,y) + f(y,x), & \text{falls } f(y,x) > 0 \\ c(x,y), & \text{sonst.} \end{cases}$$

Beispiel: Restnetzwerk

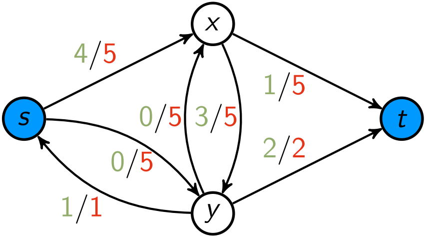
$$c_f(x,y) = \begin{cases} c(x,y) - f(x,y), & \text{falls } f(x,y) > 0 \\ c(x,y) + f(y,x), & \text{falls } f(y,x) > 0 \\ c(x,y), & \text{sonst.} \end{cases}$$



Kapazitätsobergrenze $c : A \to \mathbb{R}_{\geq 0}$

Beispiel: Restnetzwerk

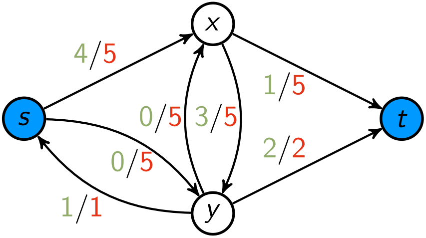
$$c_f(x,y) = \begin{cases} c(x,y) - f(x,y), & \text{falls } f(x,y) > 0 \\ c(x,y) + f(y,x), & \text{falls } f(y,x) > 0 \\ c(x,y), & \text{sonst.} \end{cases}$$

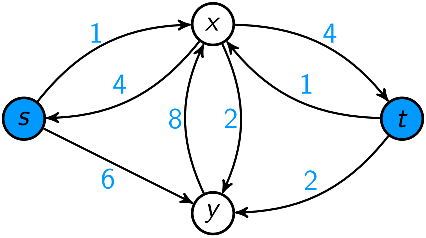


Kapazitätsobergrenze $c : A \to \mathbb{R}_{\geq 0}$
s, t-Fluss $f : A \to \mathbb{R}_{\geq 0}$

Beispiel: Restnetzwerk

$$c_f(x,y) = \begin{cases} c(x,y) - f(x,y), & \text{falls } f(x,y) > 0 \\ c(x,y) + f(y,x), & \text{falls } f(y,x) > 0 \\ c(x,y), & \text{sonst.} \end{cases}$$





Kapazitätsobergrenze $c : A \to \mathbb{R}_{\geq 0}$

s, t-Fluss $f : A \to \mathbb{R}_{\geq 0}$

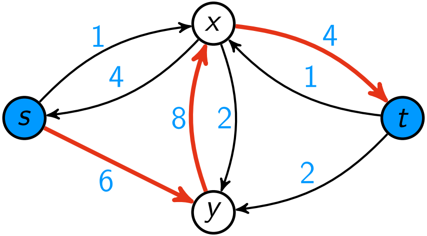
Restkapazitäten $c_f : A \to \mathbb{R}_{\geq 0}$

Gerichteter Weg

Sei $G = (V, A)$ ein gerichteter Graph. Dann heißt $(v_1, v_2, v_3, \ldots, v_l)$ gerichteter Weg in G falls $(v_i, v_{i+1}) \in A$ für alle $i \in [l-1]$.

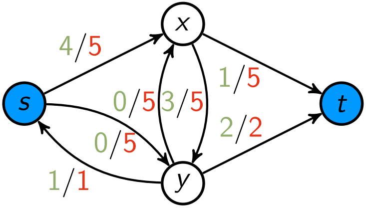
Gerichteter Weg

Sei $G = (V, A)$ ein gerichteter Graph. Dann heißt $(v_1, v_2, v_3, \ldots, v_l)$ gerichteter Weg in G falls $(v_i, v_{i+1}) \in A$ für alle $i \in [l-1]$.

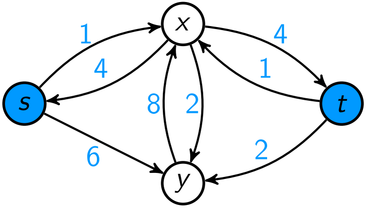


Beispiel: Fluss erhöhen

Fluss:

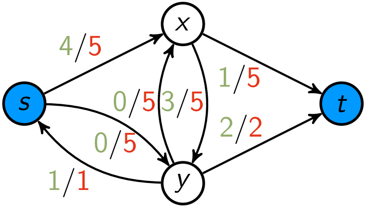


Restnetzwerk:

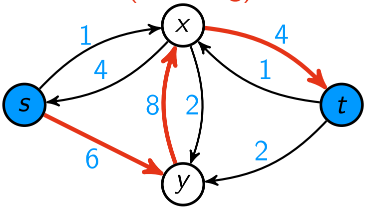


Beispiel: Fluss erhöhen

Fluss:

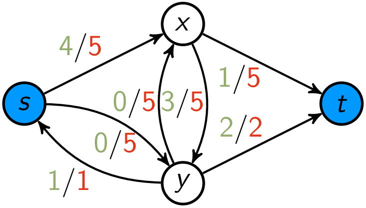


Restnetzwerk (mit Weg):

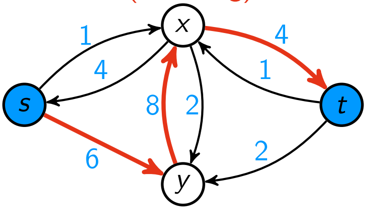


Beispiel: Fluss erhöhen

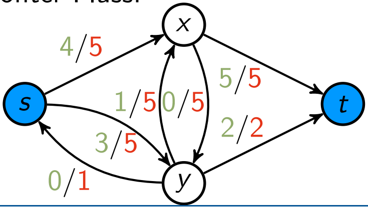
Fluss:



Restnetzwerk (mit Weg):



Erhöhter Fluss:



Satz: Fluss erhöhen

Wenn f ein Fluss in N und W ein gerichteter s,t-Weg im Restnetzwerk N_f ist, dann kann f entlang W zu einem Fluss f' erhöht werden mit

$$\mathrm{val}(f') = \mathrm{val}(f) + \delta,$$

wobei $0 < \delta = \min\{c_f(e) : W \text{ benutzt Kante } e\}$.

Satz: Fluss erhöhen

Wenn f ein Fluss in N und W ein gerichteter s, t-Weg im Restnetzwerk N_f ist, dann kann f entlang W zu einem Fluss f' erhöht werden mit

$$\mathrm{val}(f') = \mathrm{val}(f) + \delta,$$

wobei $0 < \delta = \min\{c_f(e) : W \text{ benutzt Kante } e\}$.

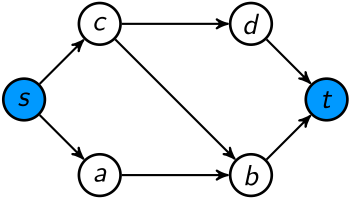
Satz: Maximaler Fluss

Sei f Fluss in N. Wenn es in N_f keinen gerichteten s, t-Weg mehr gibt, dann hat f maximalen Wert.

Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

```
for (x, y) ∈ A do
    f(x, y) ← 0;
end
while ∃ s,t-Weg W in N_f do
    Erhöhe f entlang von W;
end
```

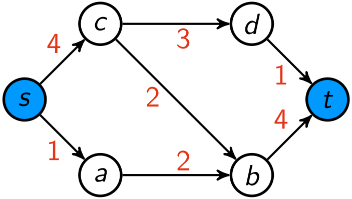


s Quelle, t Senke

Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0$;
end
while $\exists$ s, t*-Weg* W *in* N_f **do**
  Erhöhe f entlang von W;
end



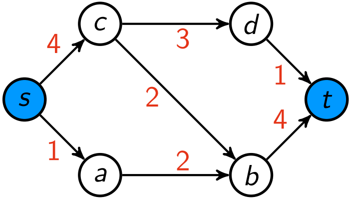
s Quelle, t Senke

c Kapazitätsobergrenze

Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0$;
end
while $\exists$ s, t-*Weg* W *in* N_f **do**
  Erhöhe f entlang von W;
end



s Quelle, t Senke

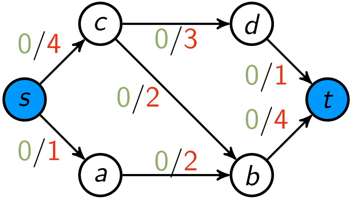
c Kapazitätsobergrenze

f Fluss

Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

```
for (x, y) ∈ A do
    f(x, y) ← 0;
end
while ∃ s, t-Weg W in N_f do
    Erhöhe f entlang von W;
end
```



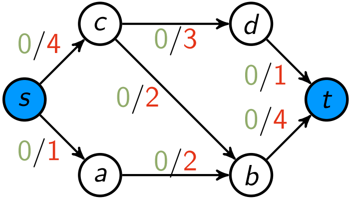
s Quelle, t Senke

c Kapazitätsobergrenze

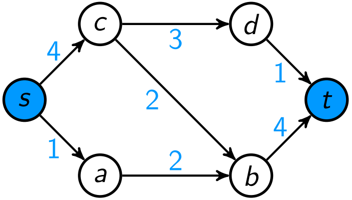
f Fluss

Input : Netzwerk
$N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0$;
end
while $\exists$ s, t-*Weg* W *in* N_f **do**
  Erhöhe f entlang von W;
end



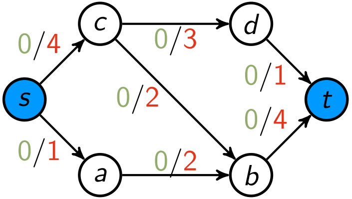
s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss



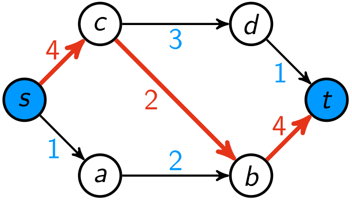
c_f Restkapazität

Input : Netzwerk $N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0;$
end
while $\exists\ s, t\text{-Weg } W \text{ in } N_f$ **do**
  Erhöhe f entlang von W;
end



s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss

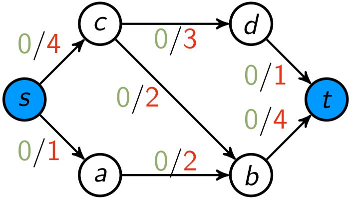


c_f Restkapazität

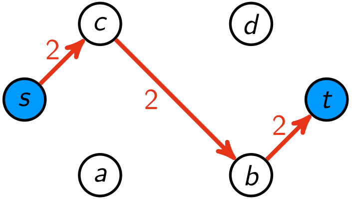
Input : Netzwerk
$N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0$;
end
while $\exists$ *s, t-Weg W in* N_f **do**
  Erhöhe f entlang von W;
end



s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss

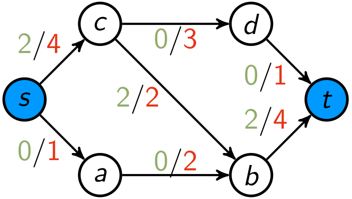


c_f Restkapazität

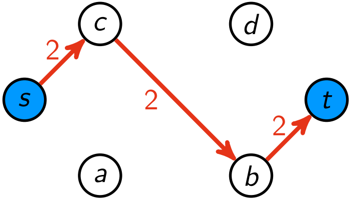
Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0;$
end
while $\exists$ s, t-*Weg* W *in* N_f **do**
  Erhöhe f entlang von W;
end



s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss

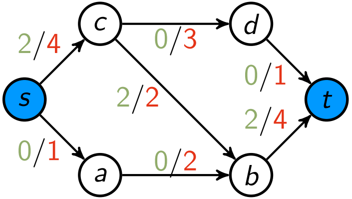


c_f Restkapazität

Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

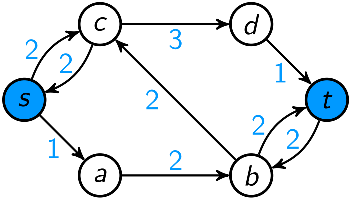
```
for (x, y) ∈ A do
    f(x, y) ← 0;
end
while ∃ s, t-Weg W in N_f do
    Erhöhe f entlang von W;
end
```



s Quelle, t Senke

c Kapazitätsobergrenze

f Fluss

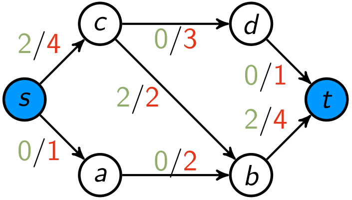


c_f Restkapazität

Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

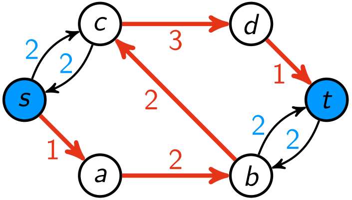
for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0;$
end
while $\exists$ *s, t-Weg W in* N_f **do**
  Erhöhe f entlang von W;
end



s Quelle, t Senke

c Kapazitätsobergrenze

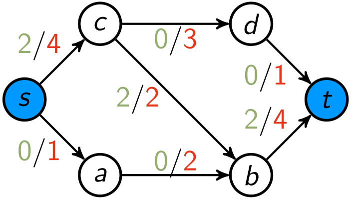
f Fluss



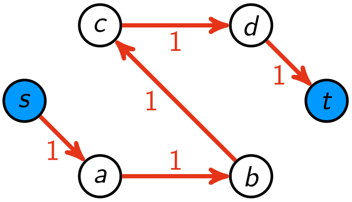
c_f Restkapazität

Input : Netzwerk $N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
    $f(x, y) \leftarrow 0$;
end
while $\exists$ s, t-*Weg* W *in* N_f **do**
    Erhöhe f entlang von W;
end



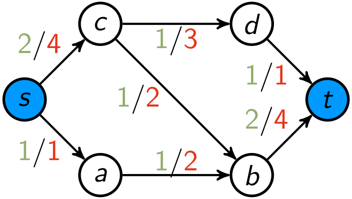
s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss



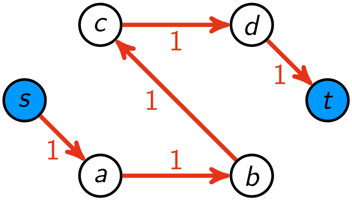
c_f Restkapazität

Input : Netzwerk $N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

```
for (x, y) ∈ A do
    f(x, y) ← 0;
end
while ∃ s, t-Weg W in N_f do
    Erhöhe f entlang von W;
end
```



s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss

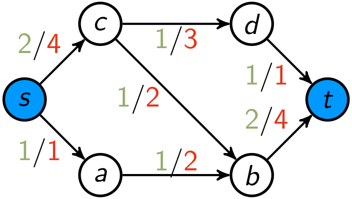


c_f Restkapazität

Input : Netzwerk
$N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

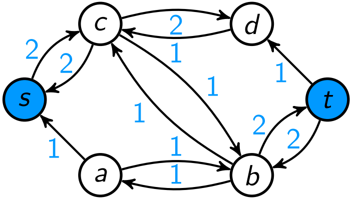
for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0$;
end
while $\exists$ *s, t-Weg W in N_f* **do**
  Erhöhe f entlang von W;
end



s Quelle, t Senke

c Kapazitätsobergrenze

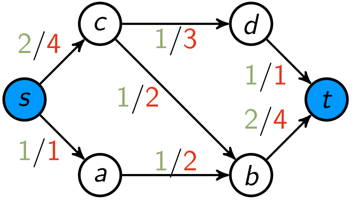
f Fluss



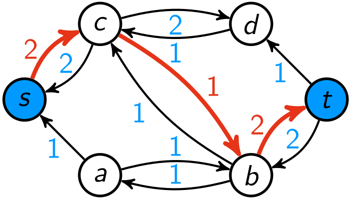
c_f Restkapazität

Input : Netzwerk
$N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0;$
end
while $\exists$ *s, t-Weg W in* N_f **do**
  Erhöhe f entlang von W;
end



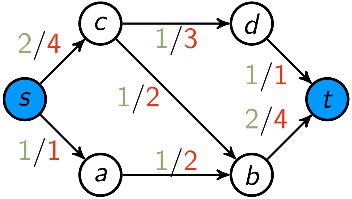
s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss



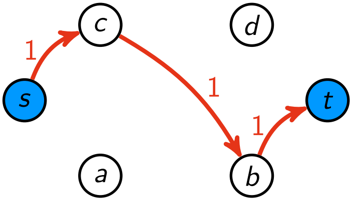
c_f Restkapazität

Input : Netzwerk $N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
    $f(x, y) \leftarrow 0;$
end
while $\exists$ *s, t-Weg W in* N_f **do**
    Erhöhe f entlang von W;
end



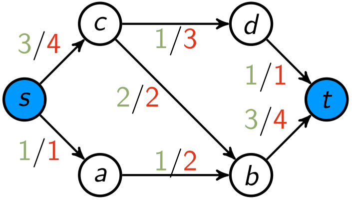
s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss



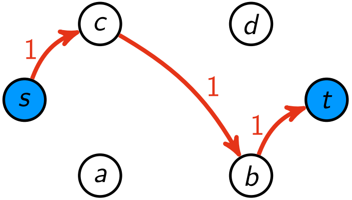
c_f Restkapazität

Input : Netzwerk
$N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
    $f(x, y) \leftarrow 0;$
end
while $\exists$ s, t-*Weg* W *in* N_f **do**
    Erhöhe f entlang von W;
end



s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss



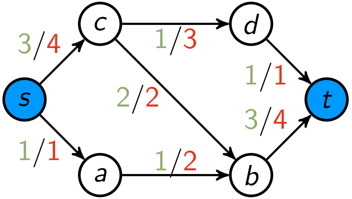
c_f Restkapazität

Input : Netzwerk $N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

---

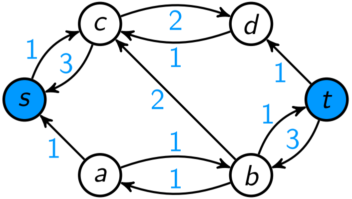
for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0$;
end
while $\exists$ *s, t-Weg W in N_f* **do**
  Erhöhe f entlang von W;
end



s Quelle, t Senke

c Kapazitätsobergrenze

f Fluss



c_f Restkapazität

Input : Netzwerk

$N = (V, A, s, t, c)$

Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**

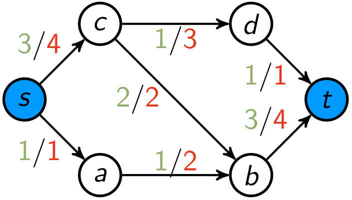
  $f(x, y) \leftarrow 0;$

end

while $\exists$ s, t-*Weg* W *in* N_f **do**

  Erhöhe f entlang von W;

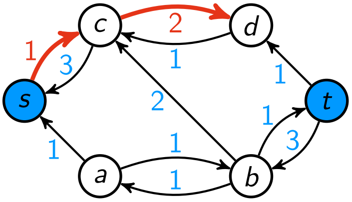
end



s Quelle, t Senke

c Kapazitätsobergrenze

f Fluss

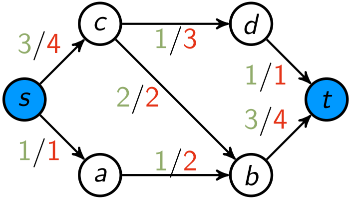


c_f Restkapazität

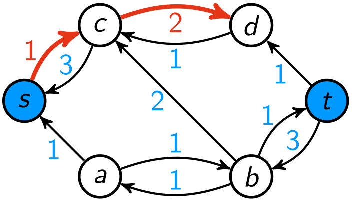
Input : Netzwerk
$N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
  $f(x, y) \leftarrow 0$;
end
while $\exists$ s, t-*Weg* W *in* N_f **do**
  Erhöhe f entlang von W;
end

$\Rightarrow \mathrm{val}(f) = 1 + 3 - 0 = 4$



s Quelle, t Senke
c Kapazitätsobergrenze
f Fluss



c_f Restkapazität

Algorithmus von Ford und Fulkerson

Input : Netzwerk $N = (V, A, s, t, c)$
Output : Fluss f mit maximalem Wert in N

for $(x, y) \in A$ **do**
    $f(x, y) \leftarrow 0$;
end
while *Es gibt einen* s, t*-Weg* W *in* N_f **do**
    Erhöhe f entlang von W;
end