Algorithms and Probability

Week 12

Georg Hasebe — 16/05/2024

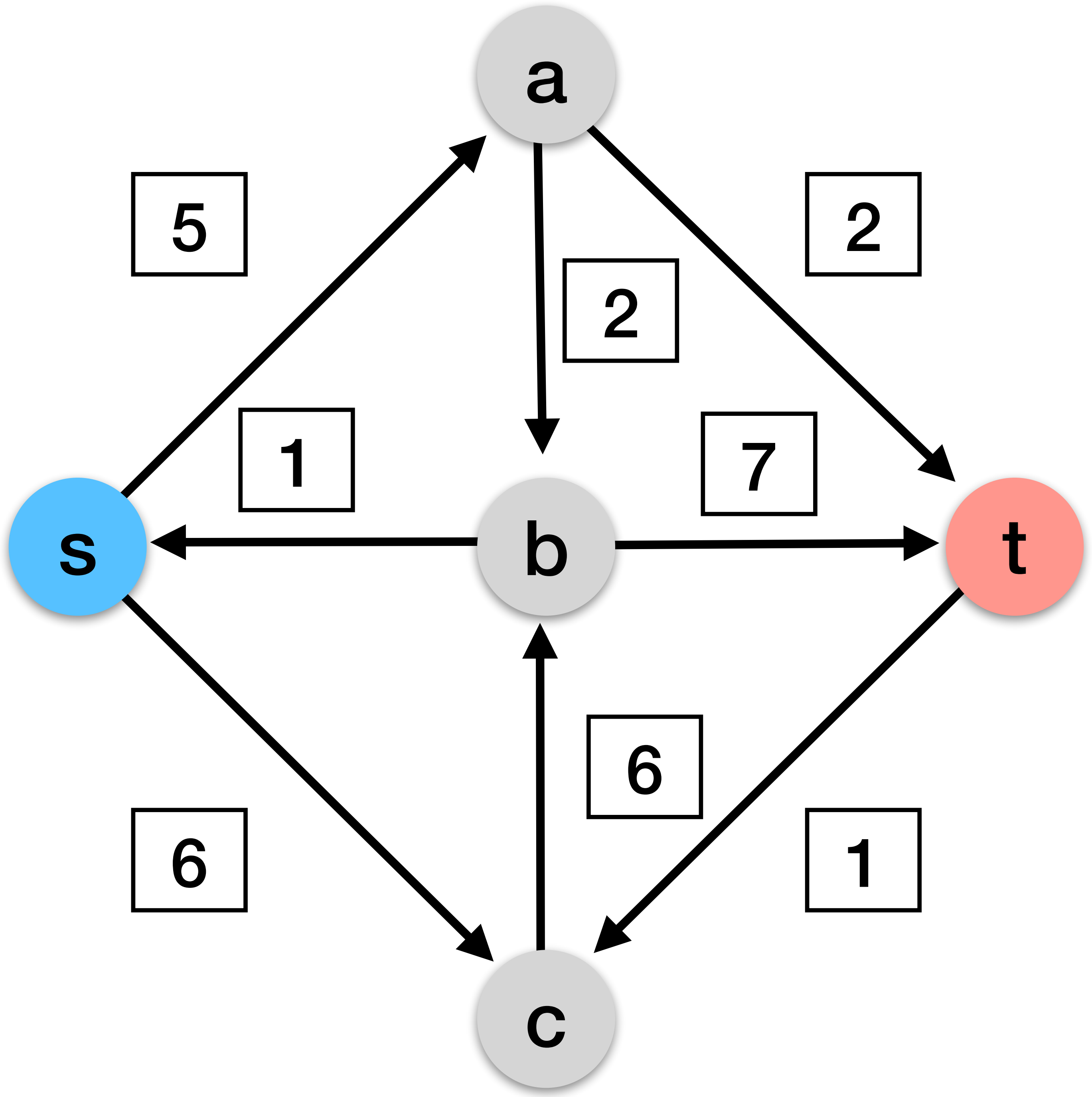
Theory Recap

Netzwerk

Ein **Netzwerk** ist ein Tupel $N = (V, A, c, s, t)$, wobei gilt:

- (V, A) ist ein gerichteter Graph (ohne Schleifen),
- $s \in V$, die **Quelle**,
- $t \in V \setminus \{s\}$, die **Senke**, und
- $c : A \to \mathbb{R}_0^+$, die **Kapazitätsfunktion**.

Netzwerk



Quelle (source)

Senke (sink)

Kapazität

Fluss

Sei $N = (V, A, c, s, t)$ ein Netzwerk. Ein **Fluss** in N ist eine Funktion $f : A \to \mathbb{R}$ mit den Bedingungen

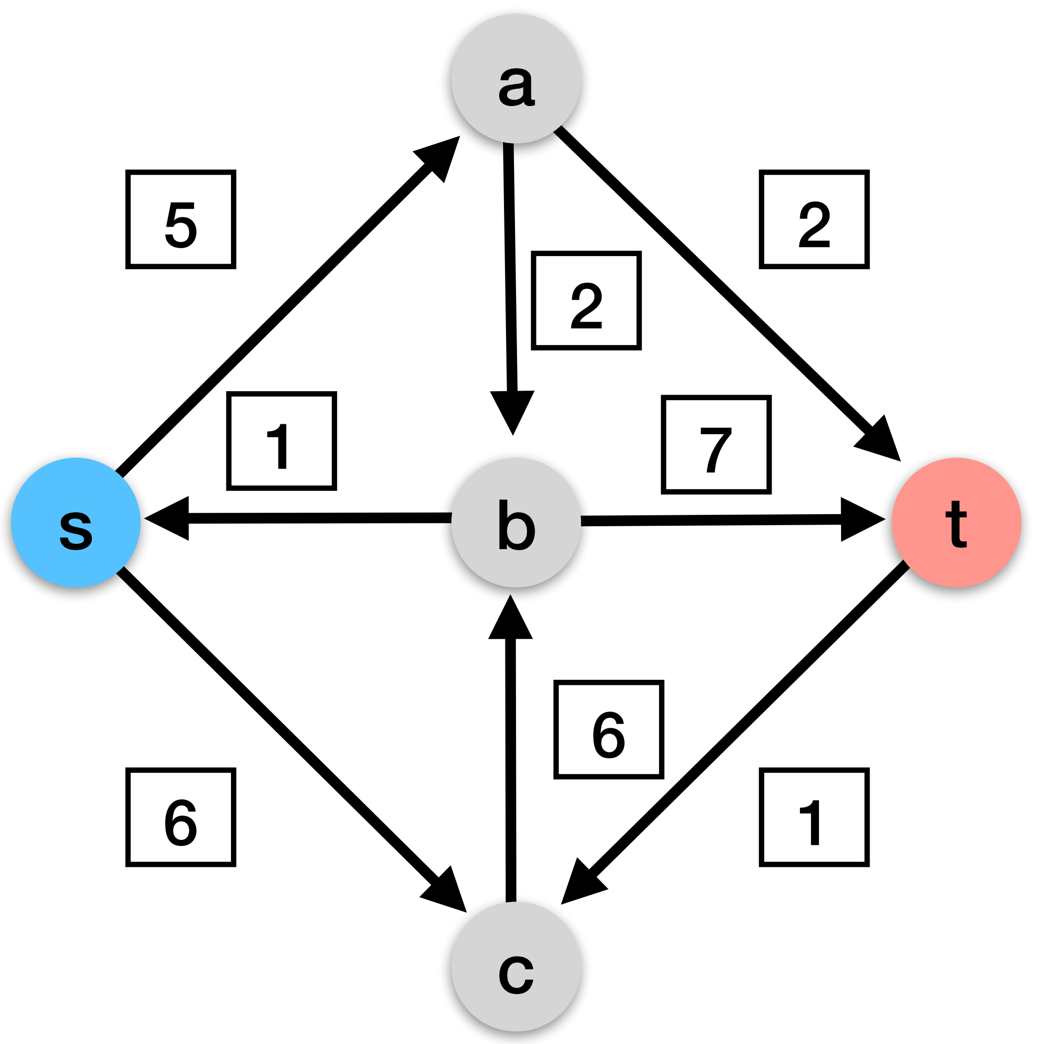
- **Zulässigkeit**: $0 \leq f(e) \leq c(e)$ für alle $e \in A$.
- **Flusserhaltung**: Für alle $v \in V \setminus \{s, t\}$ gilt

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

Der **Wert** eines Flusses f ist definiert als

$$\mathrm{val}(f) := \mathrm{netoutflow}(s) := \sum_{u \in V:\,(s,u) \in A} f(s, u) - \sum_{u \in V:\,(u,s) \in A} f(u, s) \ .$$

Fluss

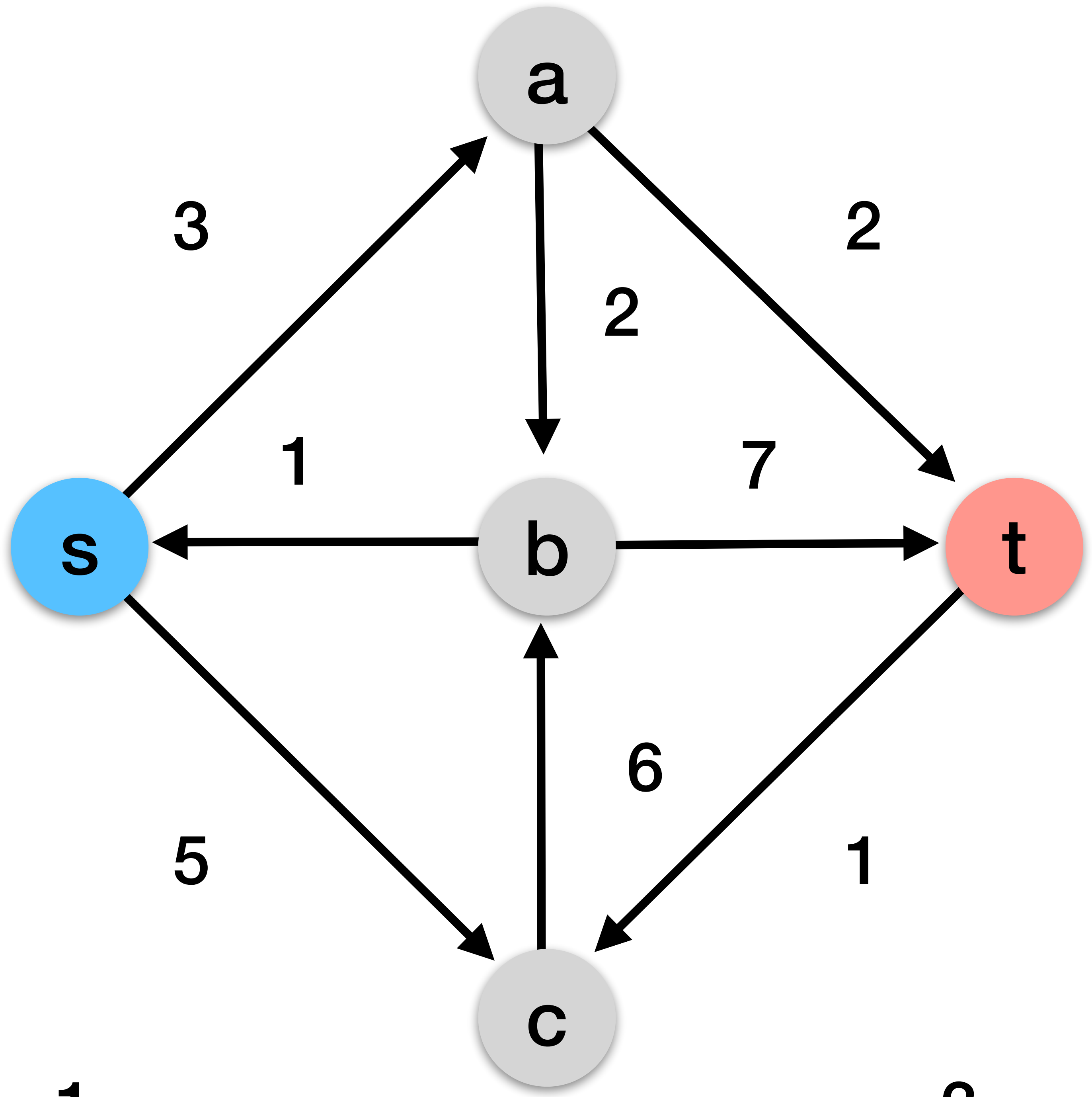


Quelle (source)

Senke (sink)

Kapazität

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



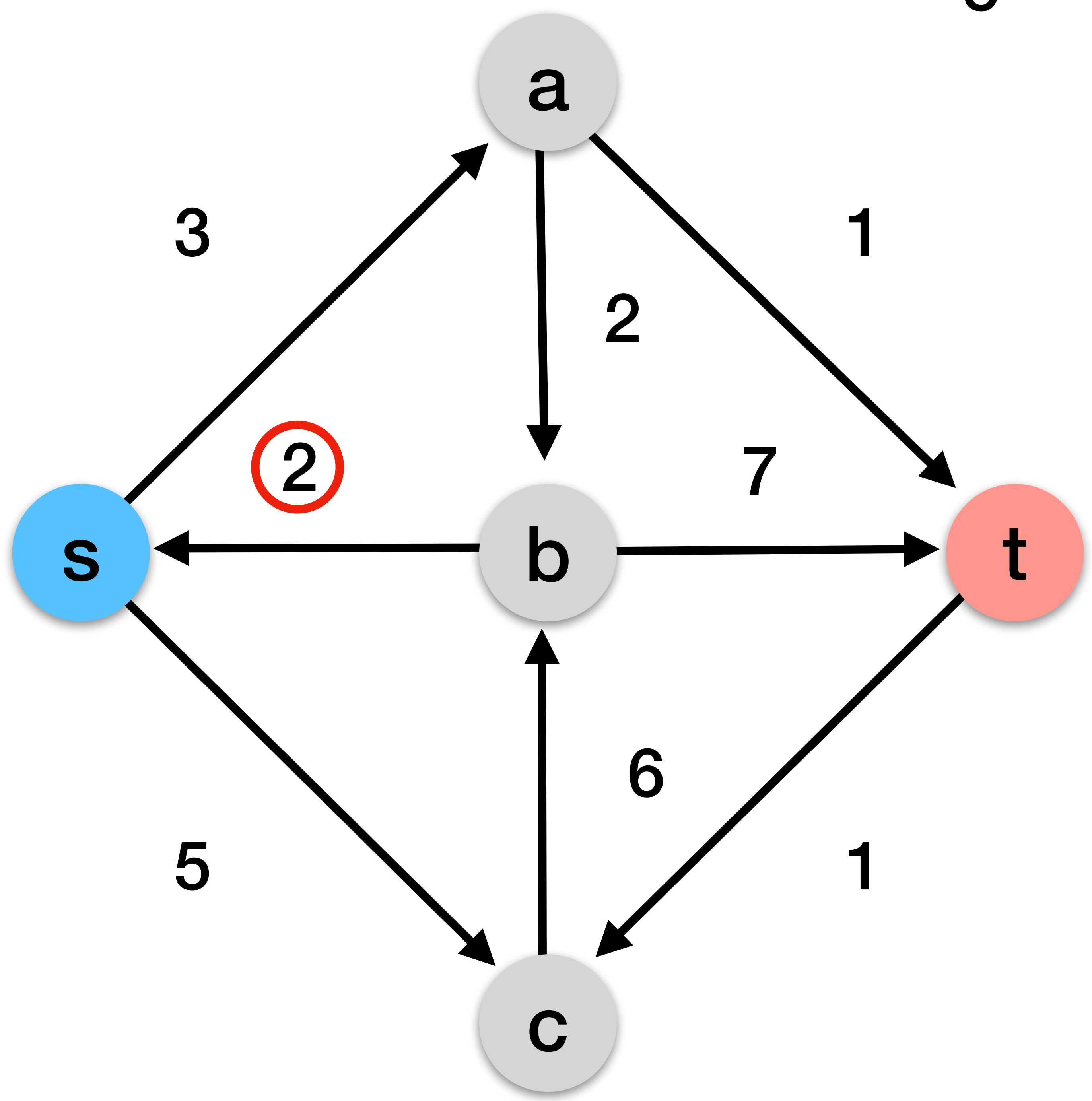
In Knoten a fliesst 3 hinein, aber 4 raus.

Regeln:

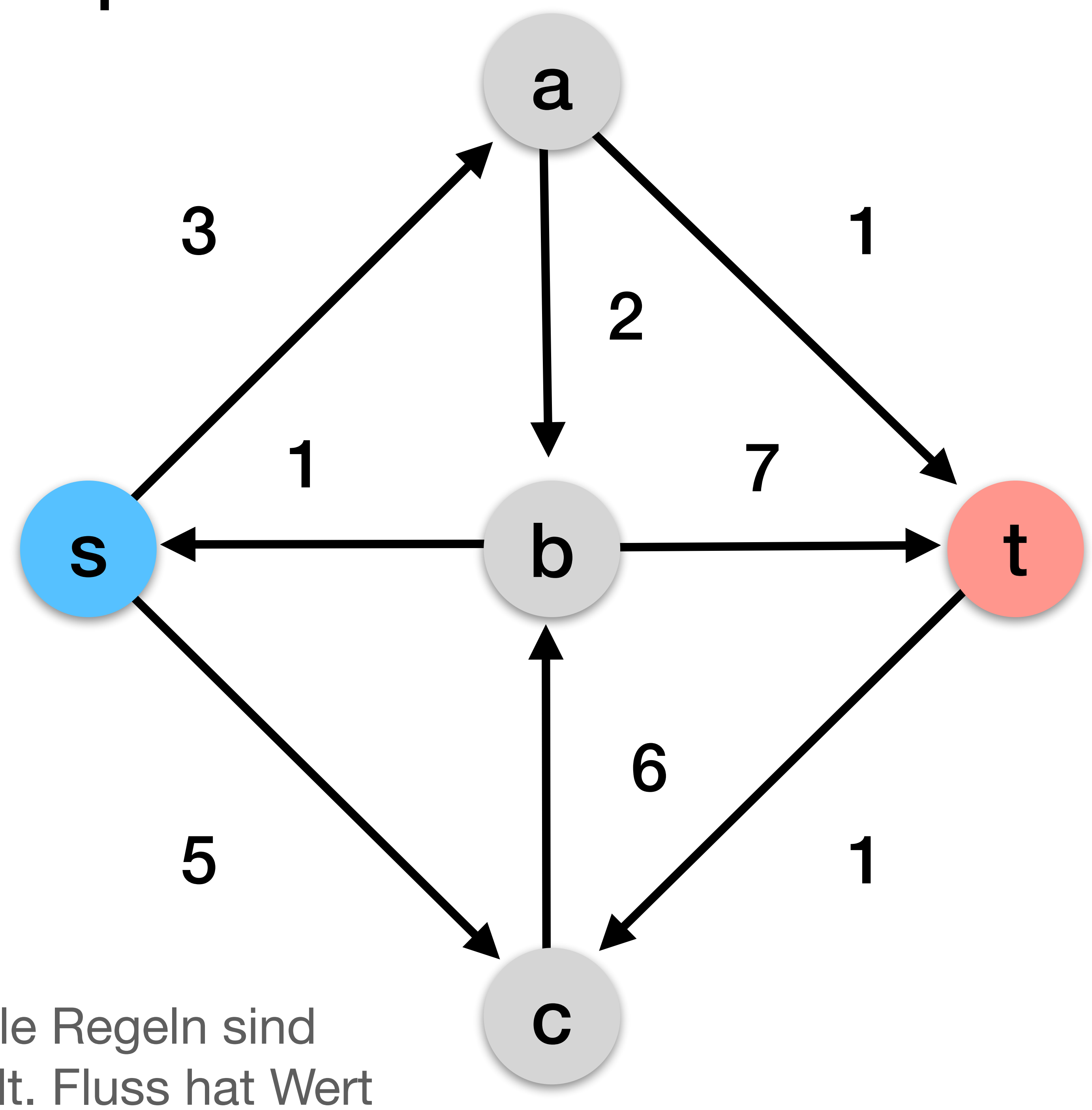
1. Zulässigkeit: $0 \leq f(e) \leq c(e)$ für alle $e \in A$.
2. Flusserhaltung: in v fliesst gleich viel raus wie rein für alle $v \in V \backslash \{s, t\}$

Flusswert/Netoutflow:

$$\text{val}(f) = \sum_{u \in V:\ (s,u) \in A} f(s, u) - \sum_{u \in V:\ (u,s) \in A} f(u, s)$$



Hier gilt $f(e) \nleq c(e)$.



Alle Regeln sind erfüllt. Fluss hat Wert $3 + 5 - 1 = 7$.

Nettozufluss

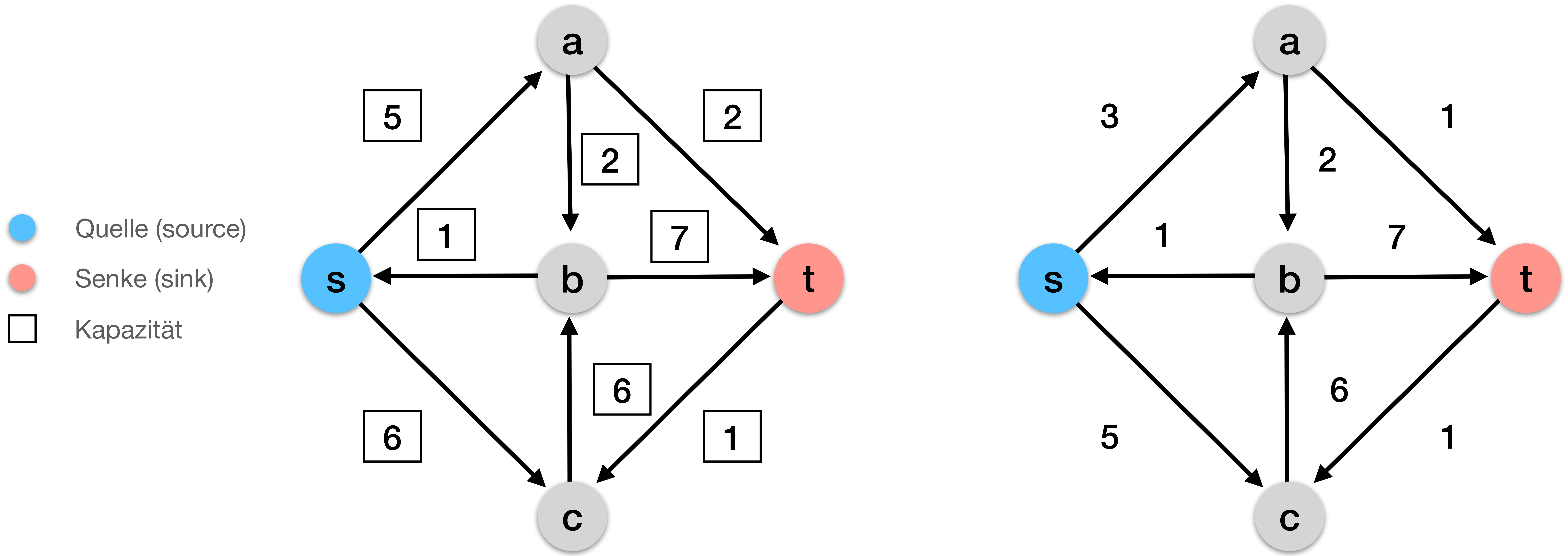
Lemma

*Der **Nettozufluss** der Senke t gleicht dem Wert des Flusses, d.h.*

$$\text{netinflow}(t) := \sum_{u \in V:\,(u,t) \in A} f(u,t) - \sum_{u \in V:\,(t,u) \in A} f(t,u) \;=\; \text{val}(f)\,.$$

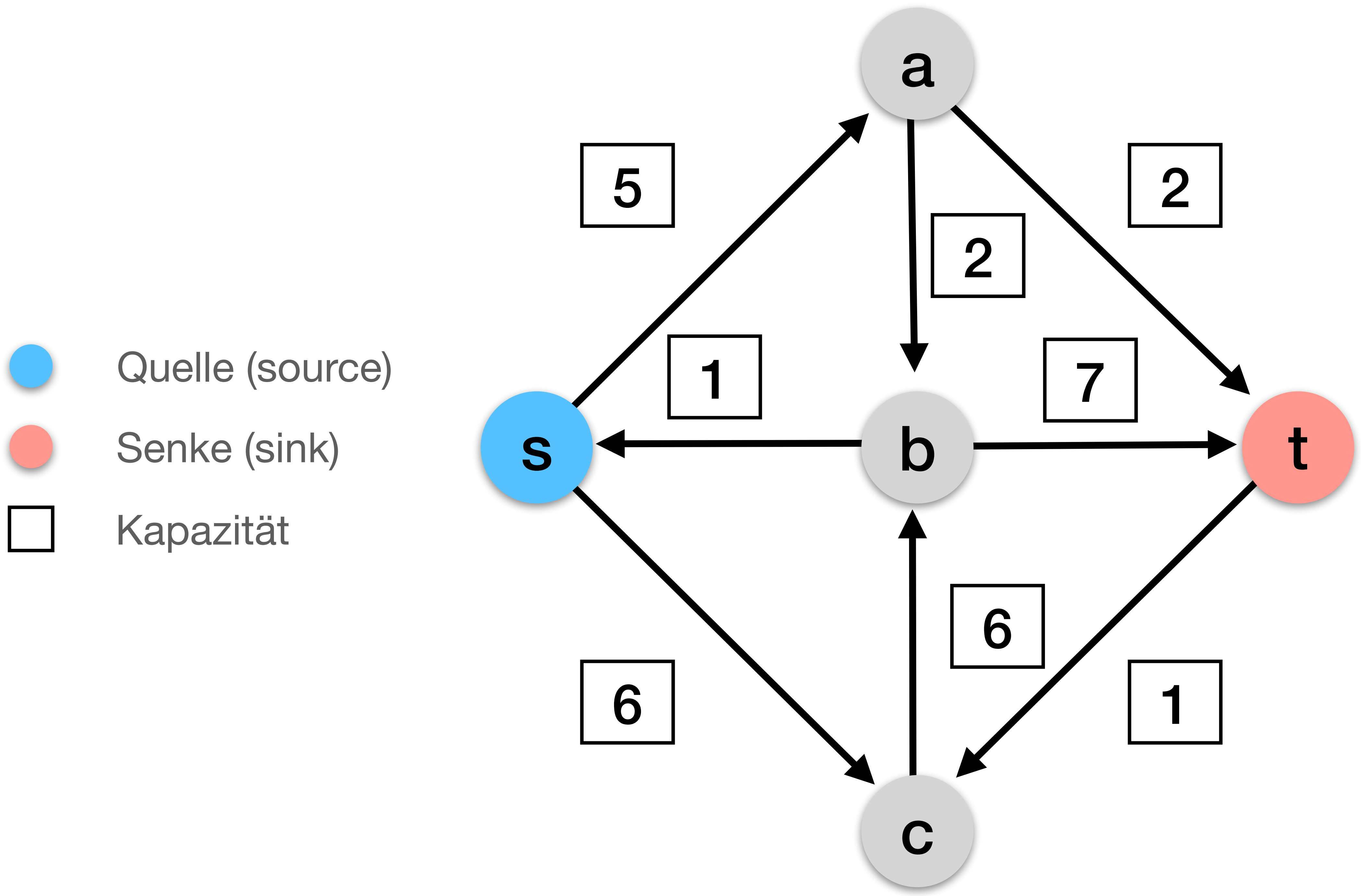
- $\text{val}(f) = \text{netoutflow}(s) = 3 + 5 - 1 = 7.$
- $\text{netinflow}(t) = 1 + 7 - 1 = 7.$

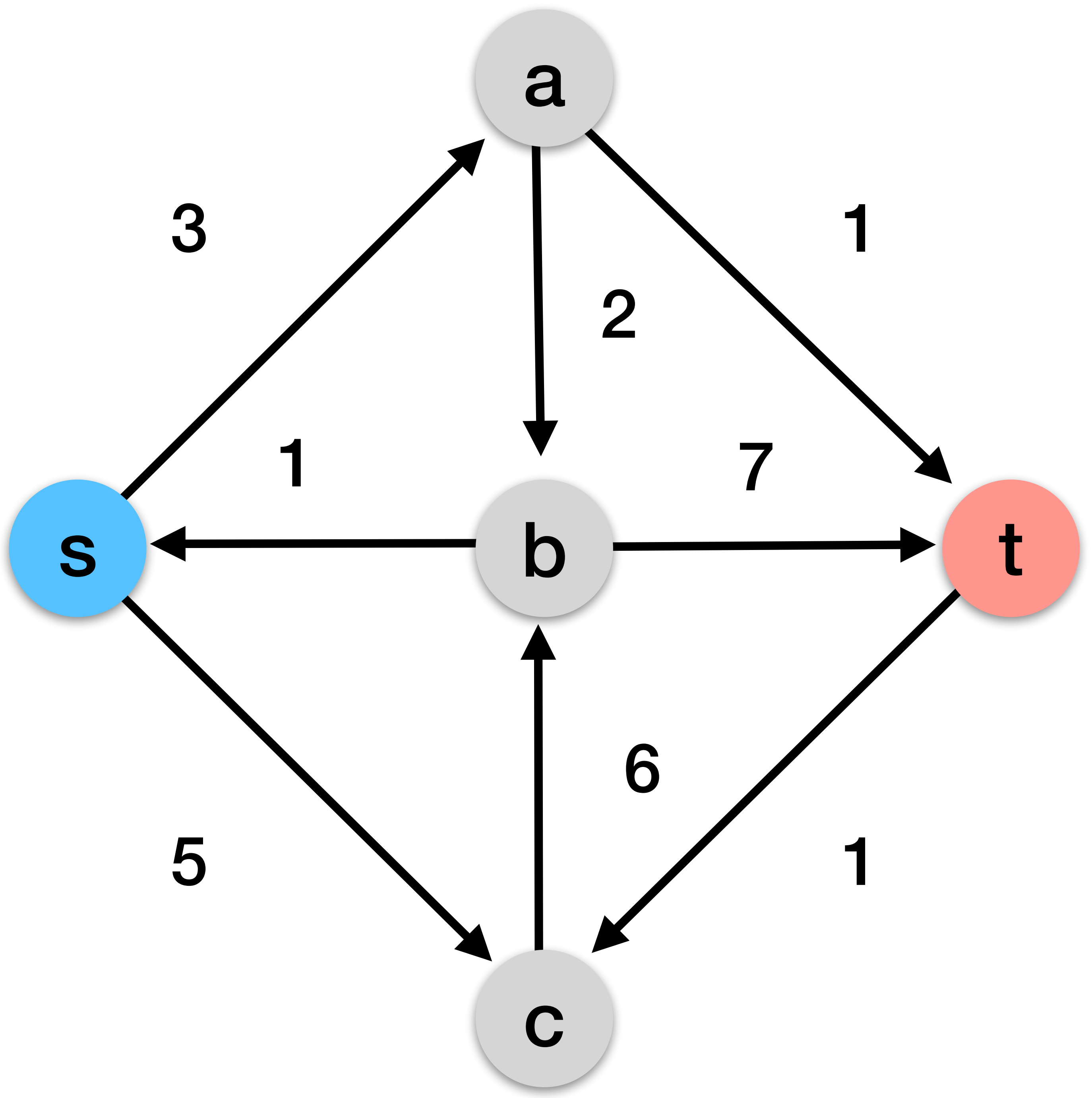
Wir sehen: $\text{netinflow}(t) = \text{val}(f)$.



Ziel

- Wir wissen nun was ein **Netwerk** und ein **Fluss** eines Netzwerkes ist.
- Wir wollen nun einen **maximalen Fluss** effizient finden. Aber wie?
- **Idee:** betrachte **Schnitte**.





Ist f maximal?

Fluss f mit Wert $3 + 5 - 1 = 7$.

Es gilt also $S \cap T = \varnothing$.

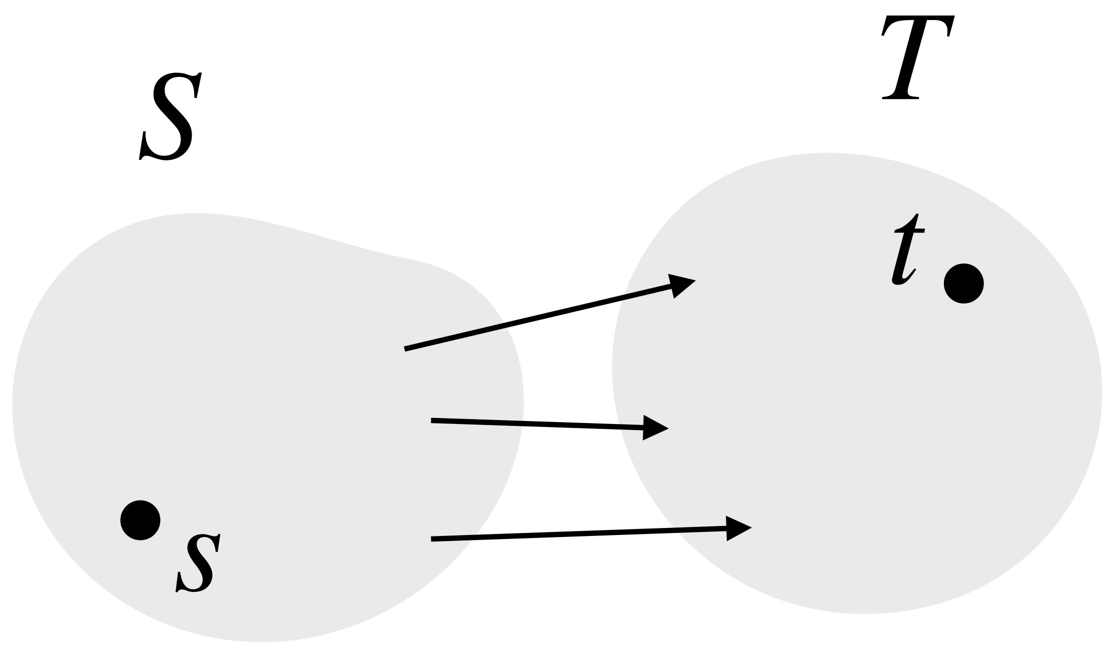
Schnitt

Ein **s-t-Schnitt** für ein Netzwerk (V, A, c, s, t) ist eine Partition (S, T) von V mit $s \in S$ und $t \in T$. Die **Kapazität** eines s-t-Schnitts (S, T) ist durch

$$\operatorname{cap}(S, T) := \sum_{(u,w) \in (S \times T) \cap A} c(u, w)$$

definiert. (Partition (S, T): $S \cup T = V$ und $S \cap T = \emptyset$)

$G = (V, A)$



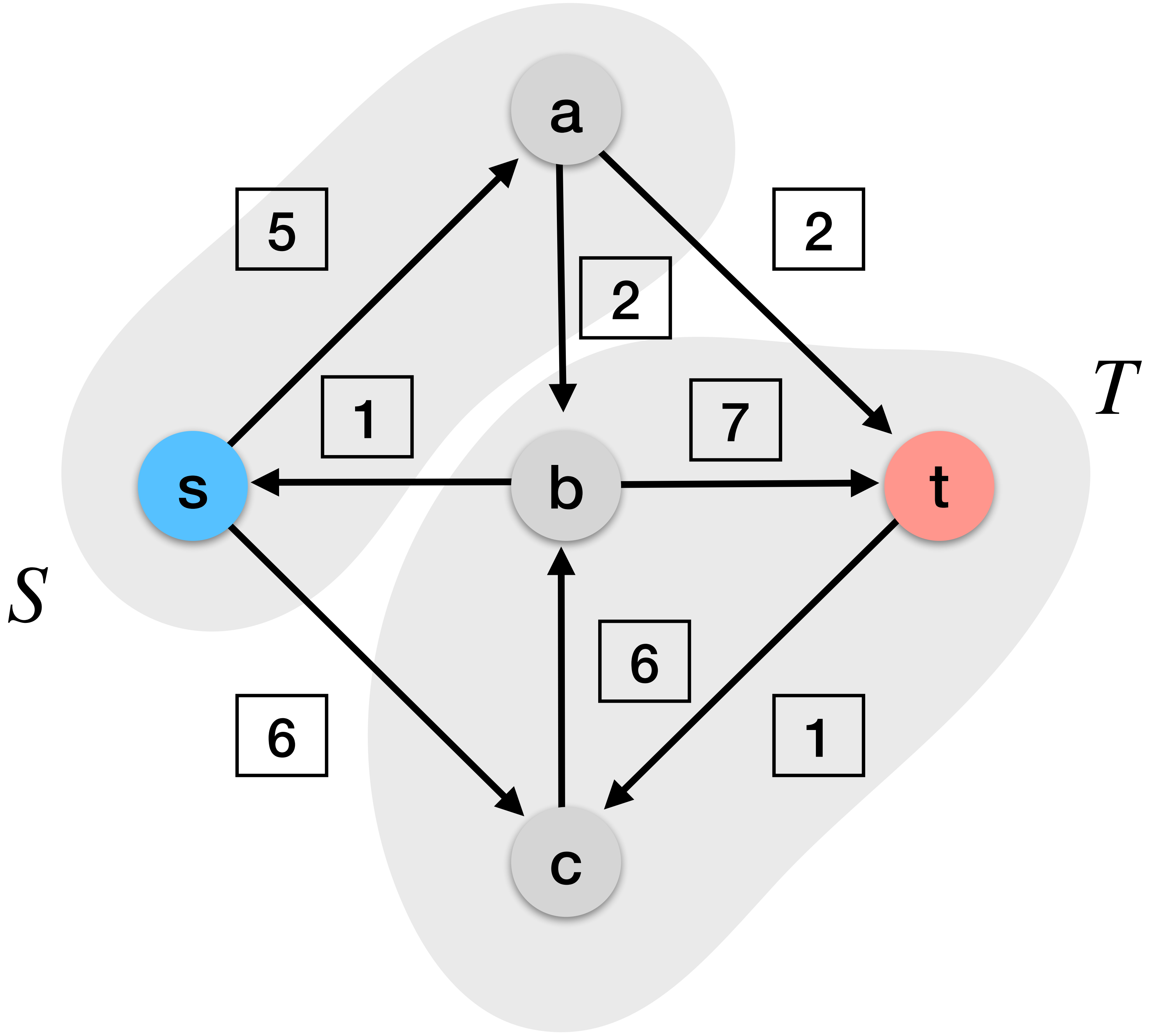
Wir ignorieren die Kanten von T nach S!

Schnitt

Quelle (source)

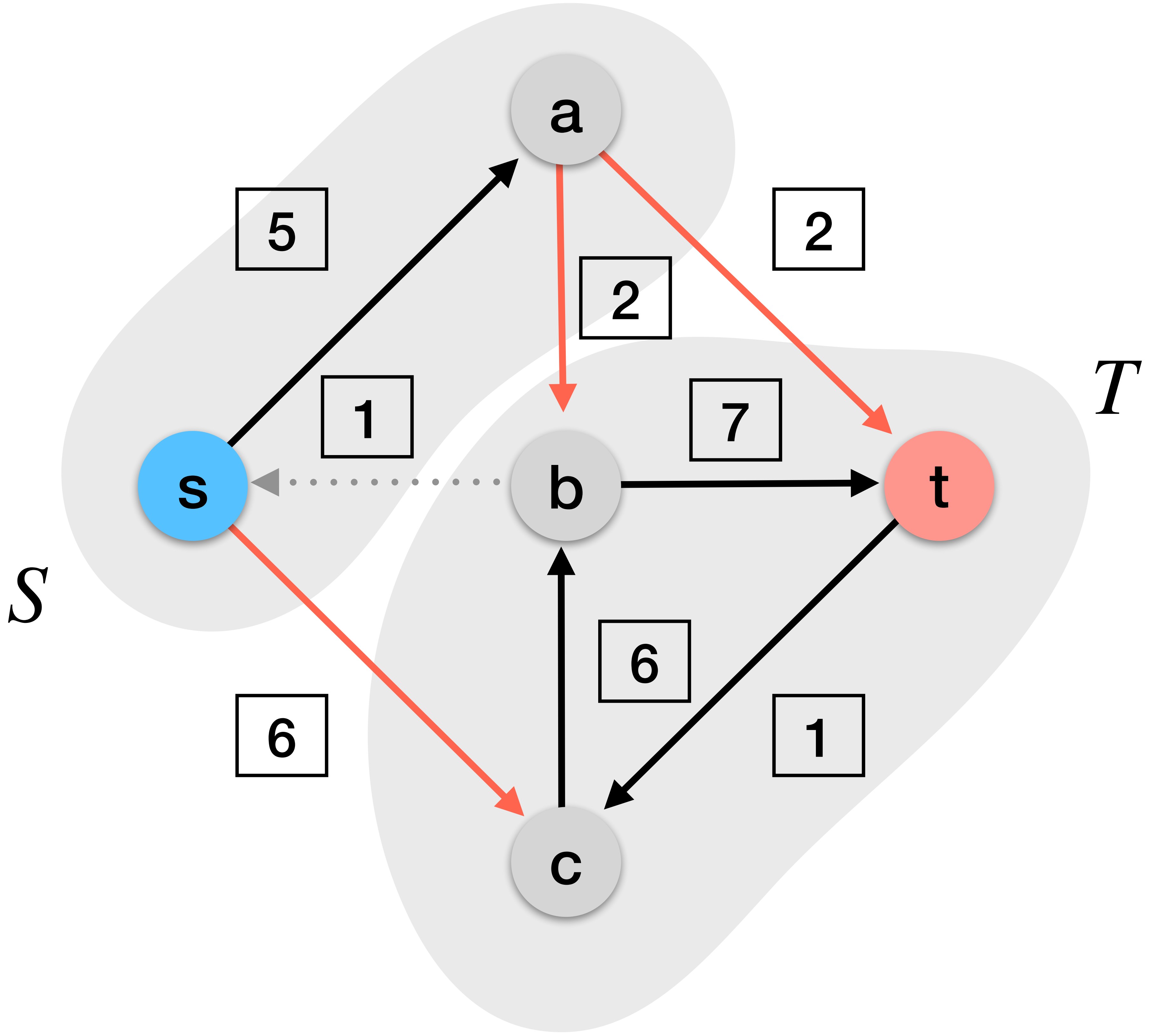
Senke (sink)

Kapazität



Was ist also $\mathrm{cap}(S,T) = \sum_{u,w \in (S \times T) \cap A} c(u,w)$?

Schnitt



Die Kante von b nach s geht zwar über den Schnitt, aber von T nach S und wird deswegen **nicht mitgezählt**!

$$\text{cap}(S, T) = \sum_{u,w \in (S \times T) \cap A} c(u, w) = 2 + 2 + 6 = 10.$$

Lemma

Ist f ein Fluss und (S, T) ein s-t-Schnitt in einem Netzwerk (V, A, c, s, t), so gilt

$$\mathrm{val}(f) \leq \mathrm{cap}(S, T)\ .$$

Ein Fluss kann nie grösser sein als die Kapazität eines s-t-Schnitts.

Finden wir zu einem Fluss f einen s-t-Schnitt (S, T) mit $\mathrm{cap}(S, T) = \mathrm{val}(f)$, so ist f ein maximaler Fluss.

Der Schnitt (S, T) is ein einfacher Beweis (ein einfaches Zertifikat) für die Maximalität von f.

Zwischenergebnisse

- Schnitte ermöglichen eine Abschätzung des Flusswertes nach oben, da

$$\text{val}(f) \leq \text{cap}(S, T)\,.$$

- Finden wir also einen Fluss f sodass $\text{val}(f) = \text{cap}(S, T)$, dann ist f maximal.

Verbleibende Fragen:

- Gibt es immer einen maximalen Fluss?
- Gibt es immer einen minimalen Schnitt sodass $\text{val}(f) = \text{cap}(S, T)$?
- Wie bestimmen wir Flüsse/Schnitte effizient?

Maxflow-Mincut Theorem

Satz („Maxflow-Mincut Theorem")

Jedes Netzwerk $N = (V, A, c, s, t)$ *erfüllt*

$$\max_{f \text{ Fluss}} \mathrm{val}(f) = \min_{(S,T) \text{ s-t-Schnitt}} \mathrm{cap}(S, T)$$

Algorithmus-Idee

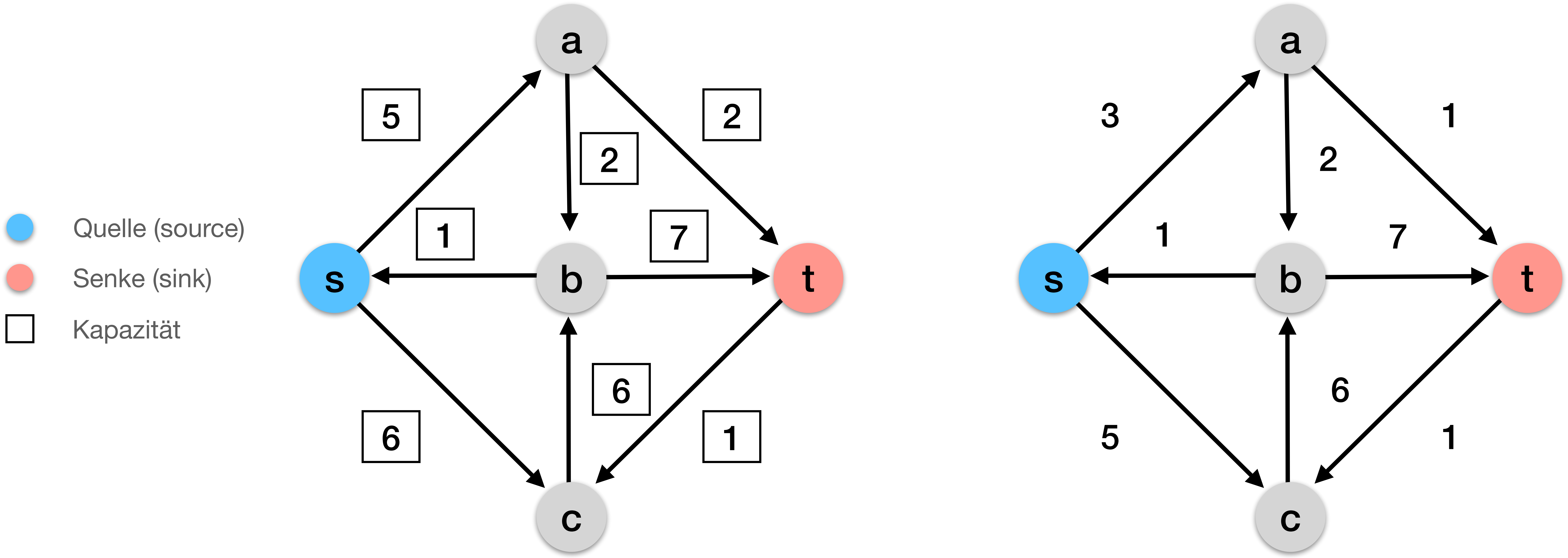
1. Wir starten mit einem Fluss mit Wert 0.
2. Wir erhöhen den Flusswert nach und nach.

Fragen:

- Wie erhöhen wir den Flusswert?
- Wie lange erhöhen wir?

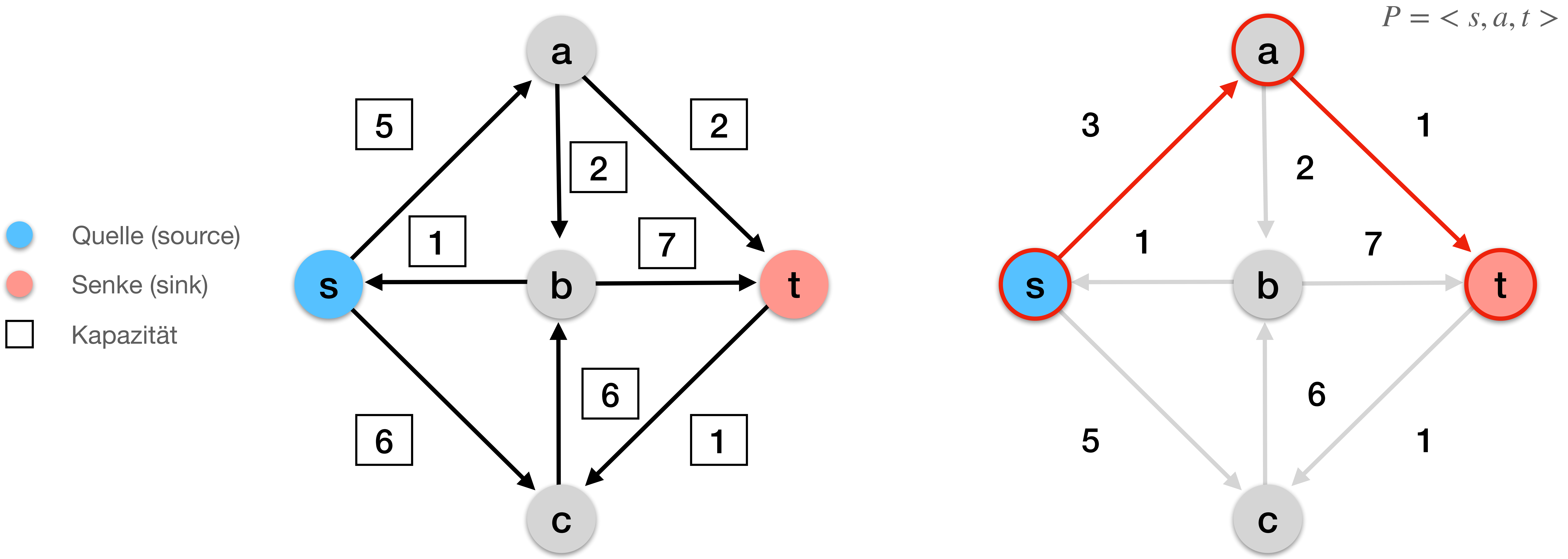
Flusswerterhöhung

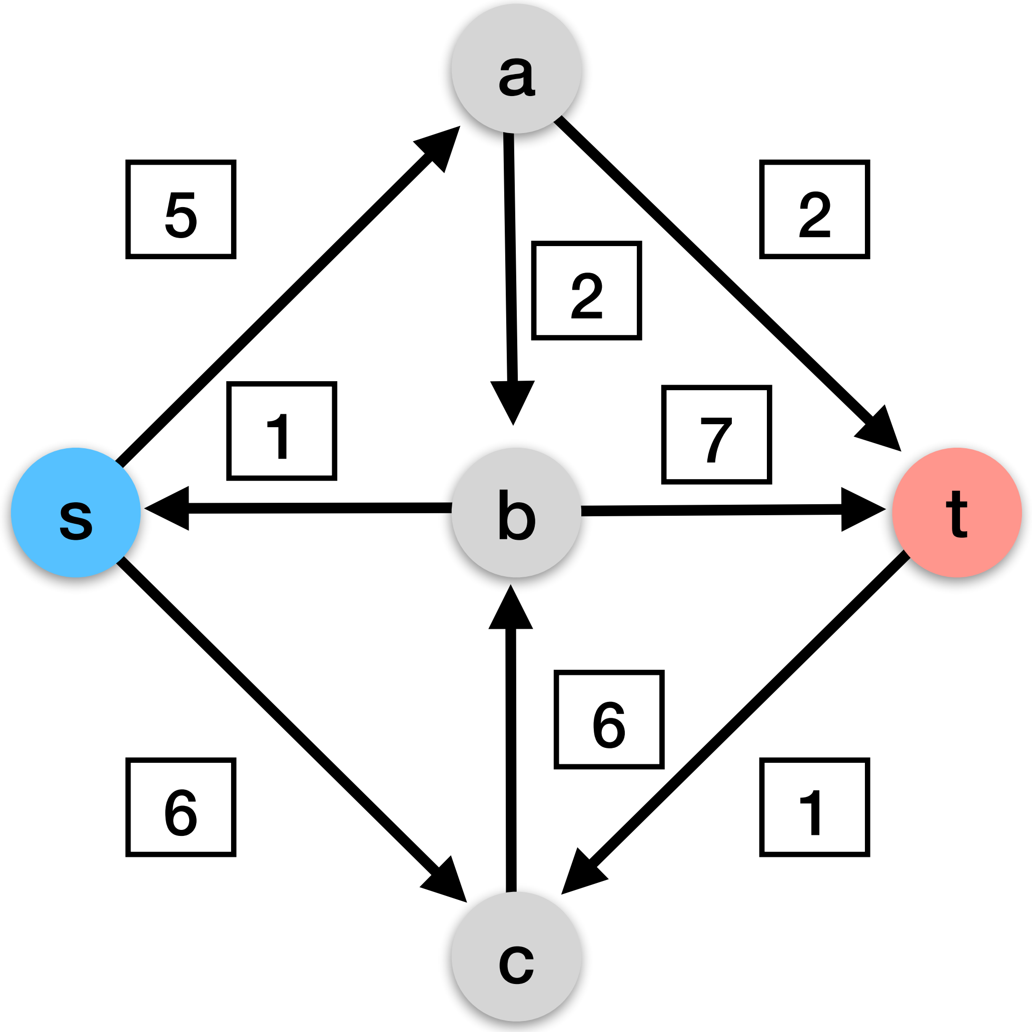
Angenommen, wir finden einen **gerichteten Pfad** P von s (Quelle) zu t (Senke), wo der Fluss auf allen Kanten die **Kapazität noch nicht erschöpft** hat, d.h. $f(e) < c(e)$ für alle e auf P.



Flusswerterhöhung

Angenommen, wir finden einen **gerichteten Pfad** P von s (Quelle) zu t (Senke), wo der Fluss auf allen Kanten die **Kapazität noch nicht erschöpft** hat, d.h. $f(e) < c(e)$ für alle e auf P.





Quelle (source)

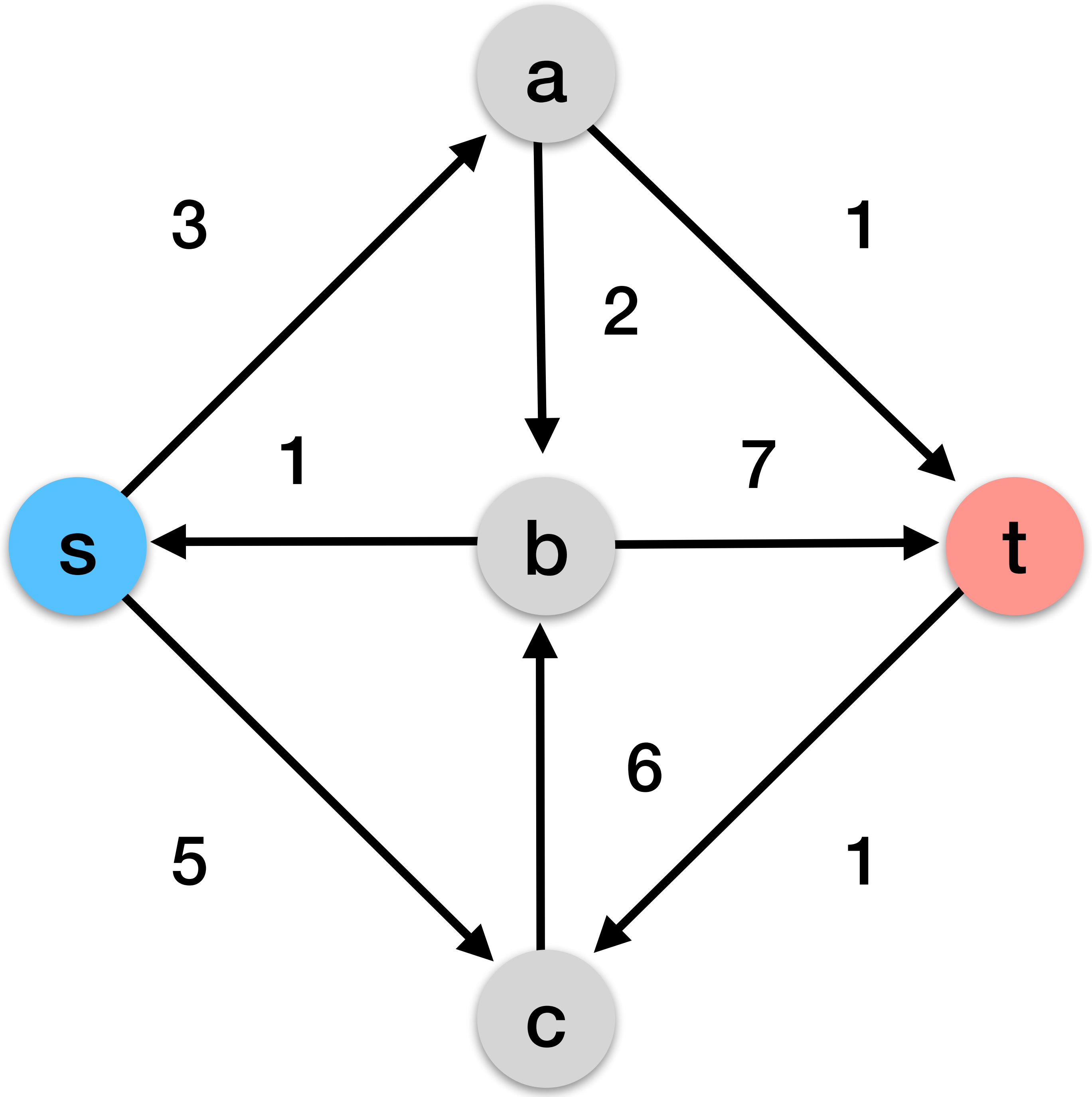
Senke (sink)

Kapazität

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

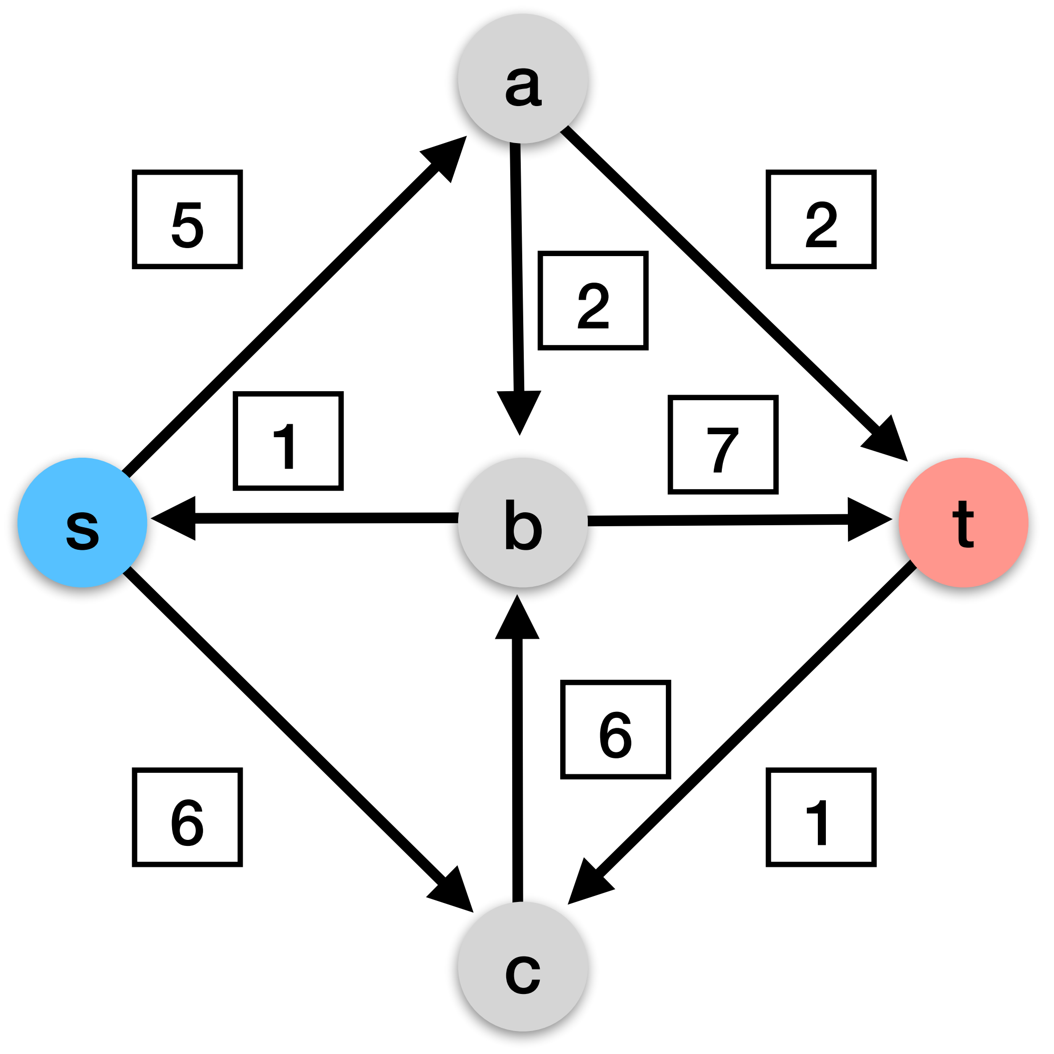
Flusswerterhöhung

Angenommen, wir finden einen **gerichteten Pfad** P von s (Quelle) zu t (Senke), wo der Fluss auf allen Kanten die **Kapazität noch nicht erschöpft** hat, d.h. $f(e) < c(e)$ für alle e auf P.



Wir definieren $\delta := \min_{e \in P} c(e) - f(e)$.

Flusswerterhöhung



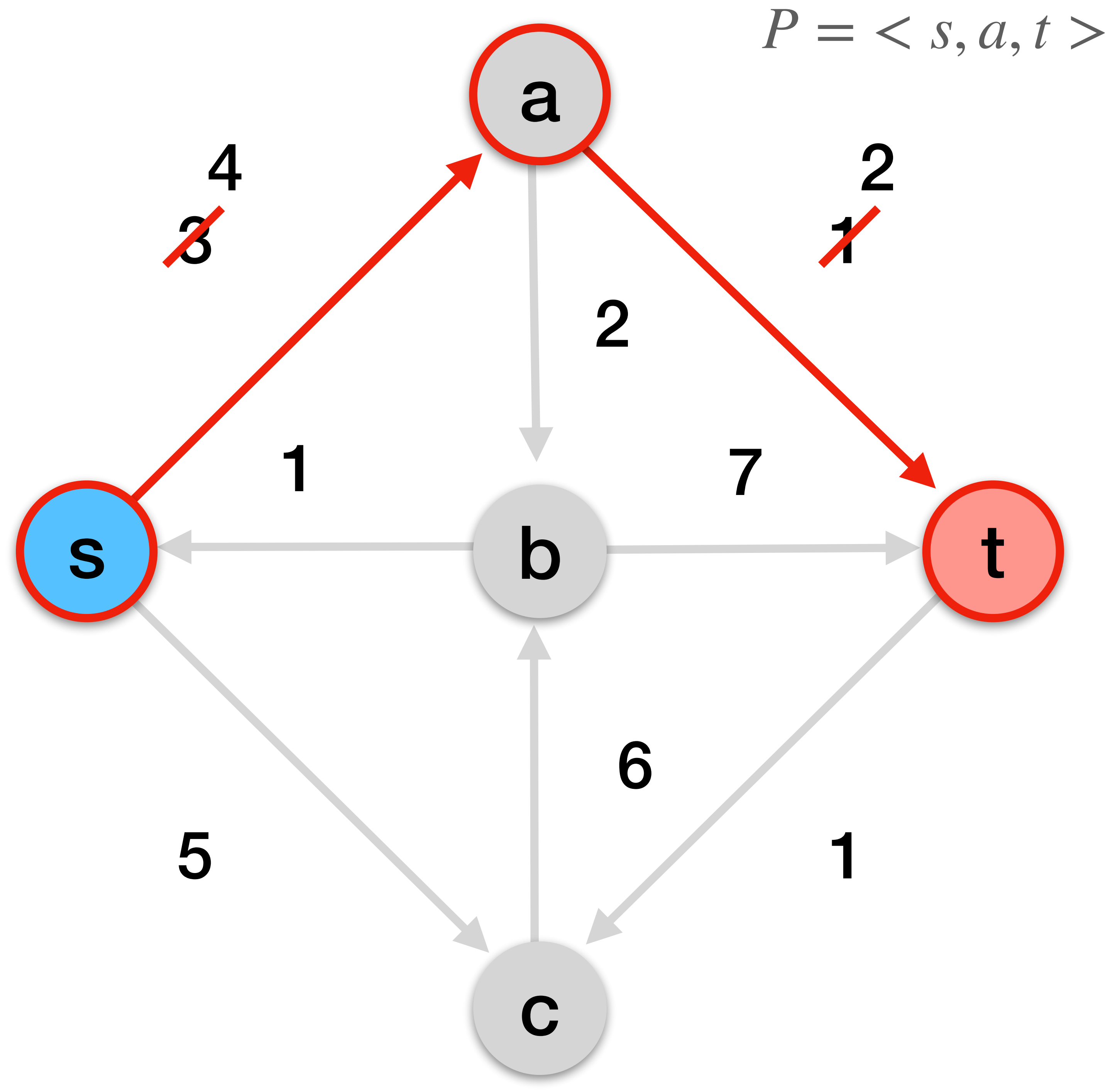
Quelle (source)

Senke (sink)

Kapazität

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

Angenommen, wir finden einen **gerichteten Pfad** P von s (Quelle) zu t (Senke), wo der Fluss auf allen Kanten die **Kapazität noch nicht erschöpft** hat, d.h. $f(e) < c(e)$ für alle e auf P.



Wir definieren $\delta := \min_{e \in P} c(e) - f(e)$.

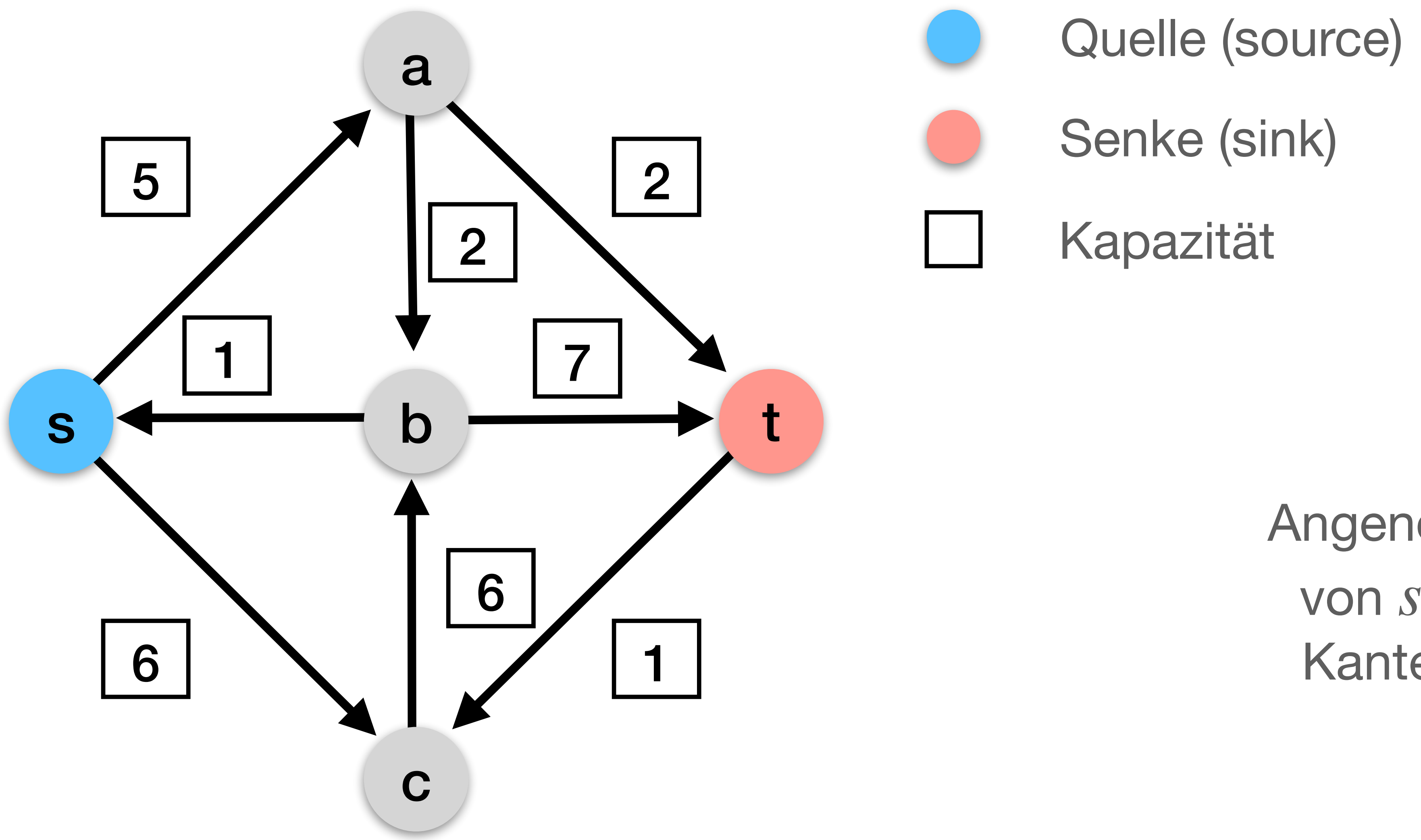
Hier: $\delta = 2 - 1 = 1$.

Erhöhe entlang P um $\delta = 1$.

1. Flusseigenschaft wurde nicht verletzt.
2. Flusswert wurde um δ erhöht.

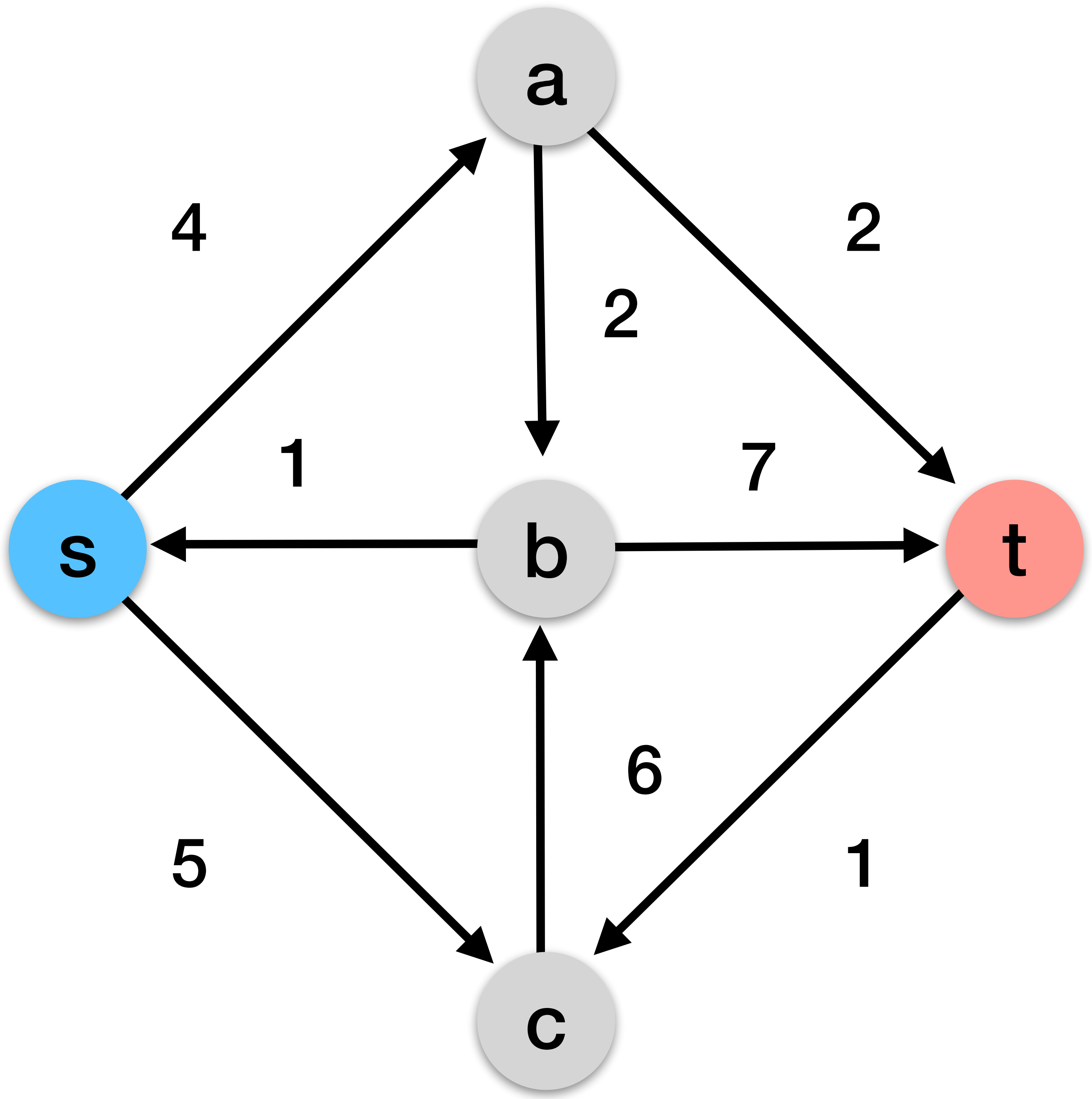
Wir haben einen Fluss mit Wert $4 + 5 - 1 = 8$.

Flusserhöhung



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

Angenommen, wir finden einen **gerichteten Pfad** P von s (Quelle) zu t (Senke), wo der Fluss auf allen Kanten die **Kapazität noch nicht erschöpft** hat, d.h. $f(e) < c(e)$ für alle e auf P.



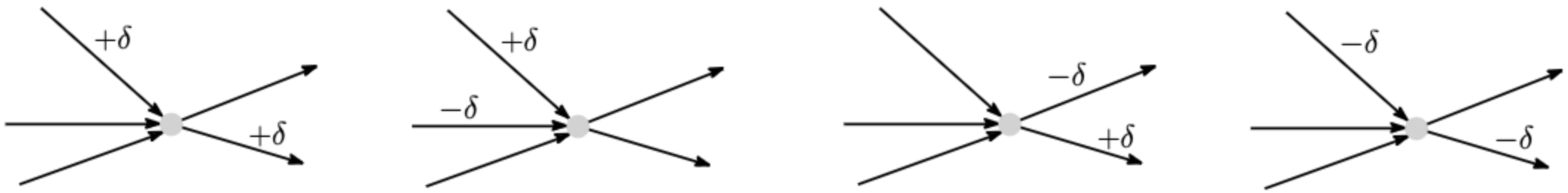
Fluss mit Wert $4 + 5 - 1 = 8$.

Man sieht schnell: es gibt keinen solchen Pfad P mehr.

Aber ist f maximal?

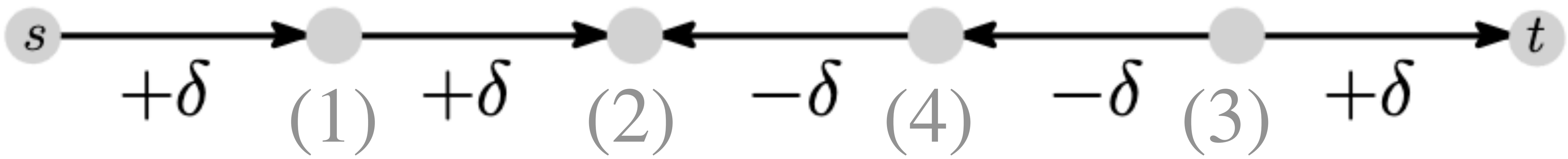
Nein! Warum?

Lokale Veränderungen des Flusses, die die Flusserhaltung erhalten:



(1) (2) (3) (4)

Quelle **Flussaugmentierung** Senke



augmentierender Pfad (ungerichteter Pfad!)

Erinnerung: augmentieren bedeutet so viel wie steigern, verstärken, erweitern.
Hier wird unser Flusswert gesteigert, denn wir haben 3-mal $+\delta$ und nur 2-mal $-\delta$, also insgesamt $+\delta$.

Restnetzwerk

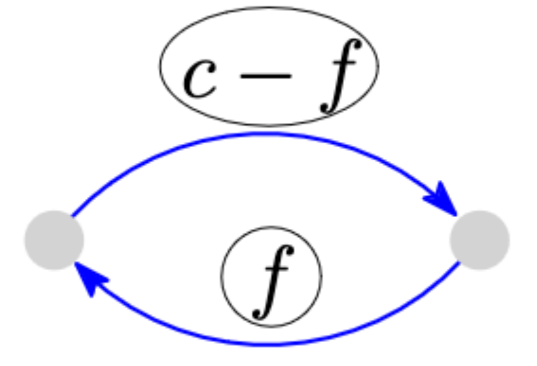
Für $e = (u, v)$, sei $e^{\text{opp}} := (v, u)$ (entgegen gerichtete Kante).

Sei $N = (V, A, c, s, t)$ ein Netzwerk ohne entgegen gerichtete Kanten[1] und sei f ein Fluss in N. Das **Restnetzwerk** $N_f := (V, A_f, r_f, s, t)$ ist wie folgt definiert:

1. Ist $e \in A$ mit $f(e) < c(e)$, dann ist e eine Kante in A_f, mit
$$r_f(e) := c(e) - f(e).$$
2. Ist $e \in A$ mit $f(e) > 0$, dann ist e^{opp} in A_f, mit
$$r_f(e^{\text{opp}}) = f(e).$$
3. A_f enthält nur Kanten wie in (1) und (2).

$r_f(e)$, $e \in A_f$, nennen wir die **Restkapazität** der Kante e.

Restkapazität = „Spielraum“



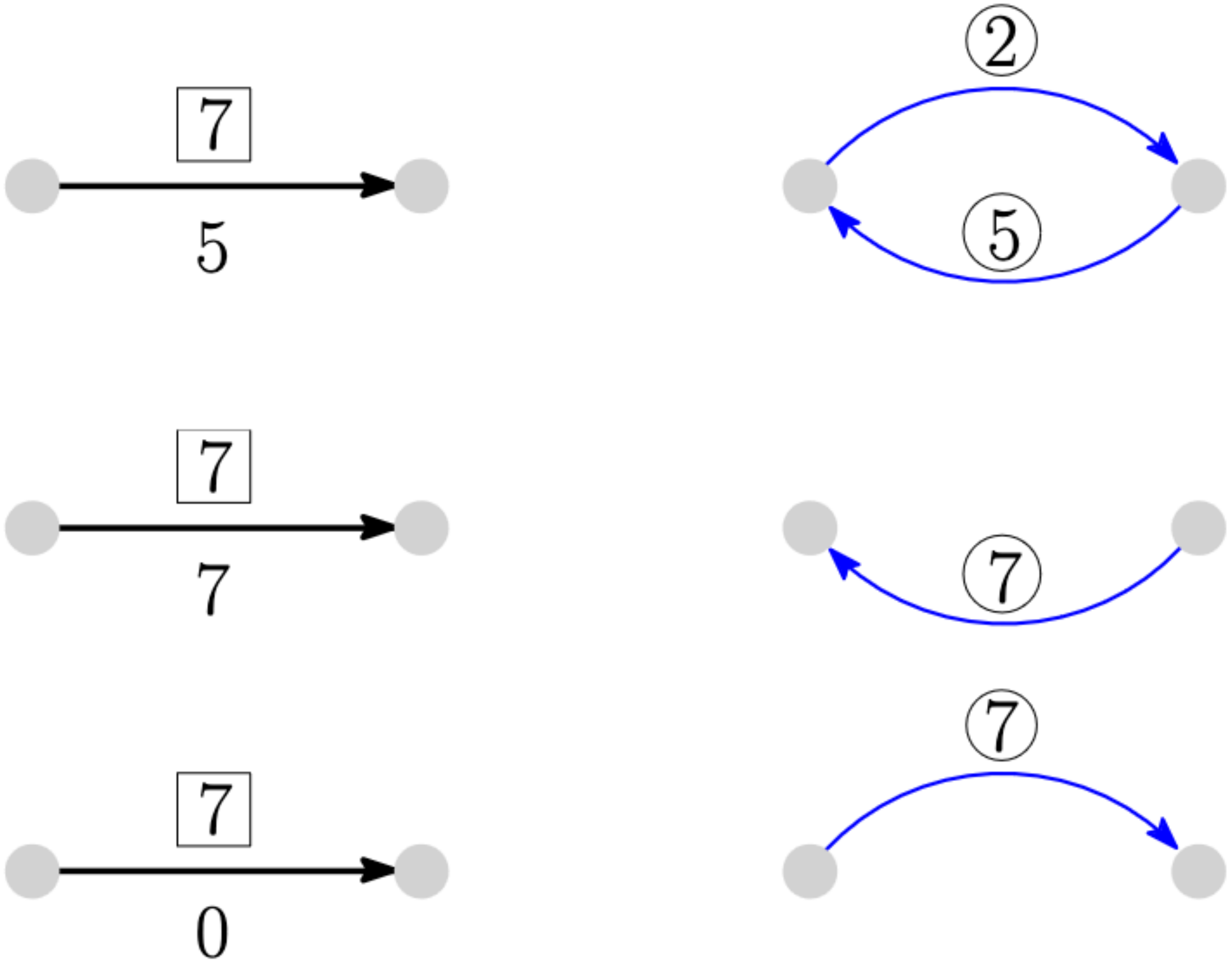
Spielraum

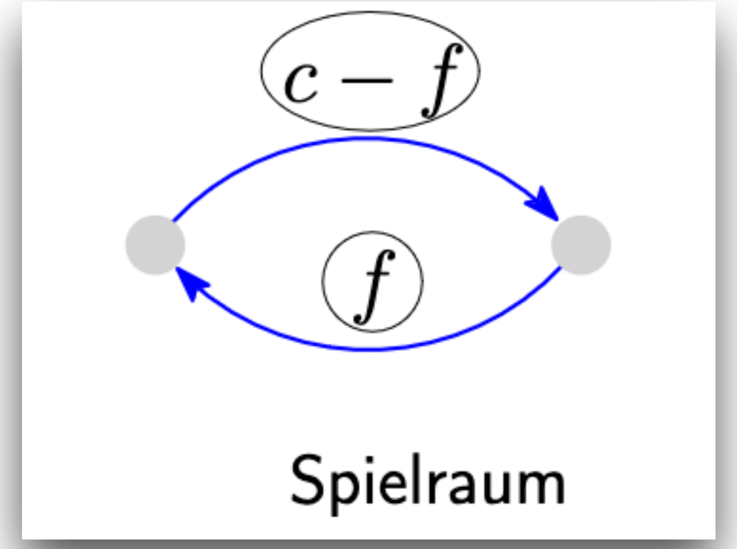
[1]: d.h. zwischen zwei Knoten gibt es nie zugleich die beiden Kanten beider Richtungen. Vereinfachende Annahme, ist aber nicht essentiell

Restnetzwerk

Netzwerk

Restnetzwerk
Restkapazität





1. Ist $e \in A$ mit $f(e) < c(e)$, dann ist e eine Kante in A_f, mit

$$r_f(e) := c(e) - f(e).$$

2. Ist $e \in A$ mit $f(e) > 0$, dann ist e^{opp} in A_f, mit

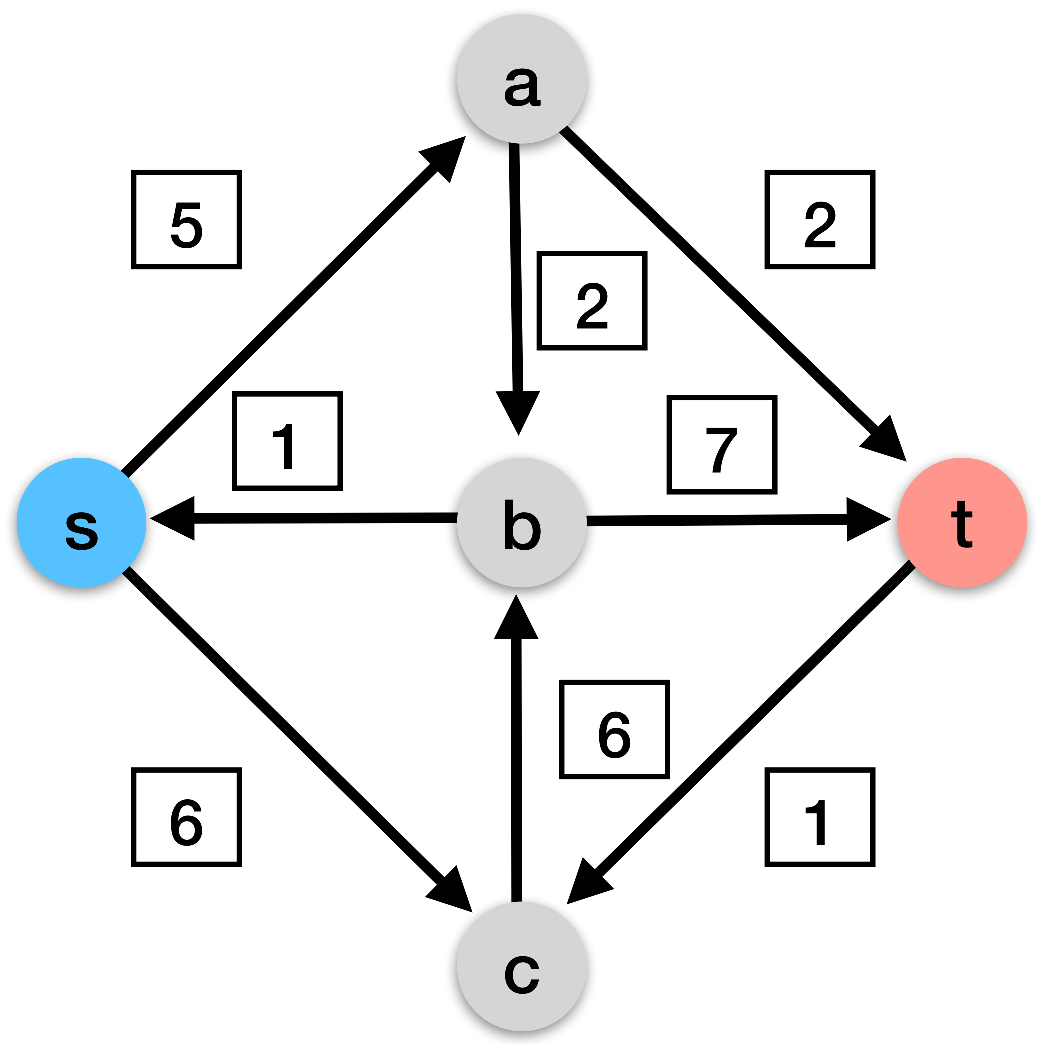
$$r_f(e^{\text{opp}}) = f(e).$$

Wegen strikt kleiner/grösser, gibt es in den beiden unteren Fällen nur jeweils eine Kante!

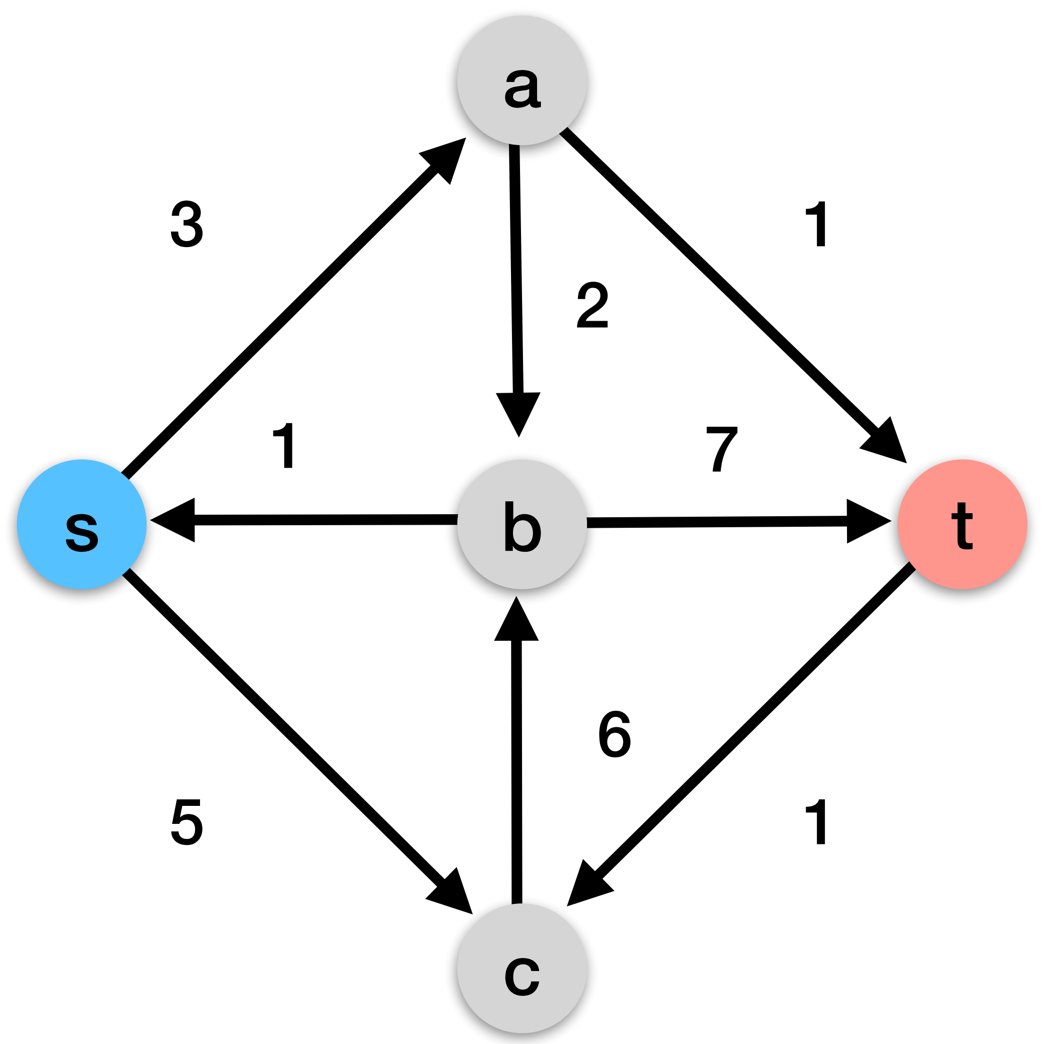
Quelle (source)

Senke (sink)

Kapazität

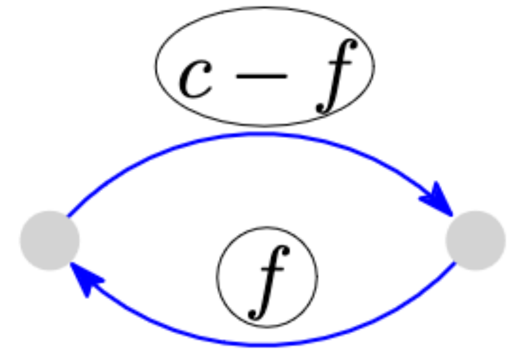


Netzwerk $N = (V, A, c, s, t)$.

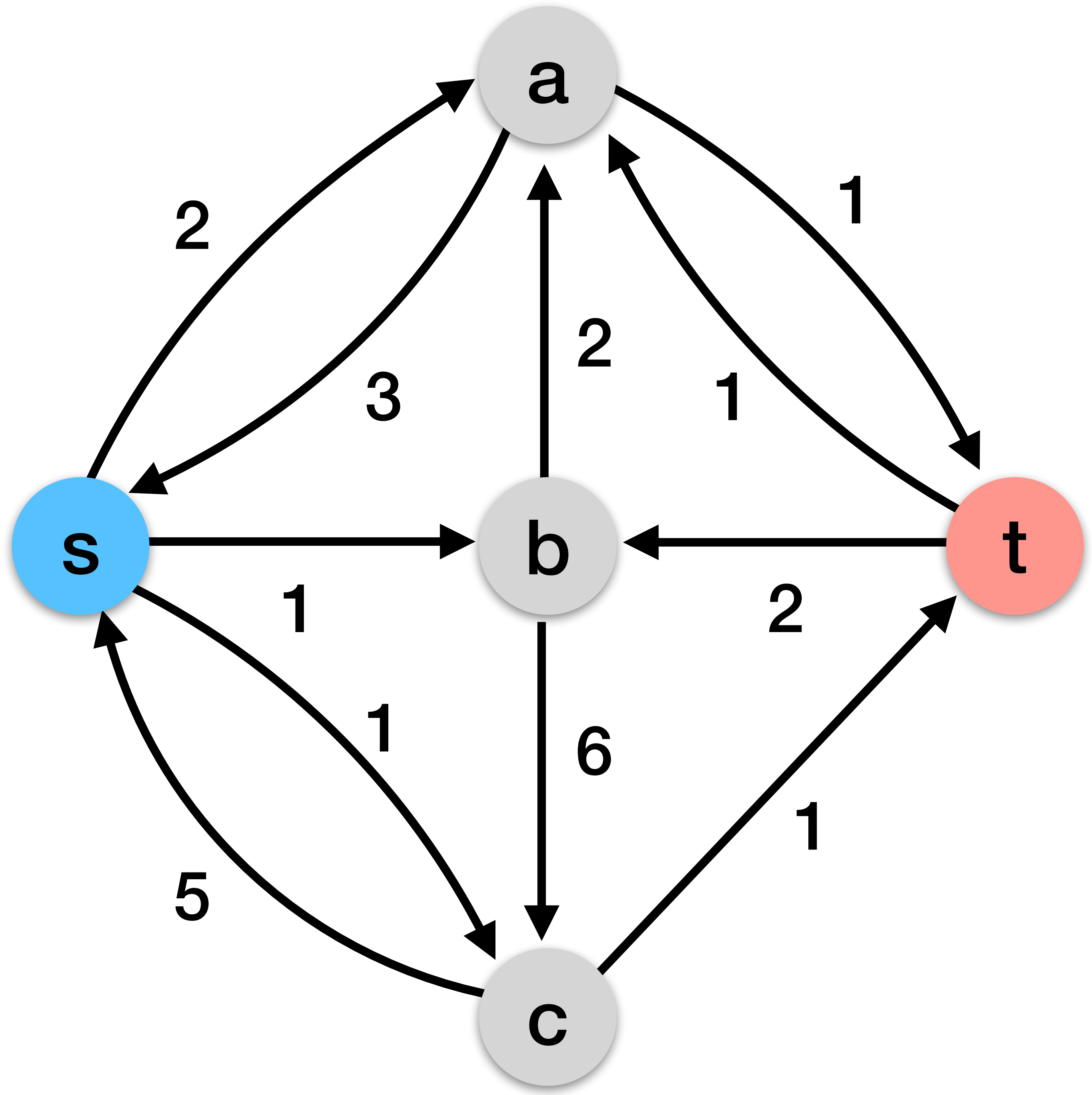


Fluss f mit Wert $3 + 5 - 1 = 7$.

1. Ist $e \in A$ mit $f(e) < c(e)$, dann ist e eine Kante in A_f, mit
$$r_f(e) := c(e) - f(e).$$
2. Ist $e \in A$ mit $f(e) > 0$, dann ist e^{opp} in A_f, mit
$$r_f(e^{\text{opp}}) = f(e).$$



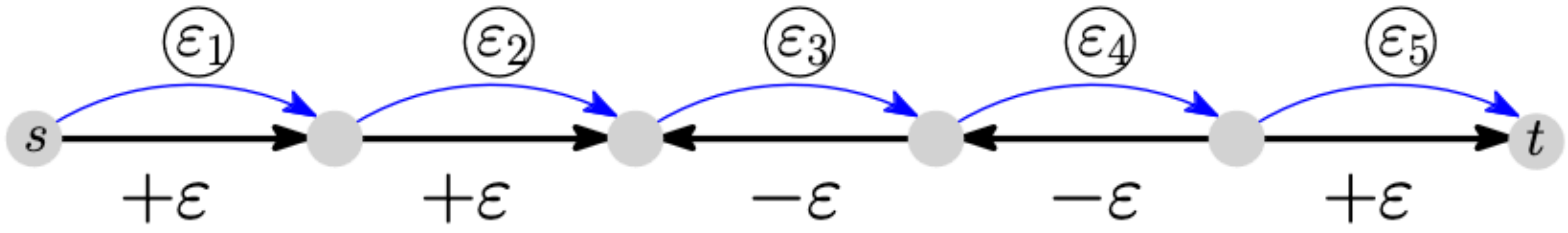
Spielraum



Restnetzwerk $N_f = (V, A_f, r_f, s, t)$.

Augmentierende Pfade

Wir betrachten einen gerichteten s-t-Pfad in N_f:

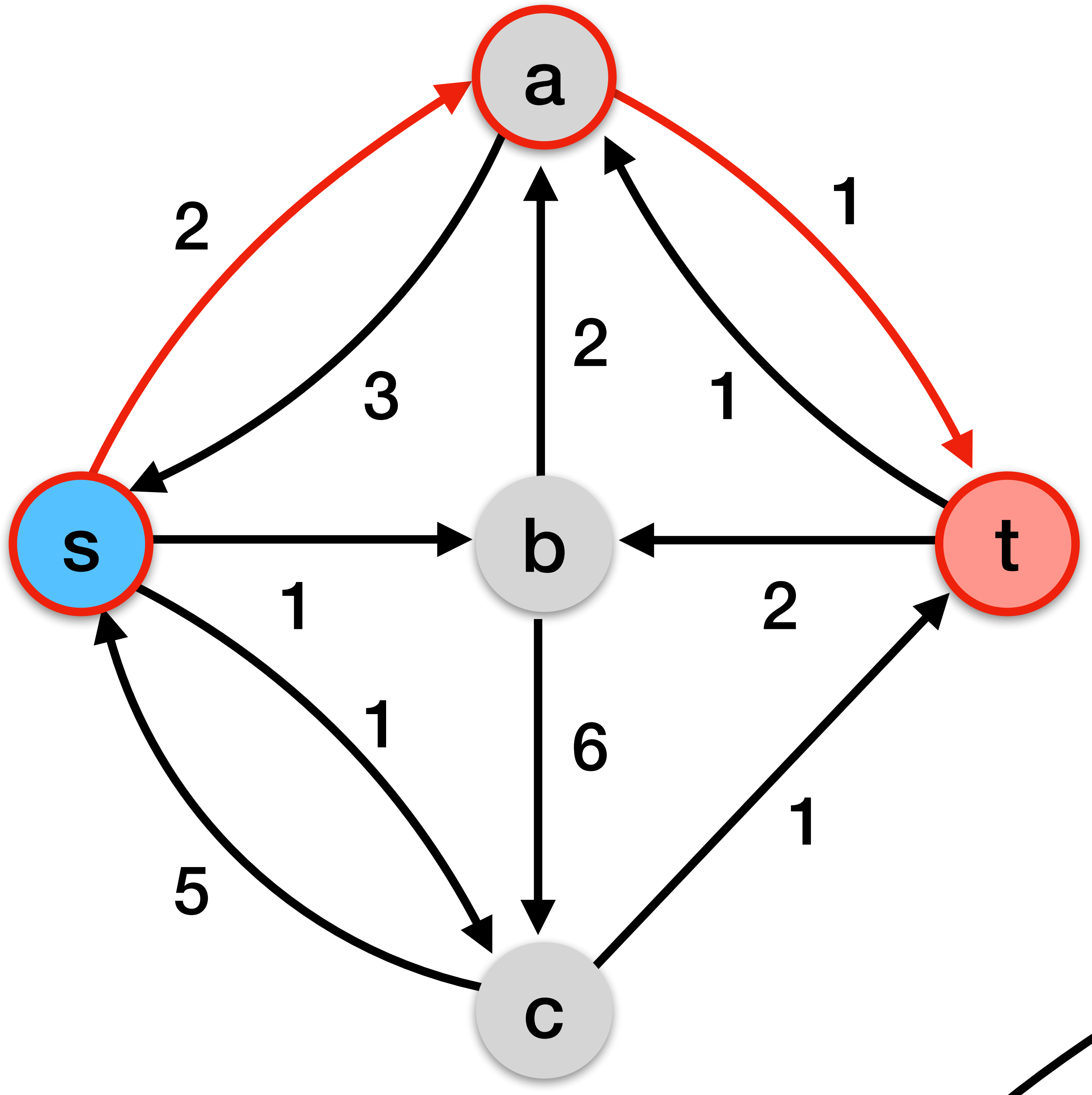


Bestimme die kleinste Restkapazität $\varepsilon := \min_i \varepsilon_i$

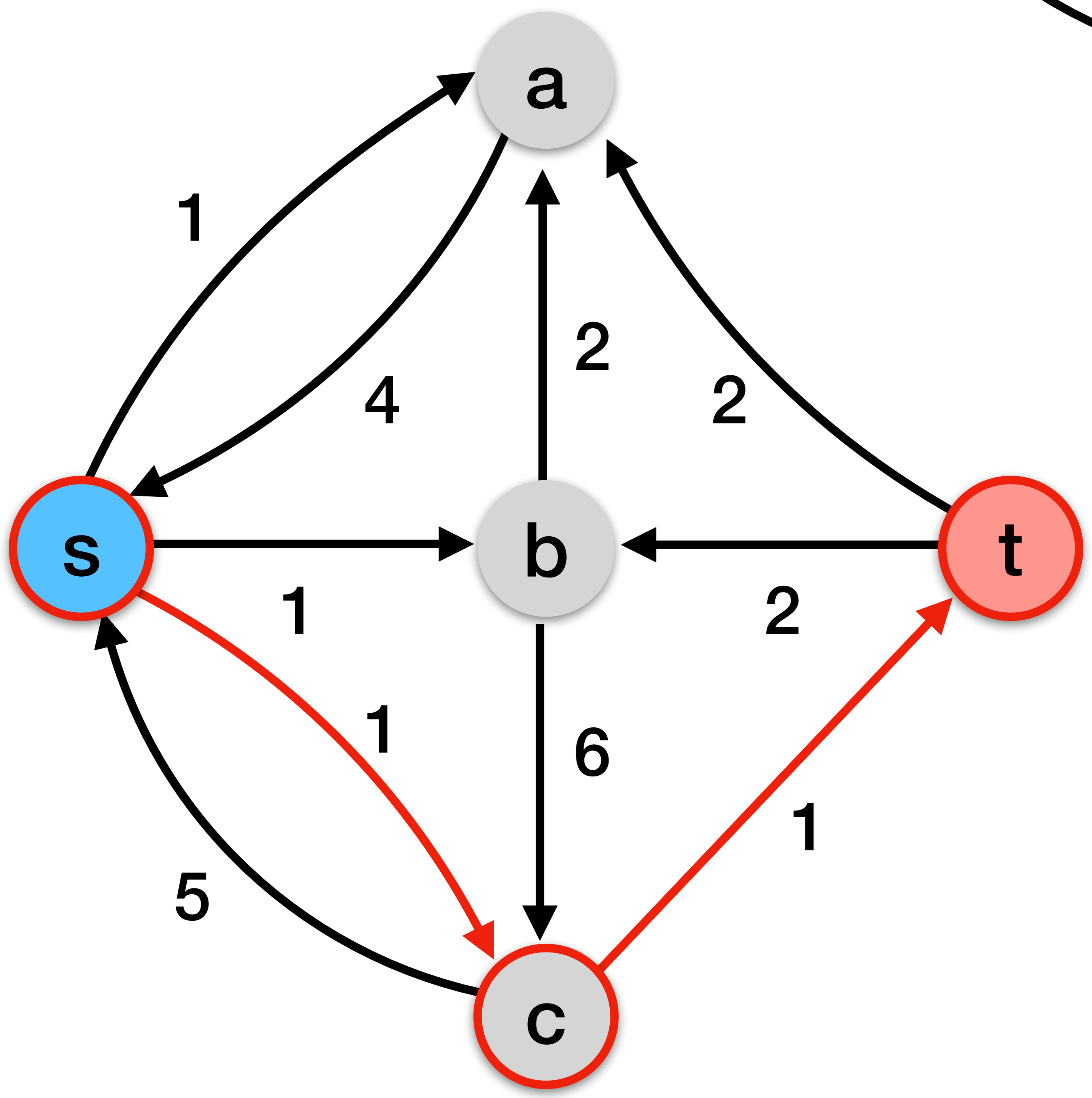
Augmentiere f entlang des Pfades um ε.

Der blaue Pfad ist ein gerichteter Pfad im Restnetzwerk N_f. Die drunterliegenden Kanten, sind die Kanten des Netzwerkes.

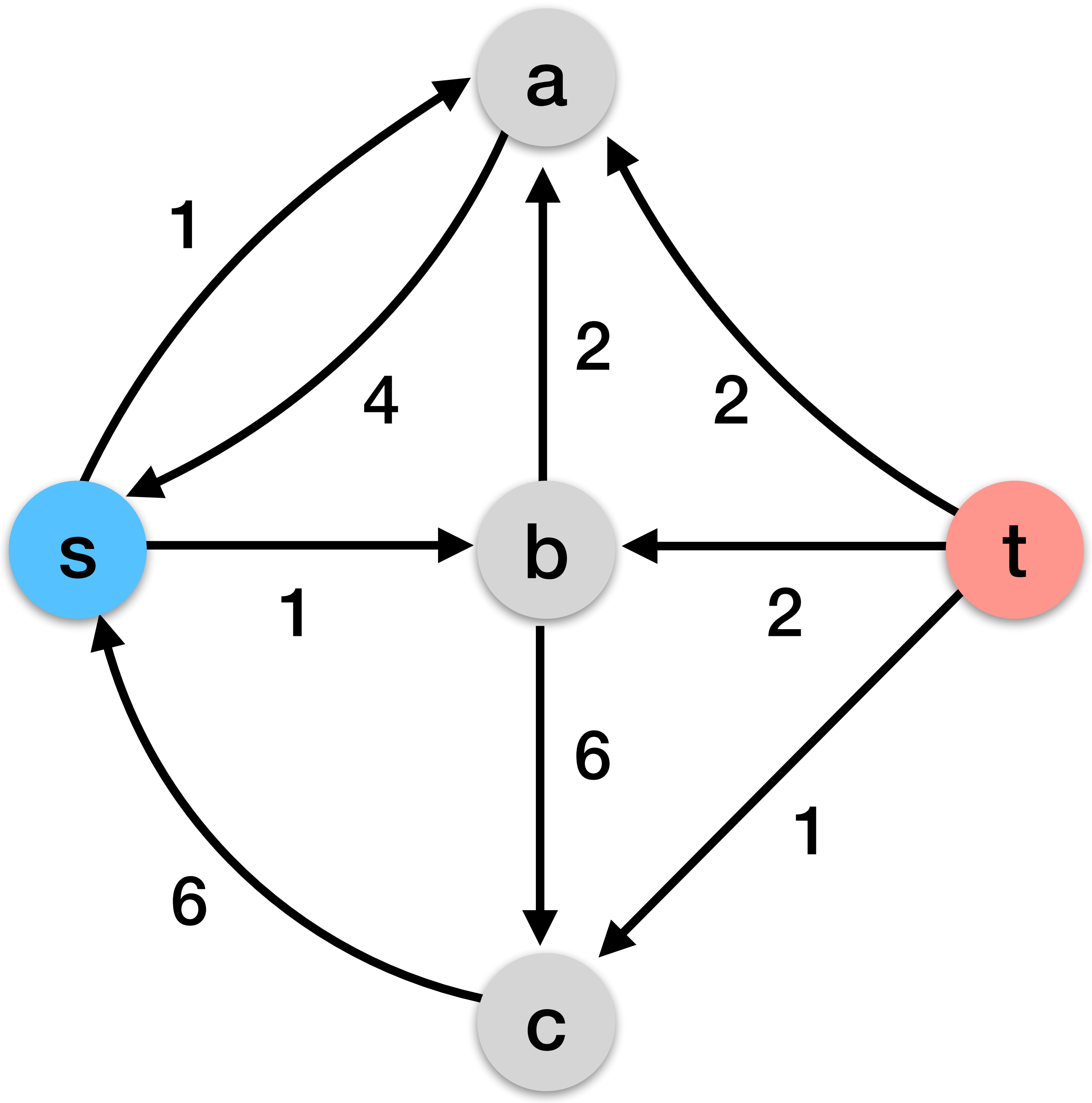
1. Zeigt die blaue Kante in die selbe Richtung zeigt, dann gehen wir entlang der Restkapazität, können also noch erhöhen.
2. Zeigt die blaue Kante in die entgegengesetzte Richtung, dann gehen wir entlang des Flusses und reduzieren.

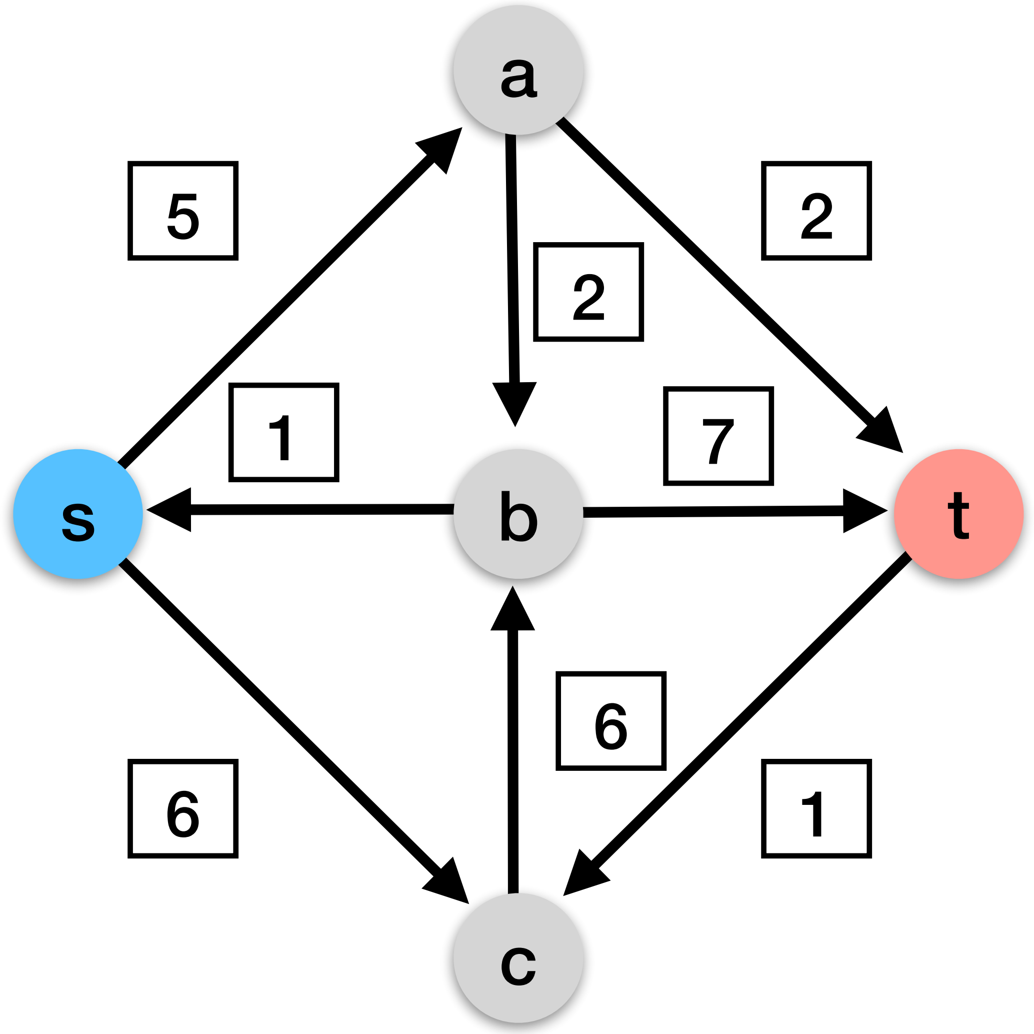


augmentiere mit $\varepsilon = 1$.

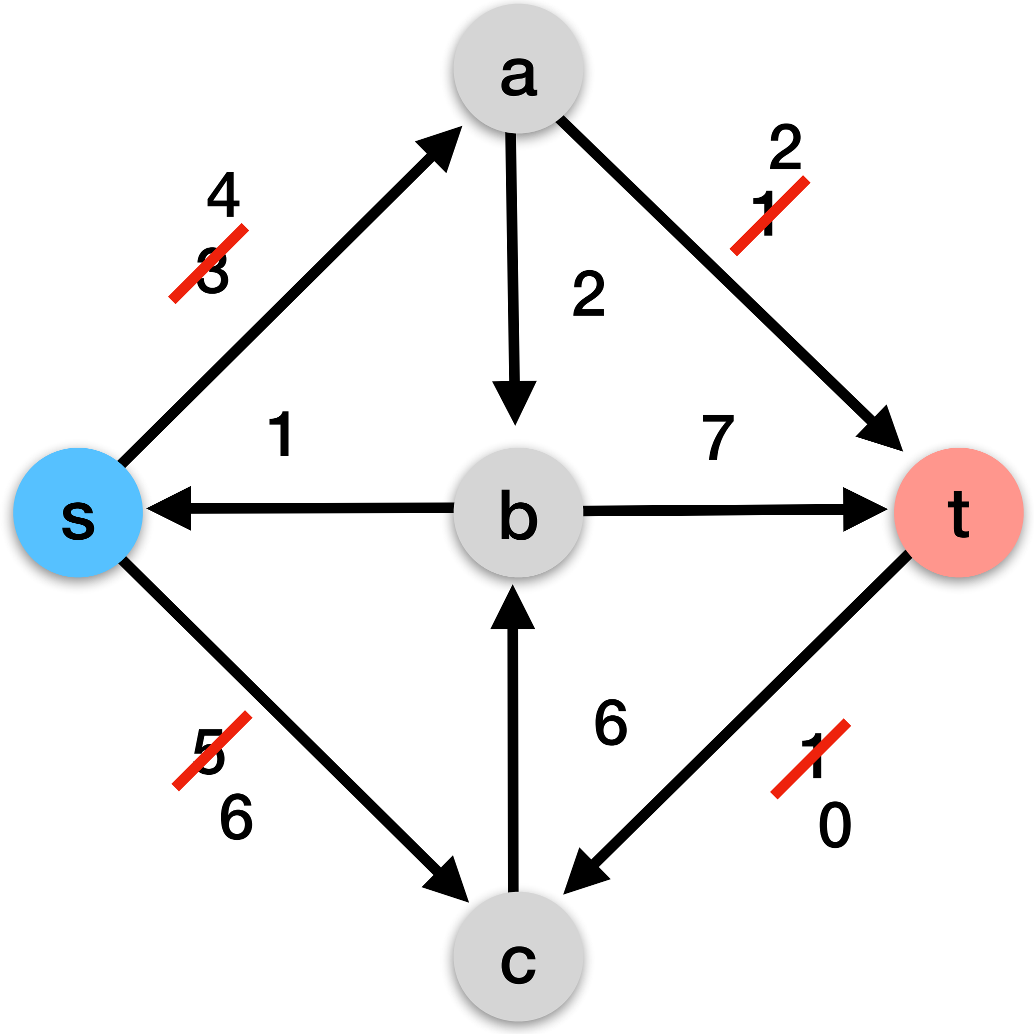


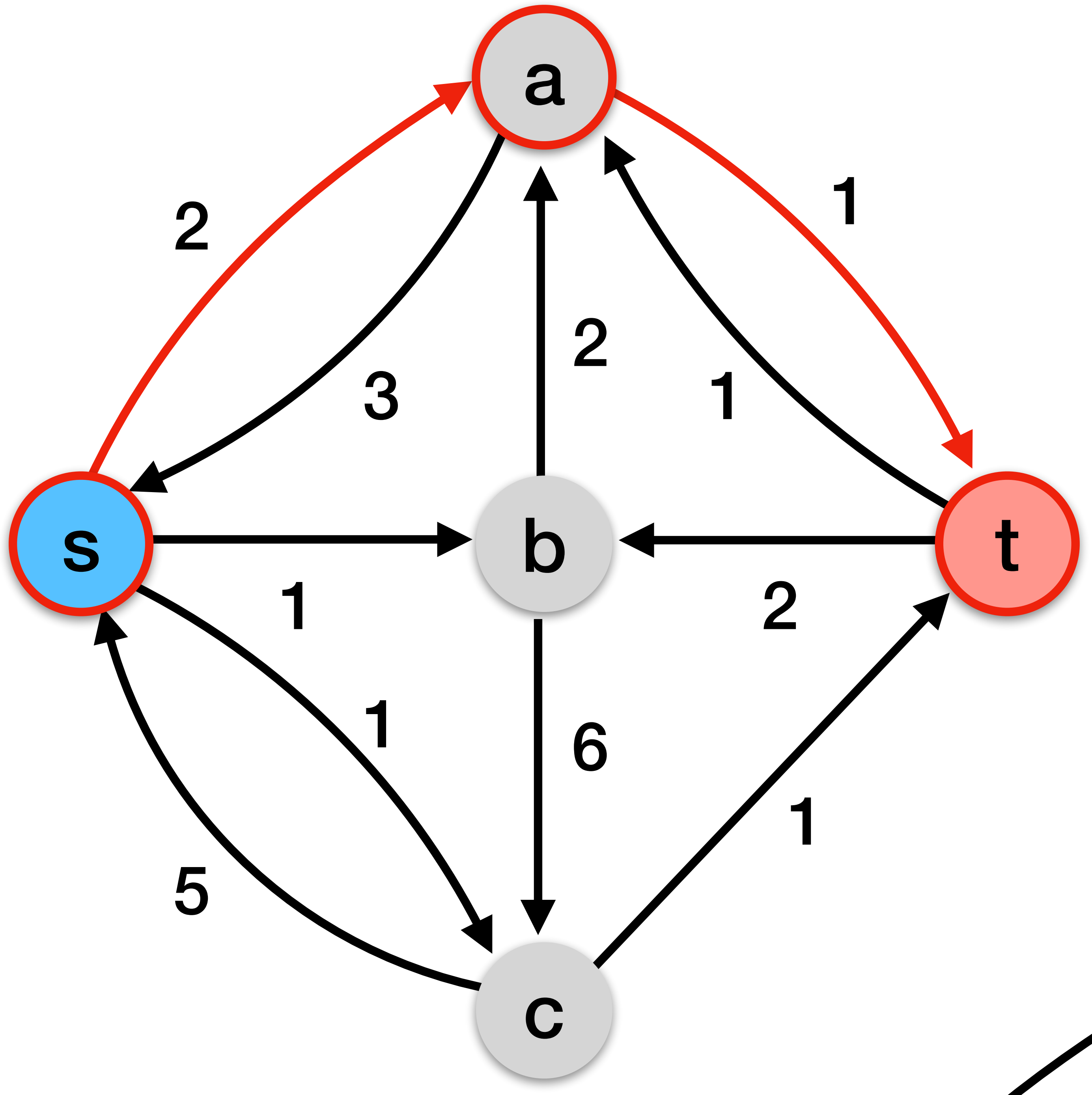
augmentiere mit $\varepsilon = 1$.

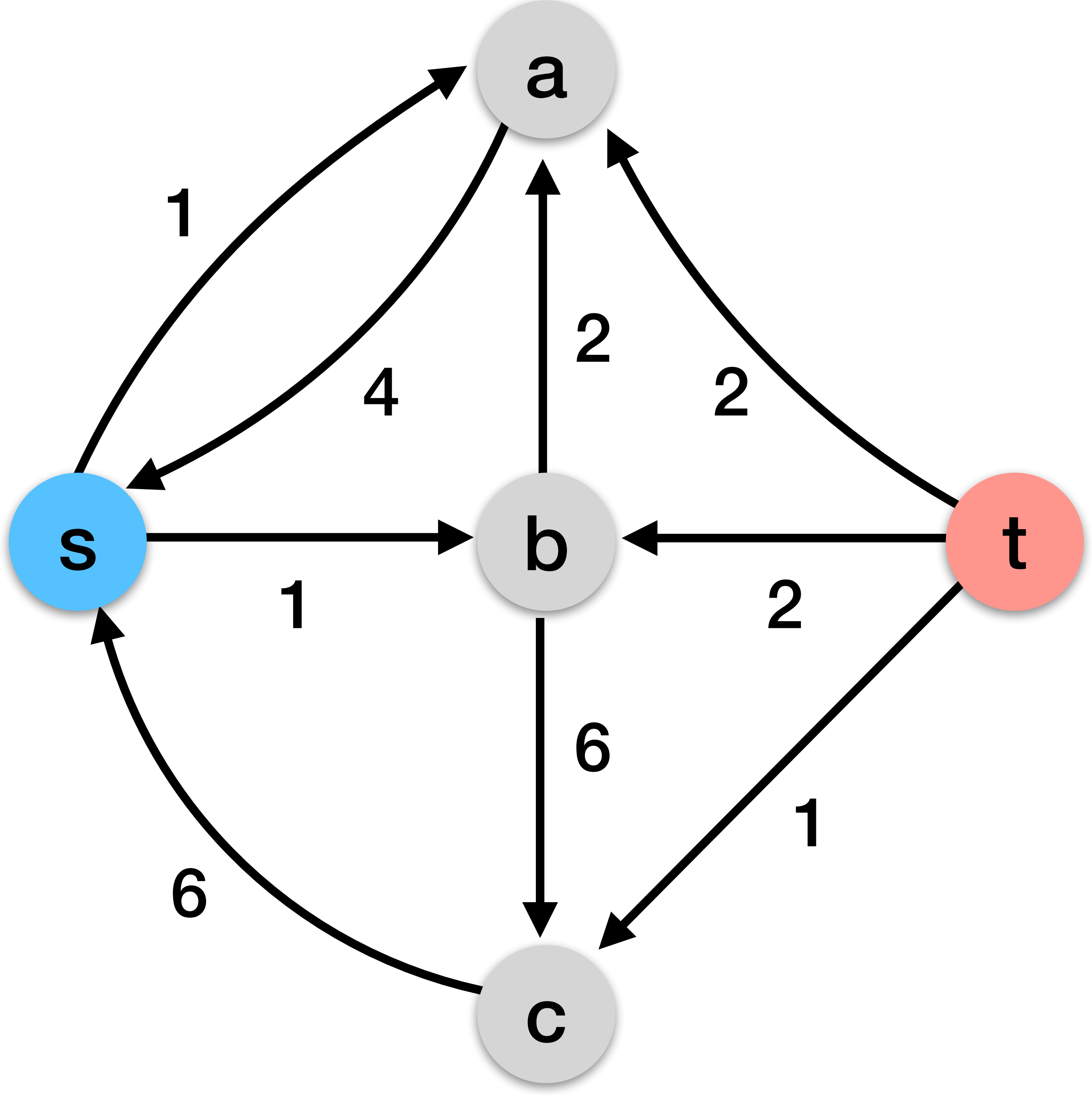


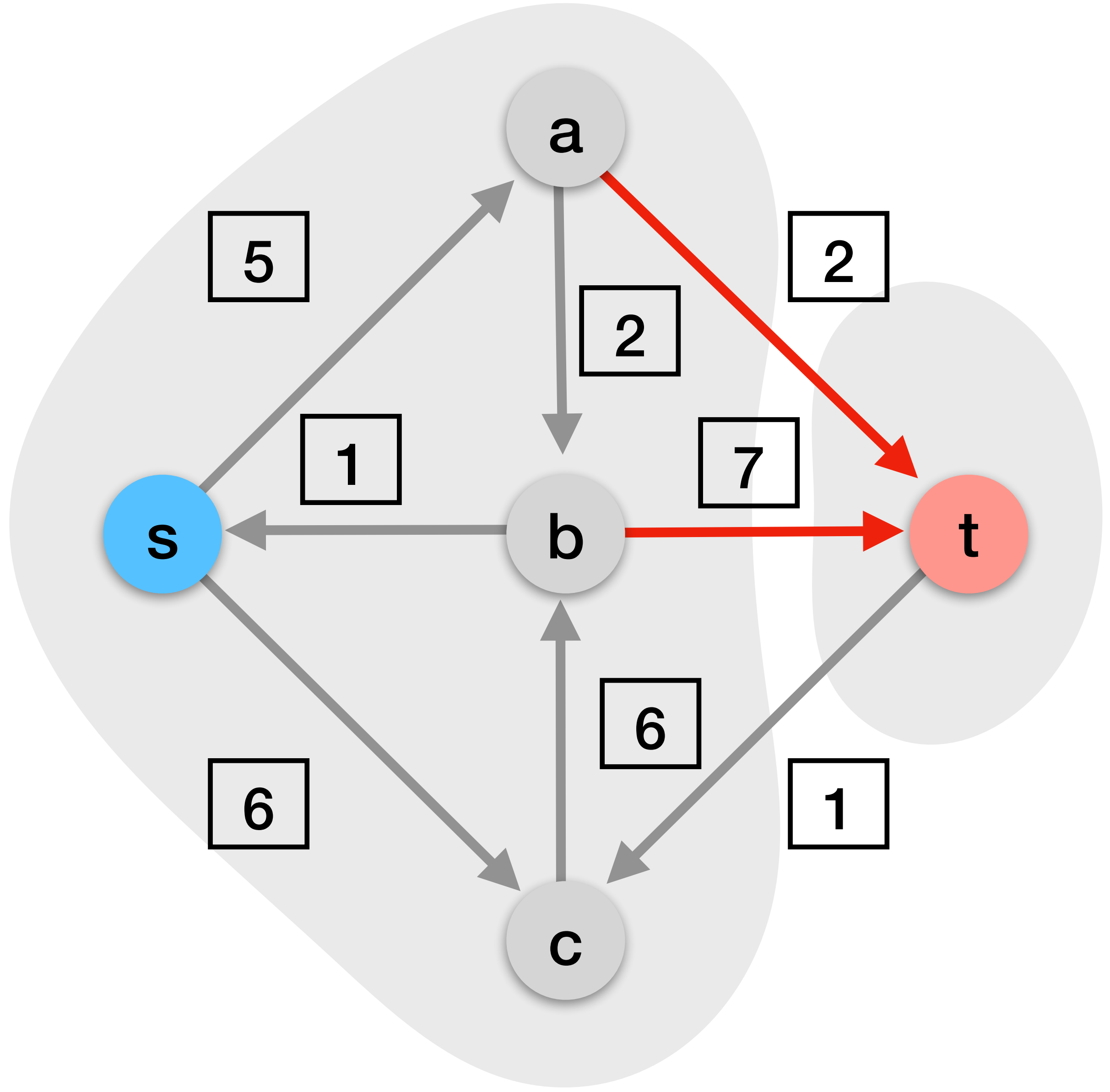


Fluss f mit Wert ~~7~~ ~~8~~ 9.



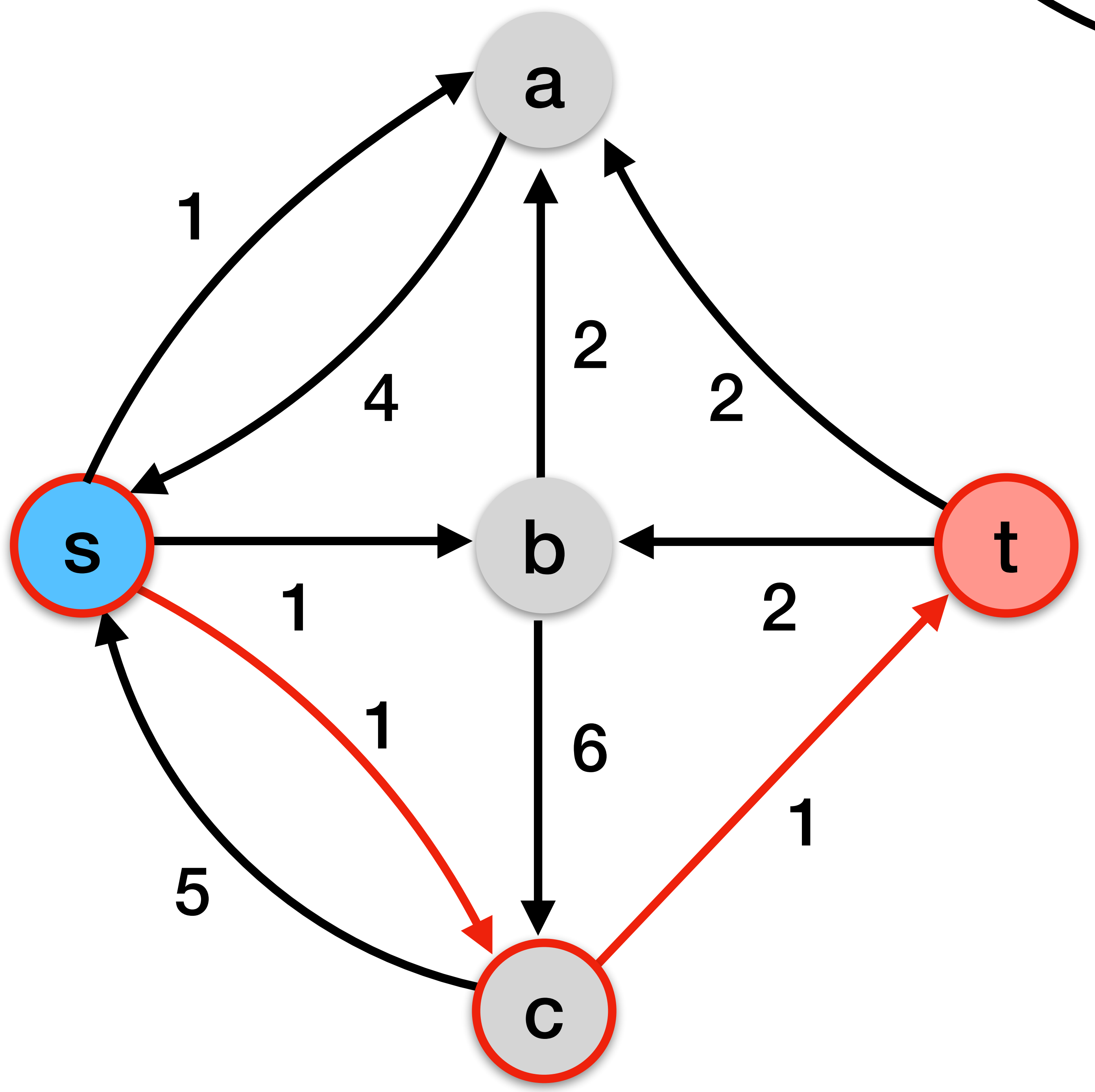




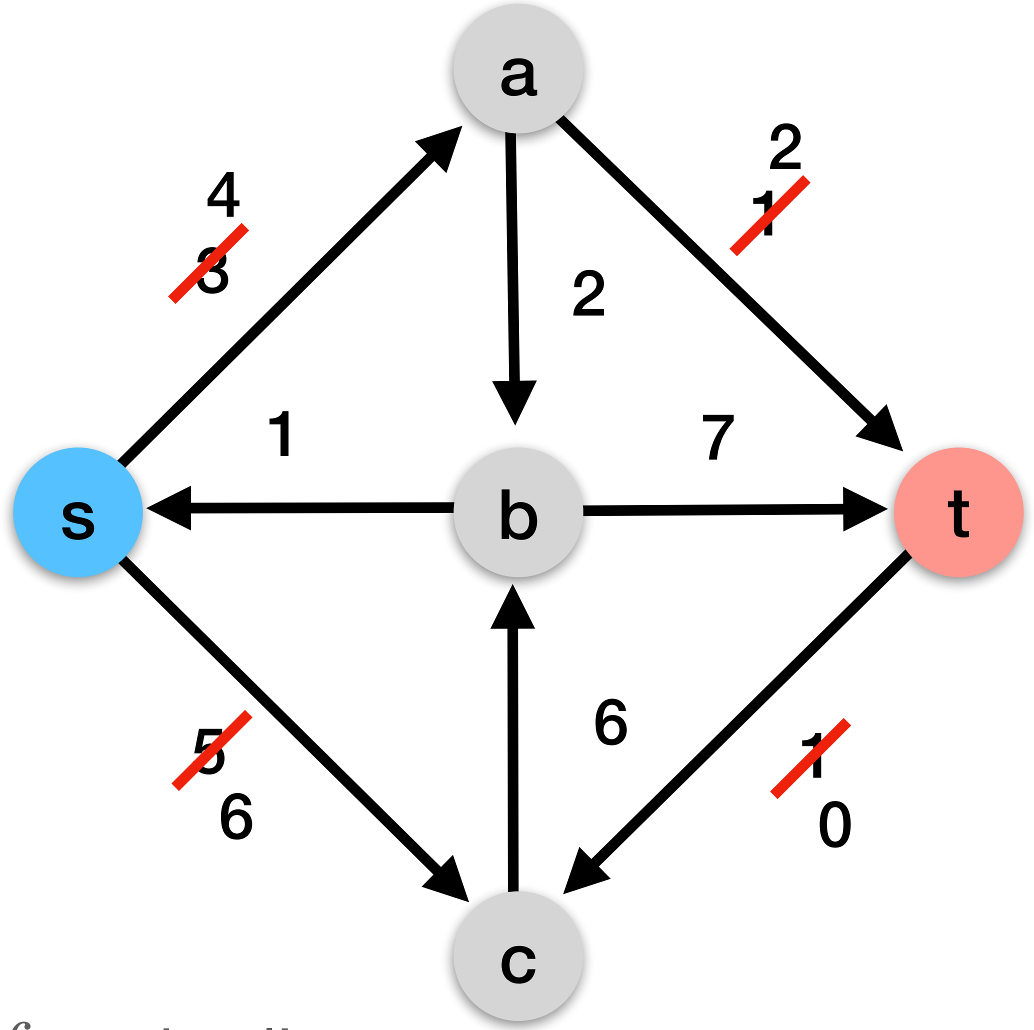


Schnitt mit cap$(S,T) = 2 + 7 = 9$.

augmentiere mit $\varepsilon = 1$.



augmentiere mit $\varepsilon = 1$.



Fluss f mit Wert $\cancel{7}$ $\cancel{8}$ 9.

Nach dem Maxflow-Mincut Theroem ist f maximal!

Resultat

Satz

Sei N ein Netzwerk (ohne entgegegen gerichtete Kanten).

$$\text{Ein Fluss } f \text{ ist maximaler Fluss}$$
$$\Leftrightarrow$$
$$\text{es im Restnetzwerk } N_f \text{ keinen gerichteten } s\text{-}t\text{-Pfad gibt.}$$

Für jeden maximalen Fluss f

gibt es einen s-t-Schnitt (S, T) mit $\mathrm{val}(f) = \mathrm{cap}(S, T)$.

Zusammenfassung

Abstraktion: Flüsse in Netzwerken.

Maximaler Fluss:
$$\max_{f \text{ Fluss}} \mathrm{val}(f)$$

Wollen Flusswert erhöhen.

Definiere Restnetzwerk.
Erhöhe solange bis kein s-t Pfad mehr existiert.

Können zeigen, dass, wenn kein s-t Pfad mehr existiert, ein Schnitt existiert mit $\mathrm{cap}(S, T) = \mathrm{val}(f)$.

Da $\mathrm{val}(f) \leq \mathrm{cap}(S, T)$, wird es nicht mehr besser.

f ist maximal.

Pretty cool if you ask me.

Ford-Fulkerson

Gegeben: Ein Netzwerk $N = (V, A, c, s, t)$.

Gesucht: Ein maximaler Fluss f.

Ford-Fulkerson(V, A, c, s, t)

$f \leftarrow \mathbf{0}$ ▷ Fluss konstant 0
while $\exists$ s-t-Pfad P in N_f **do** ▷ augmentierender Pfad
    Augmentiere den Fluss entlang P
return f ▷ maximaler Fluss

Laufzeit

Sei $n := |V|$ und $m := |A|$ für Netzwerk $N = (V, A, c, s, t)$.

- Angenommen $c\colon A \to \mathbb{N}_0$ und $U := \max_{e \in A} c(e)$. Dann gilt
$$\mathrm{val}(f) \le \mathrm{cap}(\{s\}, V \setminus \{s\}) \le (n-1)U$$
und es gibt **höchstens $(n-1)U$ Augmentierungsschritte**.
- **Ein Augmentierungsschritt**
  Suche s-t-Pfad in N_f, Augmentieren, Aktualisierung von N_f
  benötigt $O(m)$ Zeit.

Satz (Ford-Fulkerson mit ganzzahligen Kapazitäten)

Sei $N = (V, A, c, s, t)$ ein Netzwerk mit $c : A \to \mathbb{N}_0^{\le U}$, $U \in \mathbb{N}$, ohne entgegen gerichtete Kanten.[2] *Dann gibt es einen ganzzahligen maximalen Fluss. Er kann in Zeit $O(mnU)$ berechnet werden.*

1. Das Restnetzwerk wird nicht in jedem Schritt neu konstruiert, sondern schrittweise entlang des gewählten augmentierenden Pfades verändert.
2. In jedem Schritt erhöhen wir in diesem Fall den Wert des Flusses um einen ganzzahligen Wert (≥ 1). Und das gerade weil die Kapazitäten ganzzahlig sind! Bei irrationalen Kapazitäten terminiert der Algorithmus nicht immer.

Zusammenfassung

Der **Ford-Fulkerson Algorithmus** zeigt, dass es unter den gegebenen Umständen (Ganzzahligkeit, keine entgegen gerichtete Kanten) einen maximalen Fluss gibt.

Damit ist nun auch endlich das Maxflow-Mincut Theorem bewiesen, zumindest in abgeschwächter Form.

Ergebnis

Satz („Maxflow-Mincut Theorem“, ganzzahlig)

Jedes Netzwerk ohne entgegen gerichtete Kanten mit ganzzahligen Kapazitäten erfüllt

$$\max_{f \text{ Fluss}} \mathrm{val}(f) = \min_{(S,T) \text{ s-t-Schnitt}} \mathrm{cap}(S,T) \ .$$

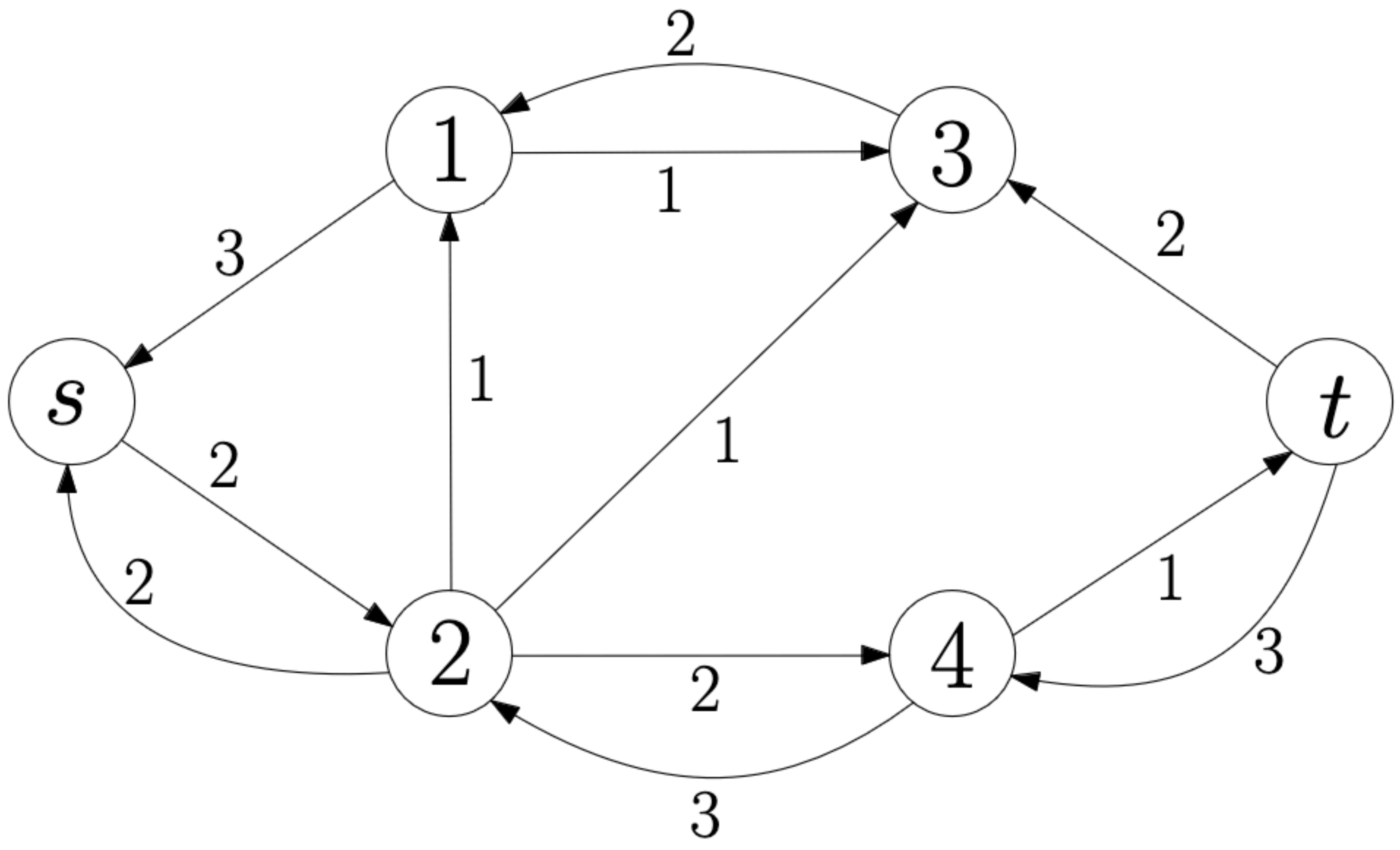
Empfehlungen

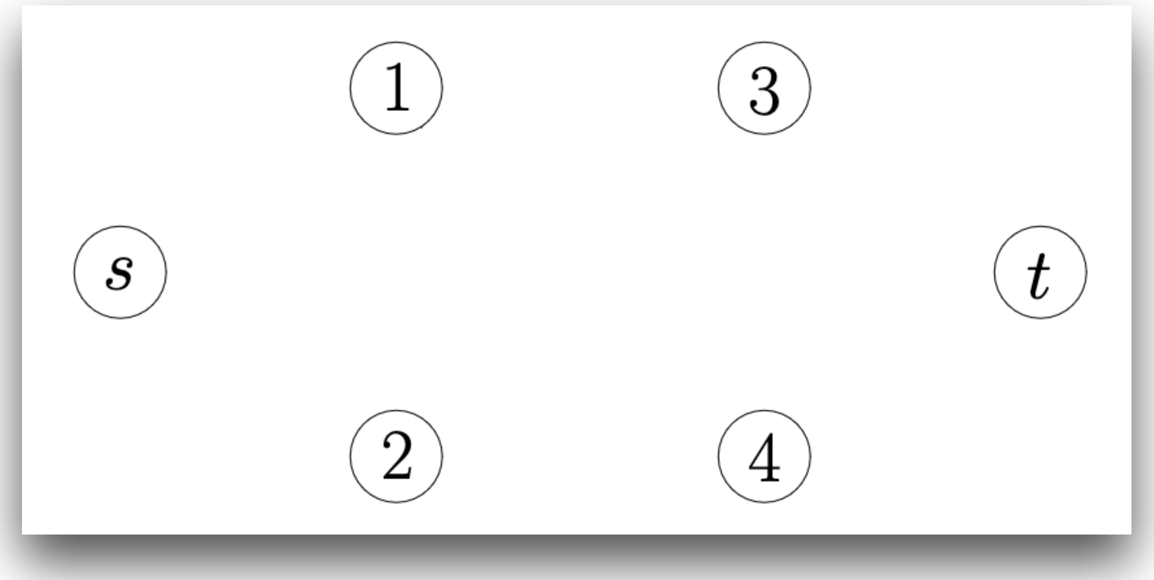
- Beweis Lemma 3.6.
- Beweis Lemma 3.8.
- Beweis Satz 3.11.

Beweise im Skript. Beweis von Lemma 3.6 mit Notizen auf meiner Website.

Aufgabe 4 – *Restnetzwerk*

Sei N ein Netzwerk ohne entgegengesetzte Kanten und sei f ein Fluss in G. Unten abgebildet sehen Sie das Restnetzwerk R_f.





This template was given.

(a) Ist f maximal? Geben Sie eine kurze Begründung für Ihre Antwort. *(1 Punkte)*

(b) Rekonstruieren sie N und f. *(4 Punkte)*

AlgoWahr FS21.

cont'd on iPad, please see notes.