Effiziente Algorithmen (SS2022)

Chapter 1
Grundlagen zu Flüssen

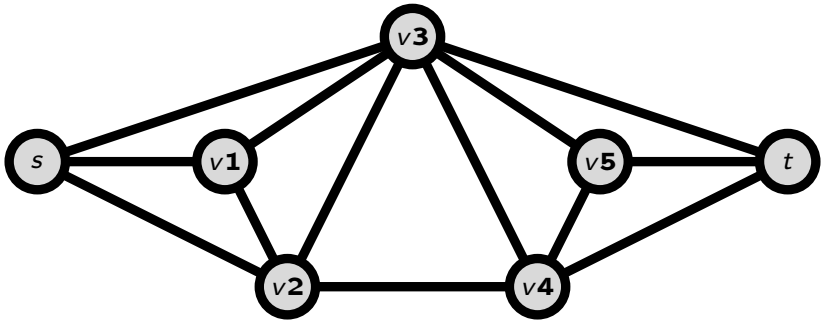
Walter Unger

Lehrstuhl für Informatik 1

30.04.2022 11:06:46

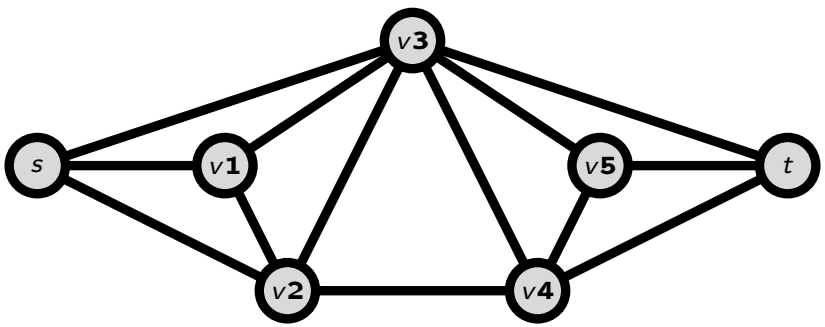
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.



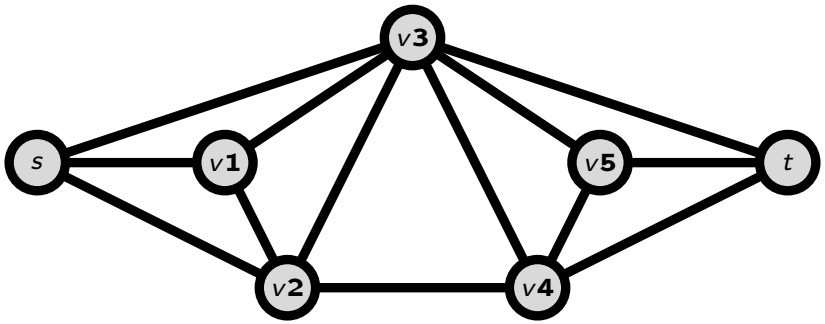
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.
- Kapazität der Datenleitungen von Amerika nach China.



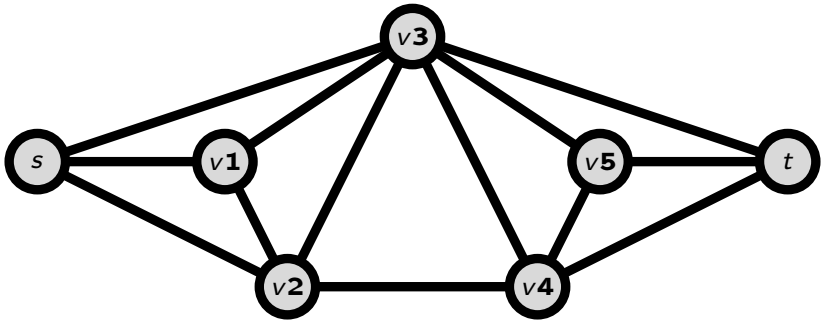
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.
- Kapazität der Datenleitungen von Amerika nach China.
- Kapazität des Deltas des Amazonas.



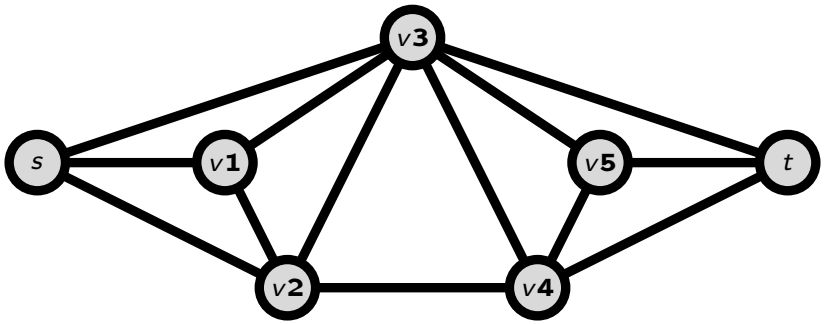
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.
- Kapazität der Datenleitungen von Amerika nach China.
- Kapazität des Deltas des Amazonas.
- Kapazität der Abwasserkanäle von Rom.



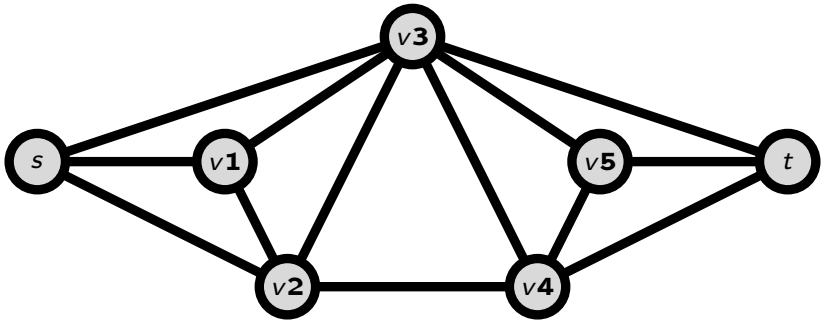
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.
- Kapazität der Datenleitungen von Amerika nach China.
- Kapazität des Deltas des Amazonas.
- Kapazität der Abwasserkanäle von Rom.
- Alle durch Graphen modellierbaren Durchsatzprobleme.



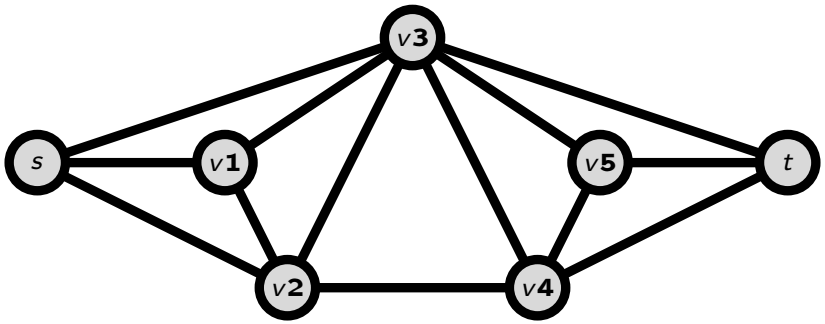
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.
- Kapazität der Datenleitungen von Amerika nach China.
- Kapazität des Deltas des Amazonas.
- Kapazität der Abwasserkanäle von Rom.
- Alle durch Graphen modellierbaren Durchsatzprobleme.
- Hilfreich bei anderen Anwendungen.



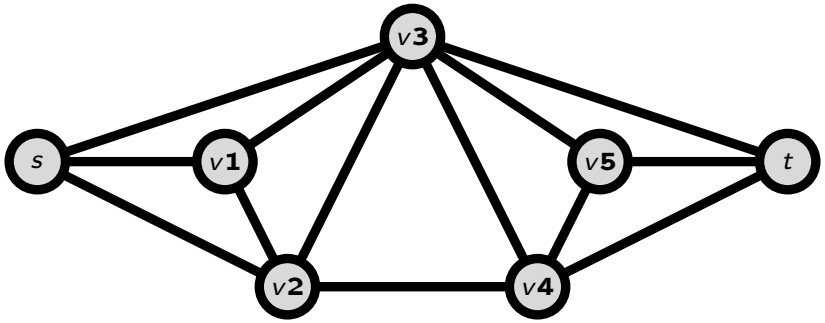
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.
- Kapazität der Datenleitungen von Amerika nach China.
- Kapazität des Deltas des Amazonas.
- Kapazität der Abwasserkanäle von Rom.
- Alle durch Graphen modellierbaren Durchsatzprobleme.
- Hilfreich bei anderen Anwendungen.
- Altes Problem.



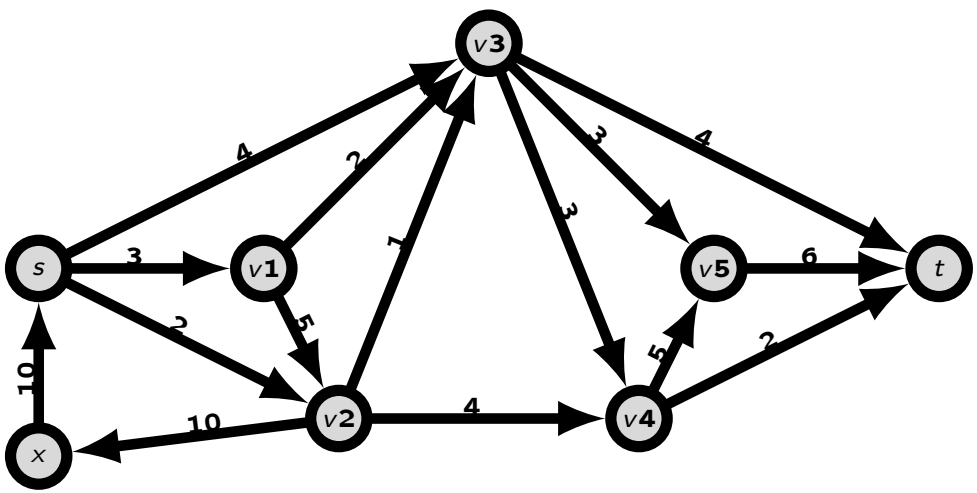
Anwendungen

- Kapazität der Straßen von Aachen nach Köln.
- Kapazität der Datenleitungen von Amerika nach China.
- Kapazität des Deltas des Amazonas.
- Kapazität der Abwasserkanäle von Rom.
- Alle durch Graphen modellierbaren Durchsatzprobleme.
- Hilfreich bei anderen Anwendungen.
- Altes Problem.



Modellierung

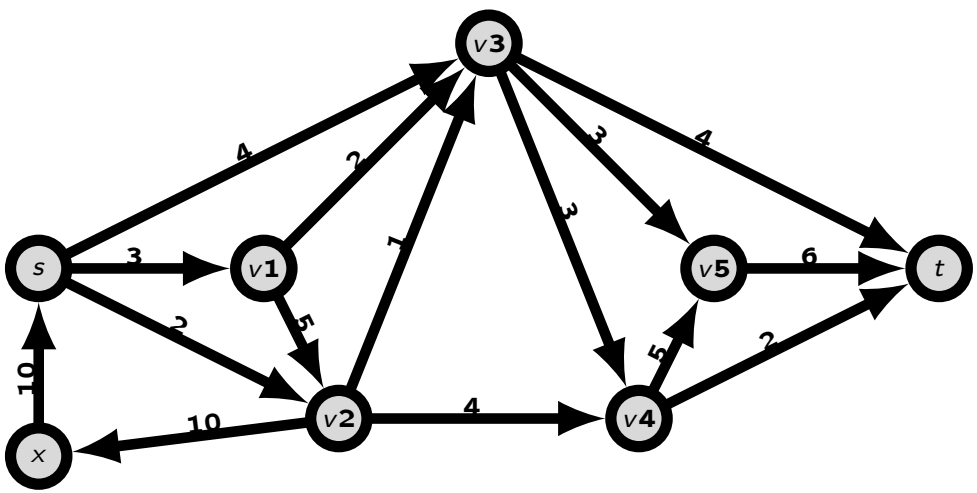
- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.



Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

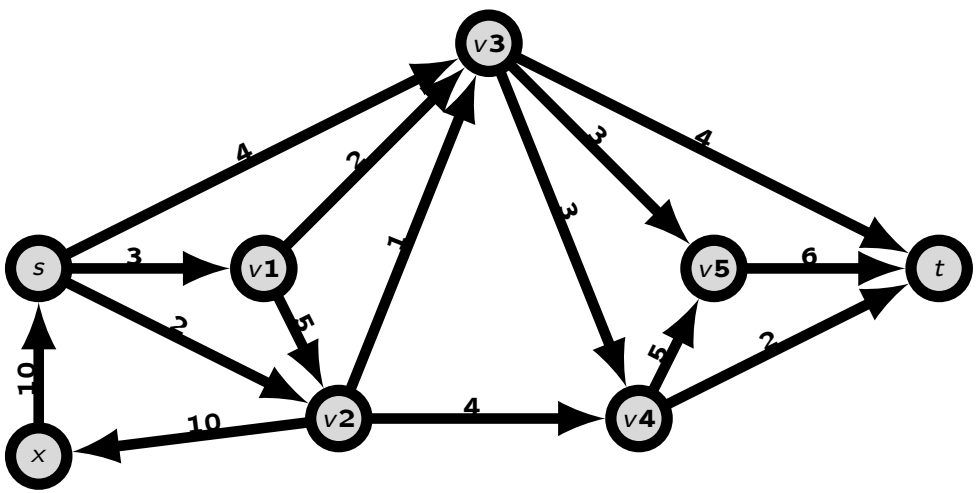


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.

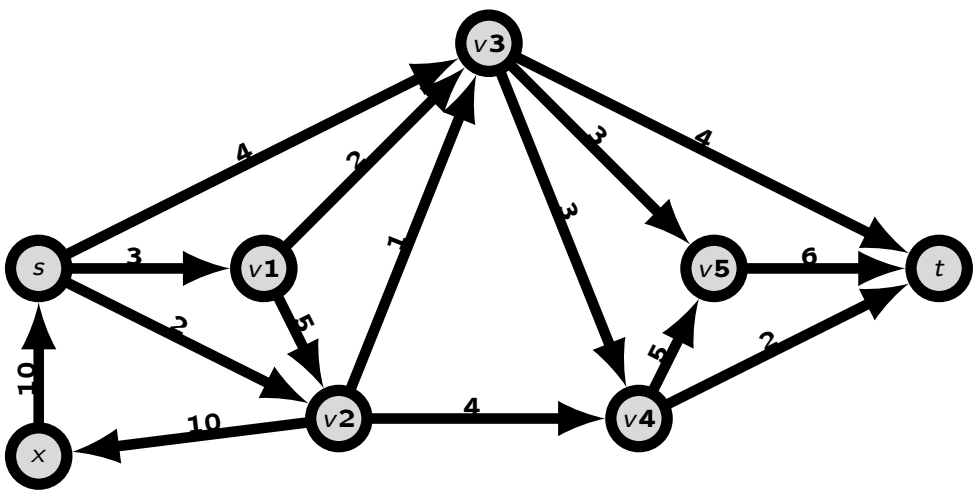


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

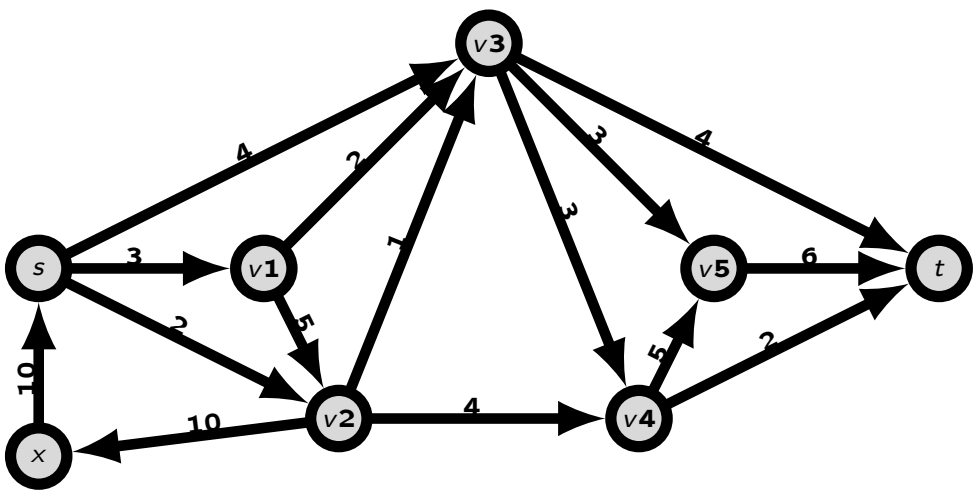


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

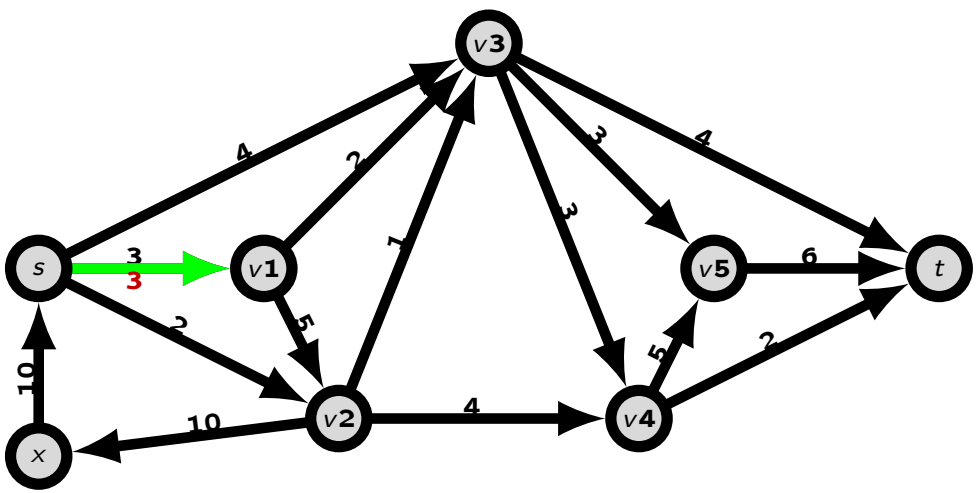


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

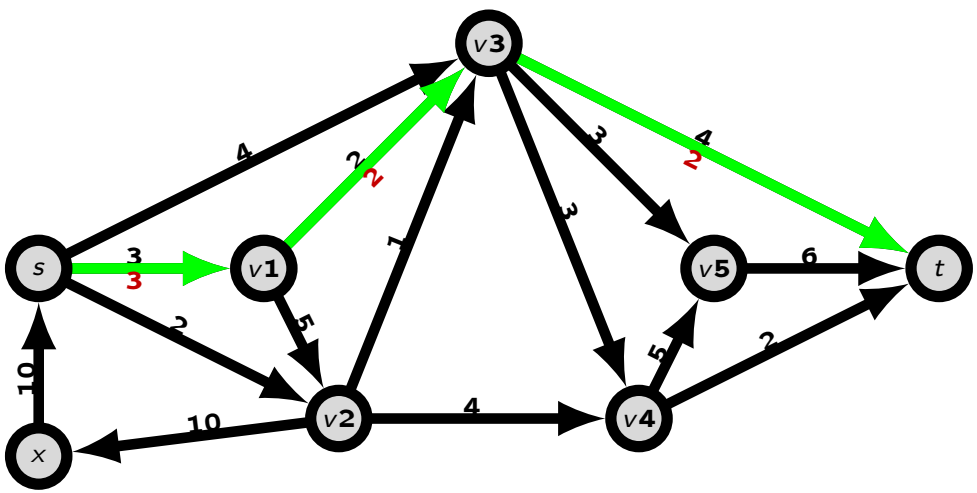


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

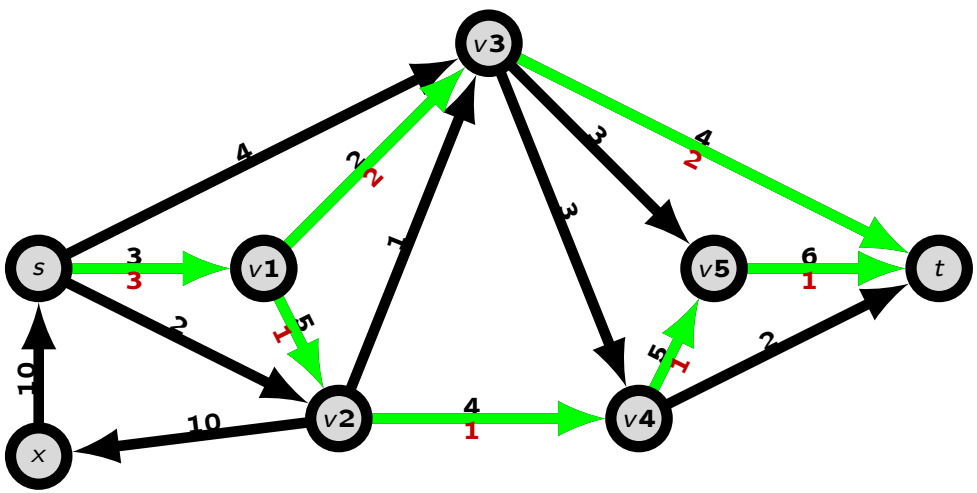


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

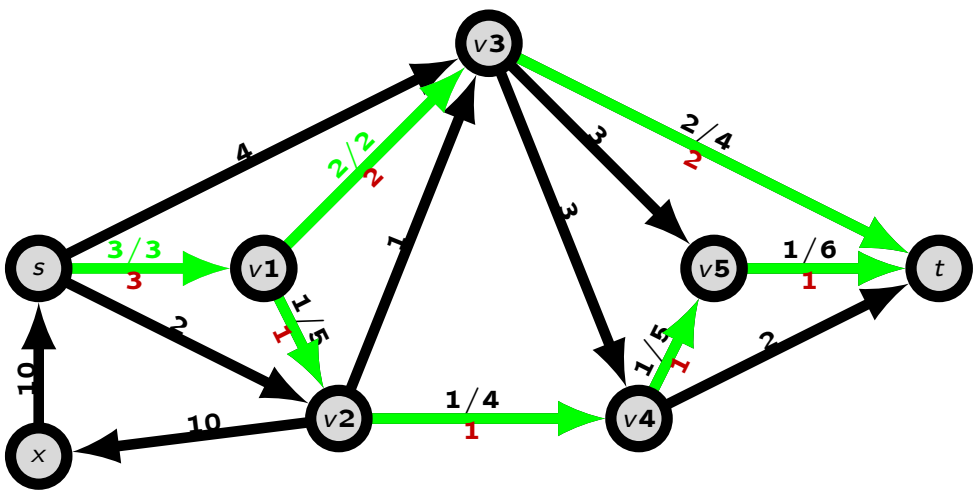


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

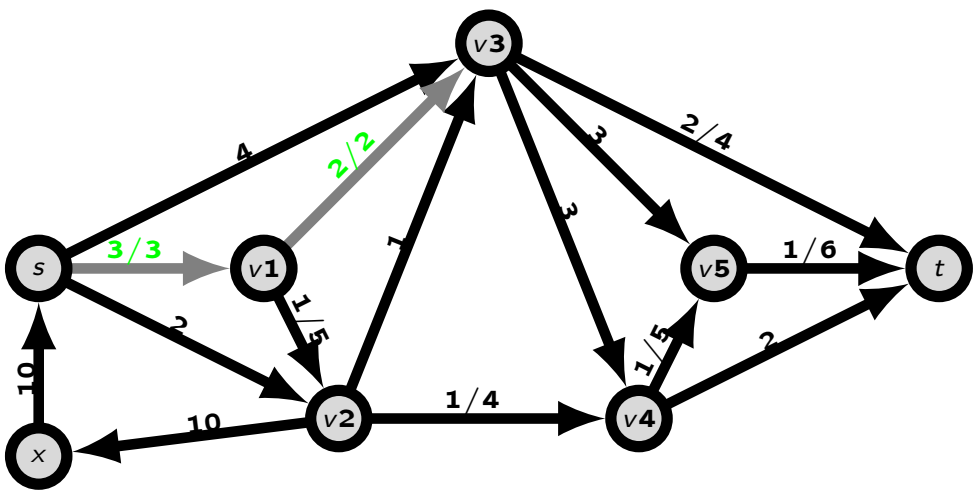


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

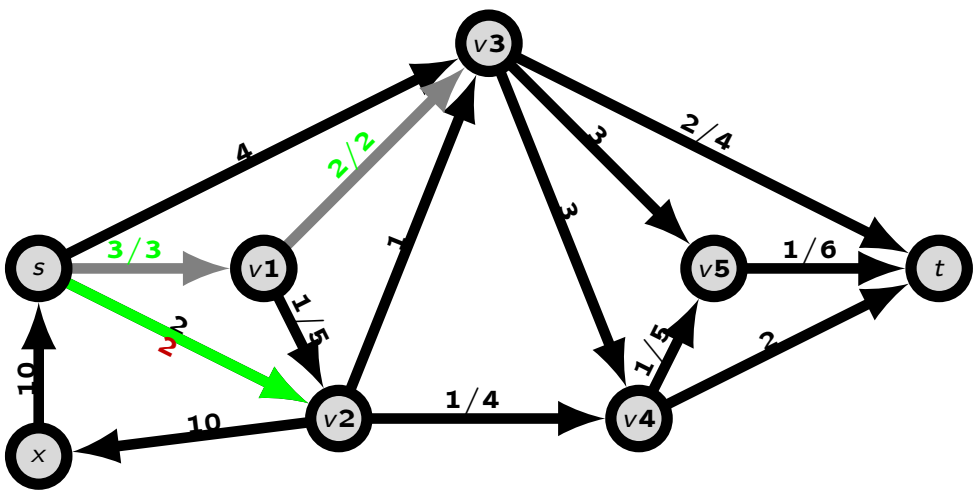


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

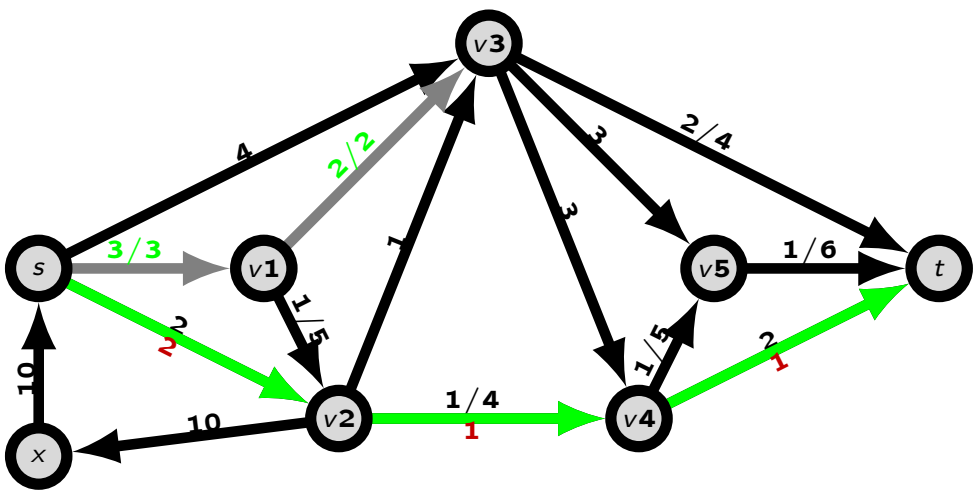


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

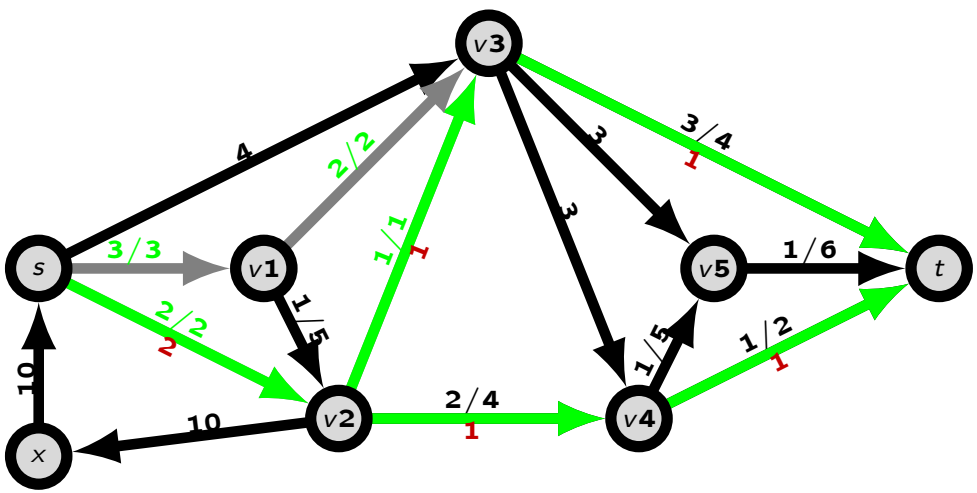


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

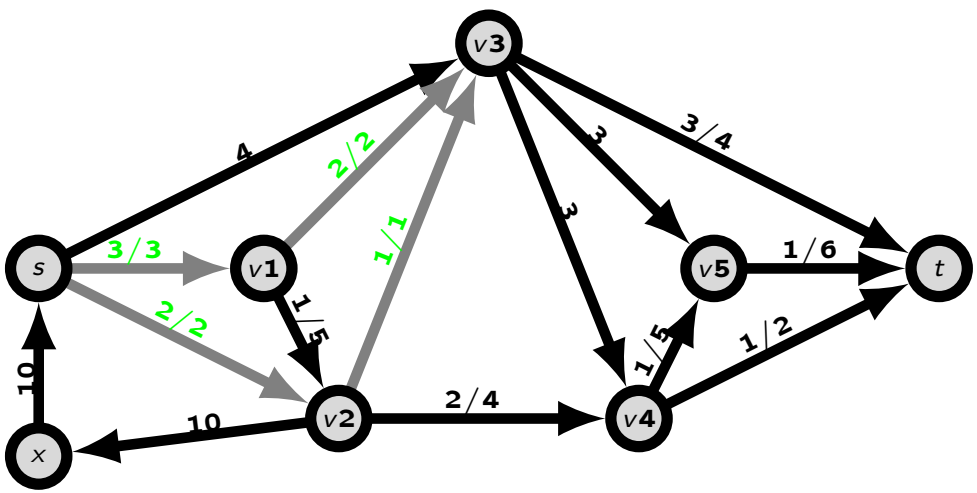


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

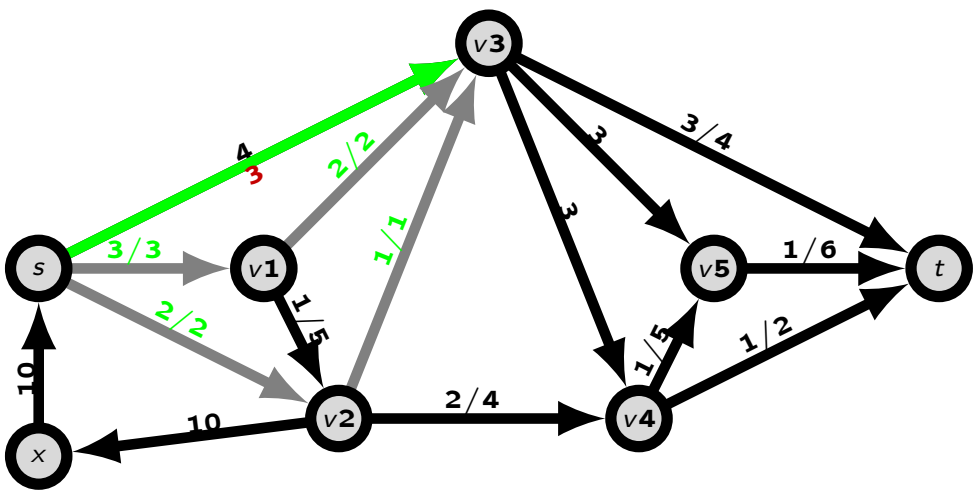


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

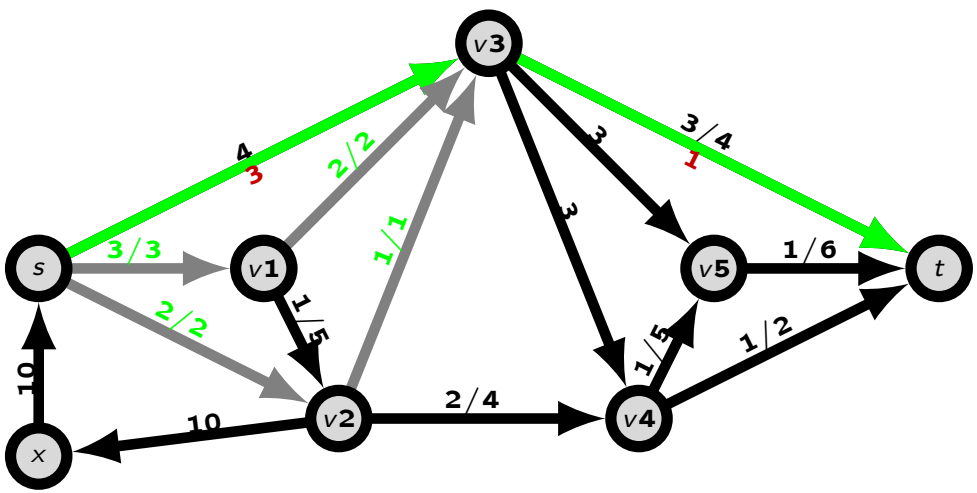


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

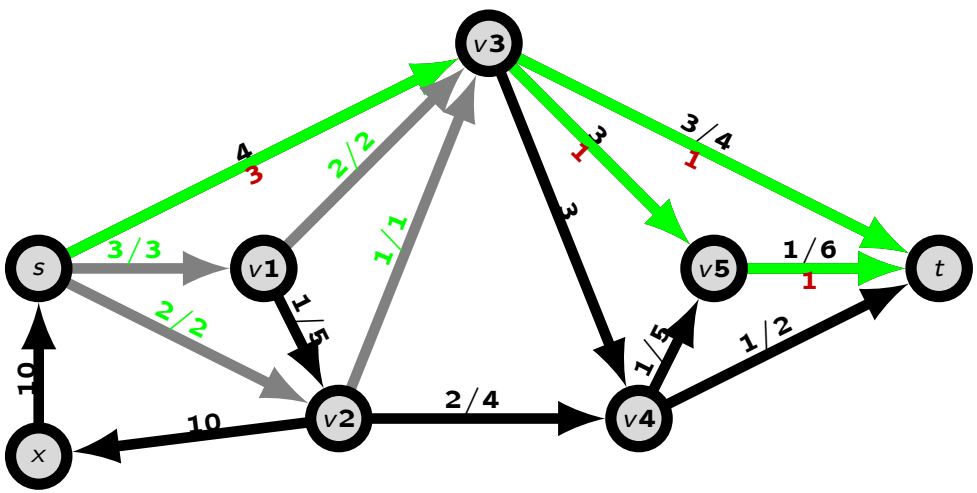


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

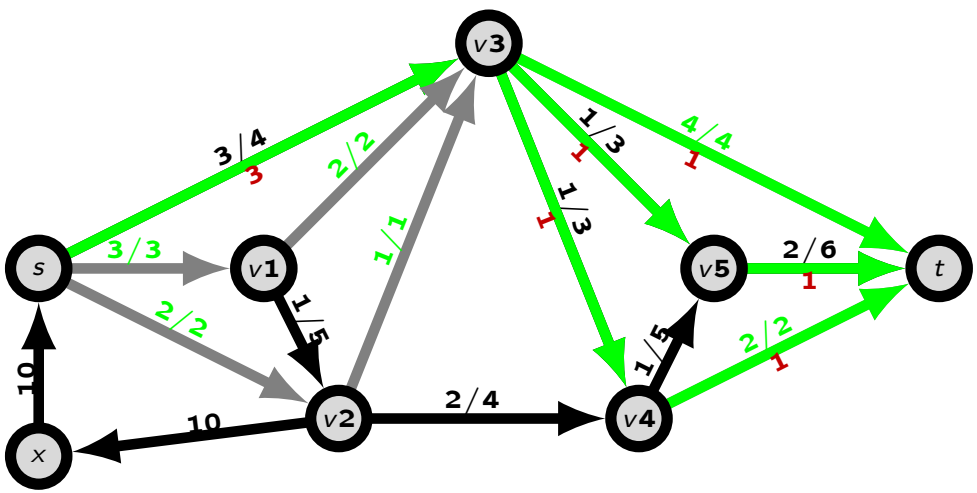


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

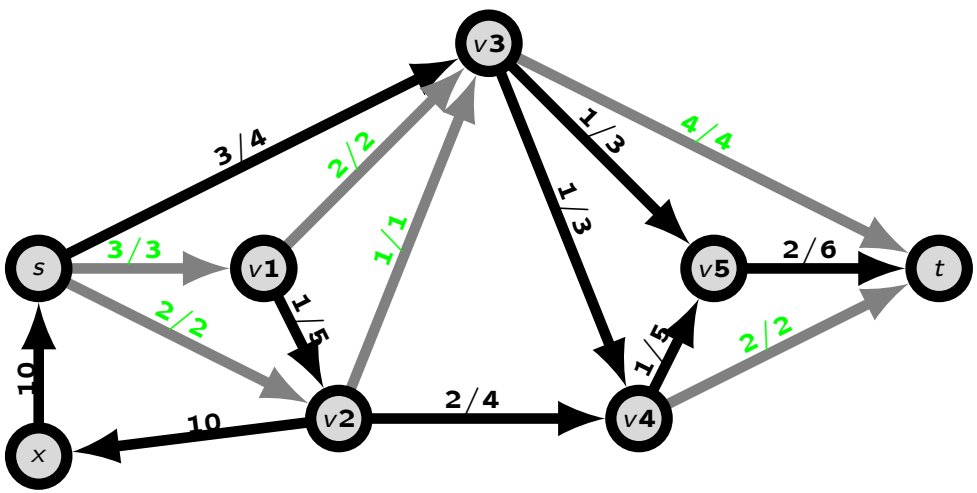


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

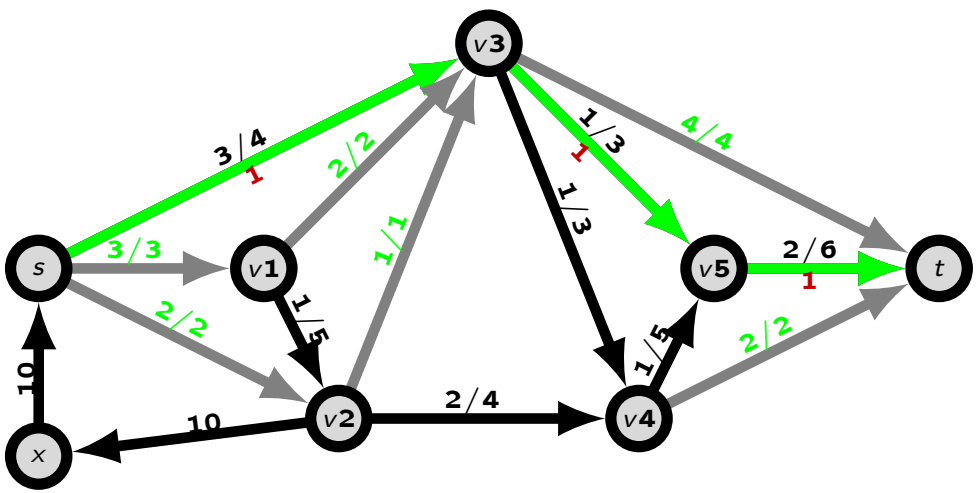


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.

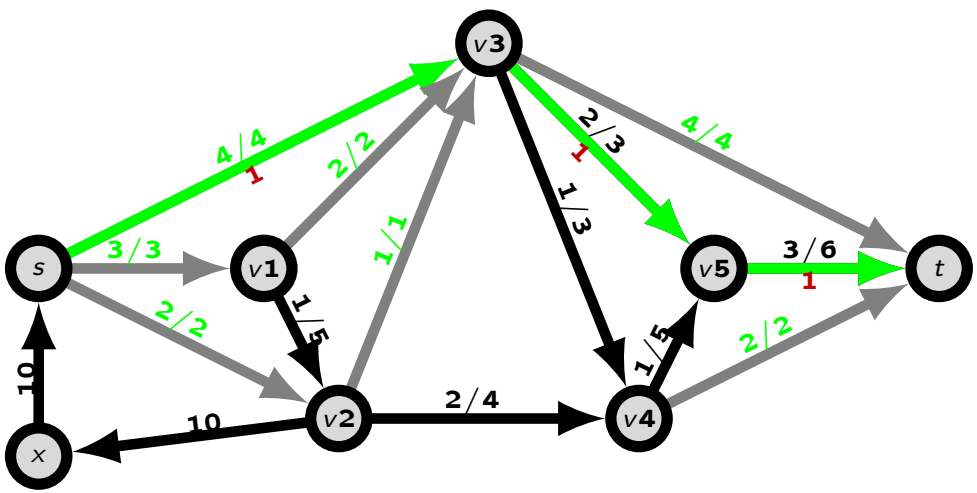


Modellierung

- Das Netzwerk wird als Graph $G = (V, E)$ modelliert.
- Die Kanten haben eine maximale Kapazität:

$$c : E \mapsto \mathbb{R}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{Q}^+ \quad \text{oder}$$
$$c : E \mapsto \mathbb{N}^+$$

- Der Fluss startet an einem Knoten $s \in V$.
- Der Fluss endet an einem Knoten $t \in V$.



Das Flussproblem

Definition (Flussproblem)

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : 0 \leqslant f(e) \leqslant c(e)$

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : 0 \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : 0 \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Ziel: Maximiere $w(f) = \sum_{(s,v)\in E} f((s, v))$.

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : 0 \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Ziel: Maximiere $w(f) = \sum_{(s,v)\in E} f((s, v))$.

- s heißt Quelle und t heißt Senke.

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : 0 \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Ziel: Maximiere $w(f) = \sum_{(s,v)\in E} f((s, v))$.

- s heißt Quelle und t heißt Senke.
- c ist die Kapazitätsfunktion.

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : 0 \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Ziel: Maximiere $w(f) = \sum_{(s,v)\in E} f((s, v))$.

- s heißt Quelle und t heißt Senke.
- c ist die Kapazitätsfunktion.
- f nennt man die Flussfunktion.

Das Flussproblem

Definition (Flussproblem)

Eingabe: $G = (V, E, s, t, c)$ mit:

- (V, E) ist ein gerichteter Graph $(n = |V|, m = |E|)$
- $s, t \in V$ und $s \neq t$
- $c : E \mapsto \mathbb{N}^+$

Ausgabe: $f : E \mapsto \mathbb{R}_0^+$ mit:

- $\forall e : 0 \leqslant f(e) \leqslant c(e)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(a,v)\in E} f((a, v)) = \sum_{(v,a)\in E} f((v, a))$

Ziel: Maximiere $w(f) = \sum_{(s,v)\in E} f((s, v))$.

- s heißt Quelle und t heißt Senke.
- c ist die Kapazitätsfunktion.
- f nennt man die Flussfunktion.

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.
- Kapazitätsbeschränkung:

$$\forall e : f(e) \leqslant c(e)$$

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.
- Kapazitätsbeschränkung:

$$\forall e : f(e) \leqslant c(e)$$

- Schreibweisen:

$$\begin{aligned} f_{in}(v) &= \sum_{(a,v)\in E} f((a, v)) \\ f_{out}(v) &= \sum_{(v,a)\in E} f((v, a)) \end{aligned}$$

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.
- Kapazitätsbeschränkung:

$$\forall e : f(e) \leqslant c(e)$$

- Schreibweisen:

$$\begin{aligned} f_{in}(v) &= \sum_{(a,v)\in E} f((a, v)) \\ f_{out}(v) &= \sum_{(v,a)\in E} f((v, a)) \end{aligned}$$

- Flusserhaltung:

$$\forall v \in V \setminus \{s, t\} : f_{in}(v) = f_{out}(v).$$

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.
- Kapazitätsbeschränkung:

$$\forall e : f(e) \leqslant c(e)$$

- Schreibweisen:

$$\begin{aligned} f_{in}(v) &= \sum_{(a,v)\in E} f((a, v)) \\ f_{out}(v) &= \sum_{(v,a)\in E} f((v, a)) \end{aligned}$$

- Flusserhaltung:

$$\forall v \in V \setminus \{s, t\} : f_{in}(v) = f_{out}(v).$$

- Wert des Flusses f ist: $w(f) = f_{out}(s) = f_{in}(t)$.

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.
- Kapazitätsbeschränkung:

$$\forall e : f(e) \leqslant c(e)$$

- Schreibweisen:

$$\begin{aligned} f_{in}(v) &= \sum_{(a,v)\in E} f((a, v)) \\ f_{out}(v) &= \sum_{(v,a)\in E} f((v, a)) \end{aligned}$$

- Flusserhaltung:

$$\forall v \in V \setminus \{s, t\} : f_{in}(v) = f_{out}(v).$$

- Wert des Flusses f ist: $w(f) = f_{out}(s) = f_{in}(t)$.
- Eine Kapazitätsfunktion $c : E \mapsto \mathbb{Q}^+$ kann durch $c : E \mapsto \mathbb{N}^+$ modelliert werden.

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.
- Kapazitätsbeschränkung:

$$\forall e : f(e) \leqslant c(e)$$

- Schreibweisen:

$$\begin{aligned} f_{in}(v) &= \sum_{(a,v)\in E} f((a, v)) \\ f_{out}(v) &= \sum_{(v,a)\in E} f((v, a)) \end{aligned}$$

- Flusserhaltung:

$$\forall v \in V \setminus \{s, t\} : f_{in}(v) = f_{out}(v).$$

- Wert des Flusses f ist: $w(f) = f_{out}(s) = f_{in}(t)$.
- Eine Kapazitätsfunktion $c : E \mapsto \mathbb{Q}^+$ kann durch $c : E \mapsto \mathbb{N}^+$ modelliert werden.
- Eine Kapazitätsfunktion $c : E \mapsto \mathbb{R}^+$ führt zu "Problemen".

Bemerkungen zum Flussproblem

$n = |V|, m = |E|$

- O.B.d.A.: Quelle s hat Eingangsgrad 0.
- O.B.d.A.: Senke t hat Ausgangsgrad 0.
- $f(v, w) = f(e)$ falls $e \in E$ und $e = (v, w)$.
- Kapazitätsbeschränkung:

$$\forall e : f(e) \leqslant c(e)$$

- Schreibweisen:

$$\begin{aligned} f_{in}(v) &= \sum_{(a,v)\in E} f((a, v)) \\ f_{out}(v) &= \sum_{(v,a)\in E} f((v, a)) \end{aligned}$$

- Flusserhaltung:

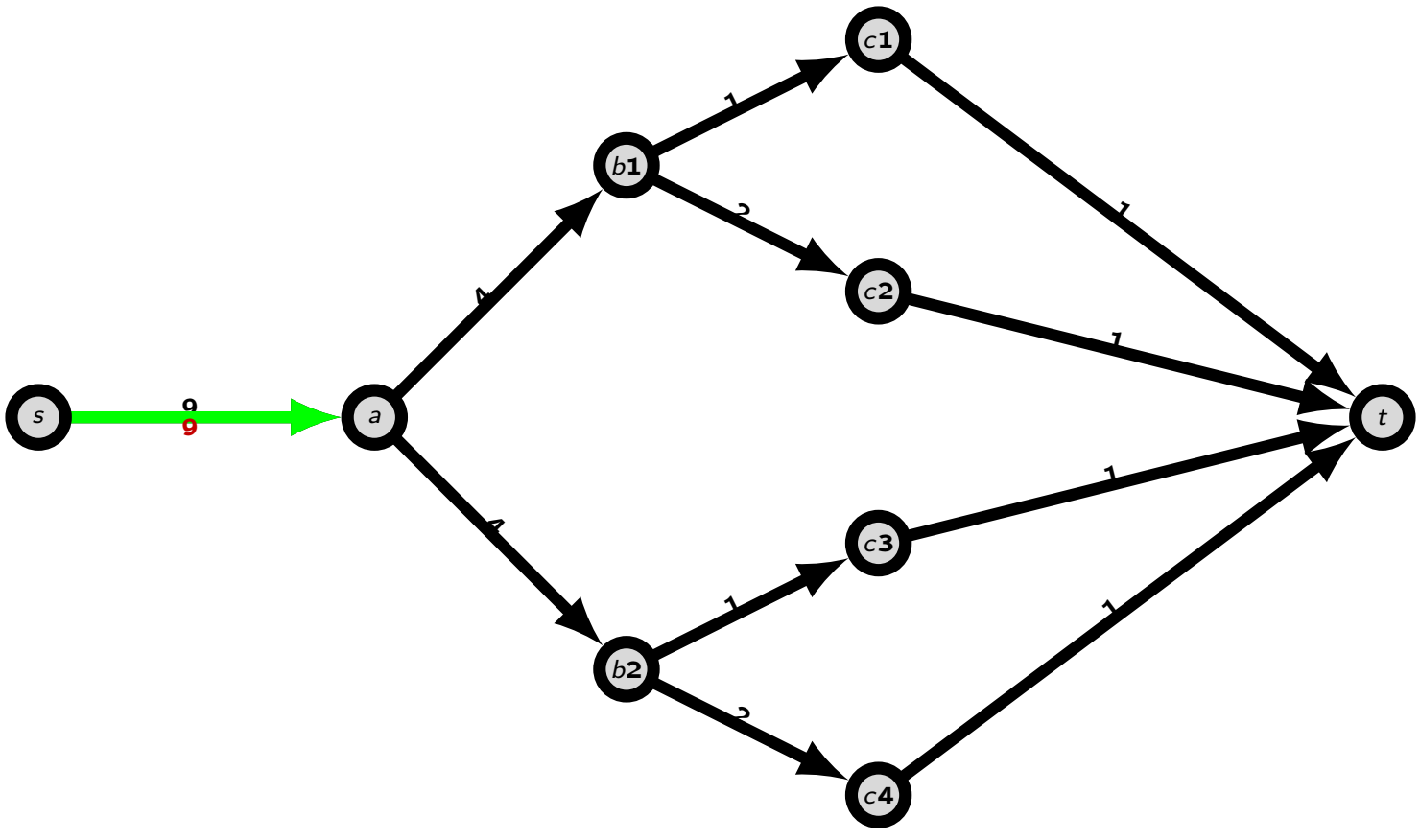
$$\forall v \in V \setminus \{s, t\} : f_{in}(v) = f_{out}(v).$$

- Wert des Flusses f ist: $w(f) = f_{out}(s) = f_{in}(t)$.
- Eine Kapazitätsfunktion $c : E \mapsto \mathbb{Q}^+$ kann durch $c : E \mapsto \mathbb{N}^+$ modelliert werden.
- Eine Kapazitätsfunktion $c : E \mapsto \mathbb{R}^+$ führt zu "Problemen".

Versuch am einfachen Beispiel

$n = |V|, m = |E|$

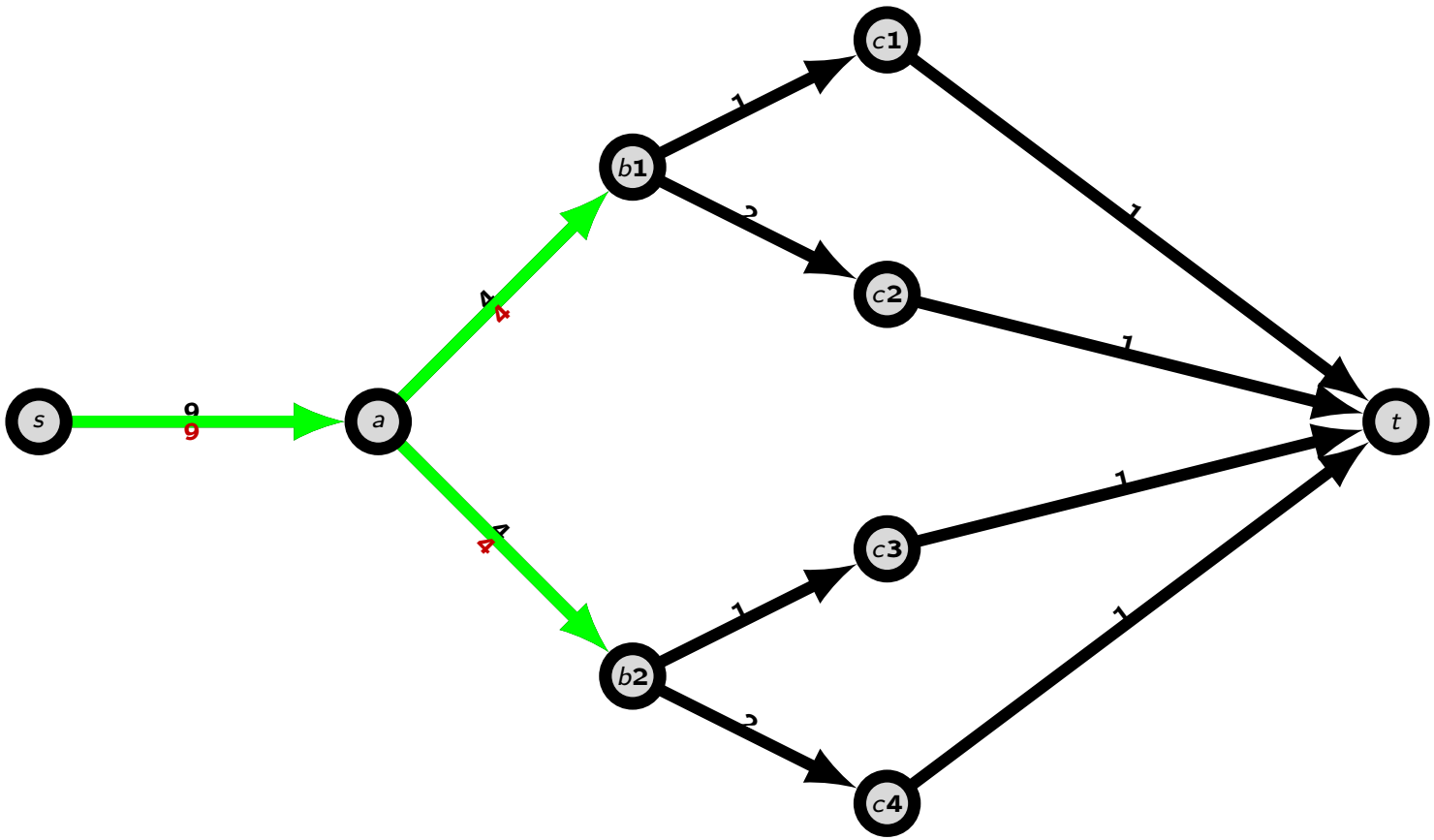
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

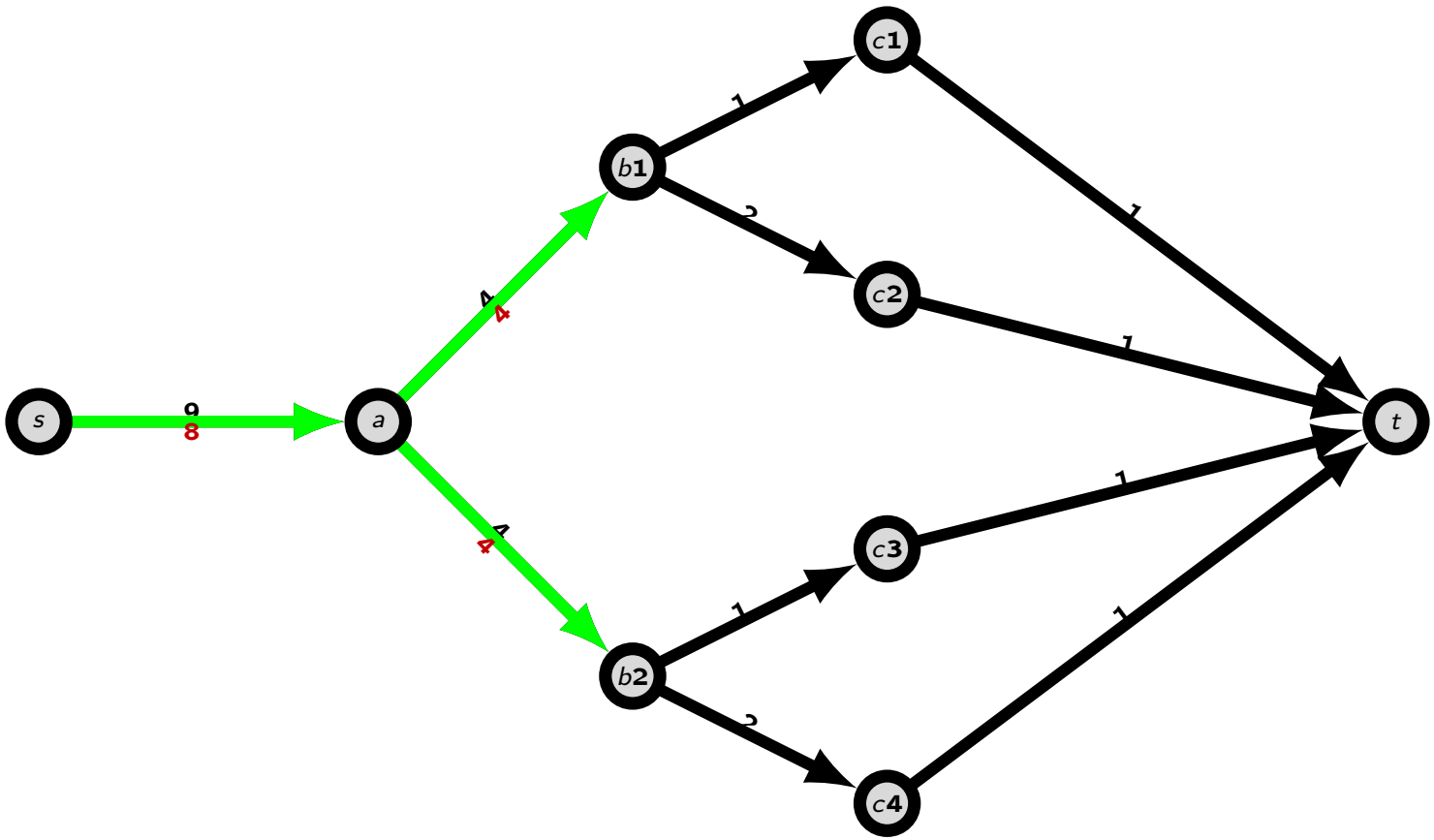
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

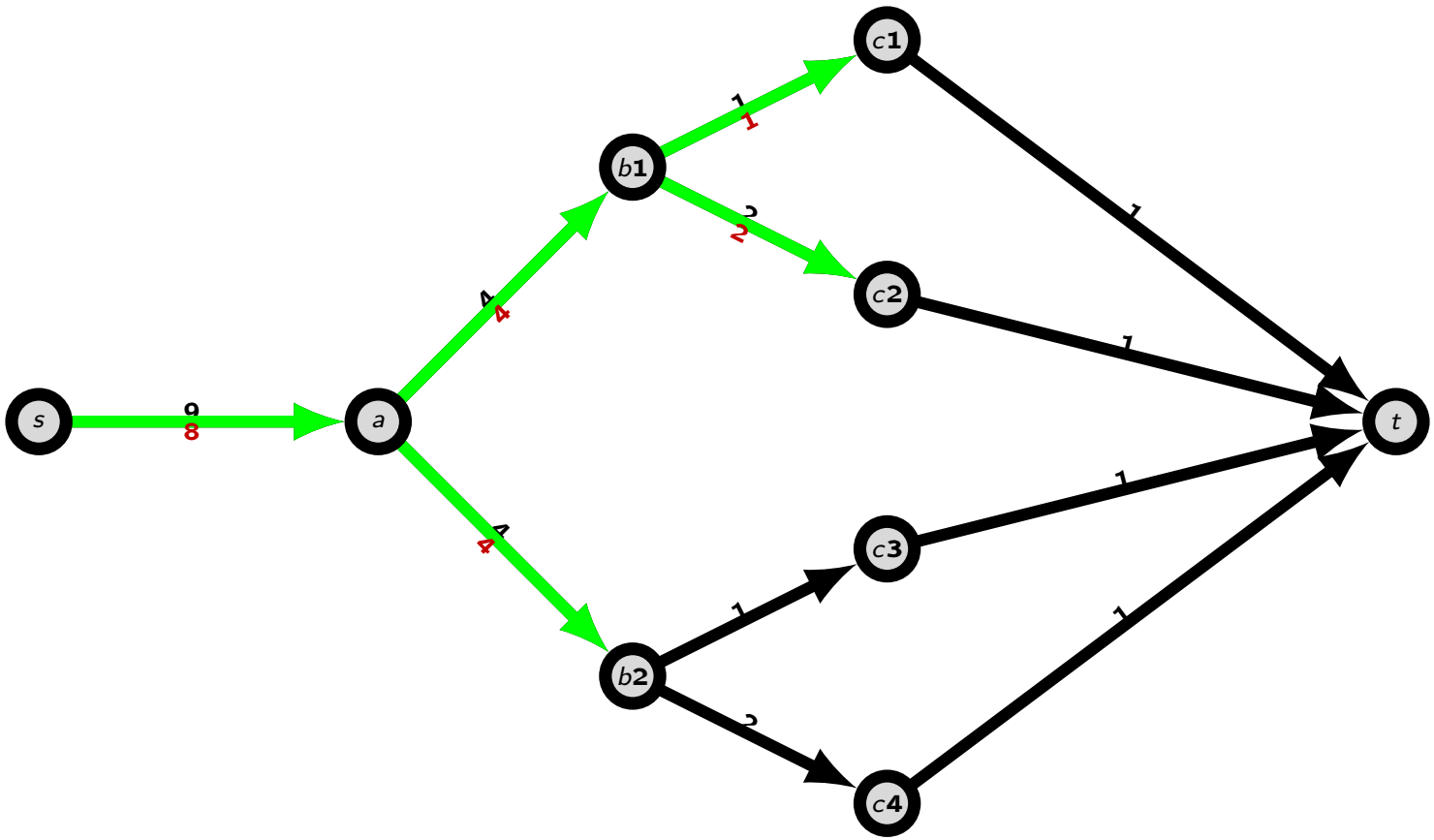
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

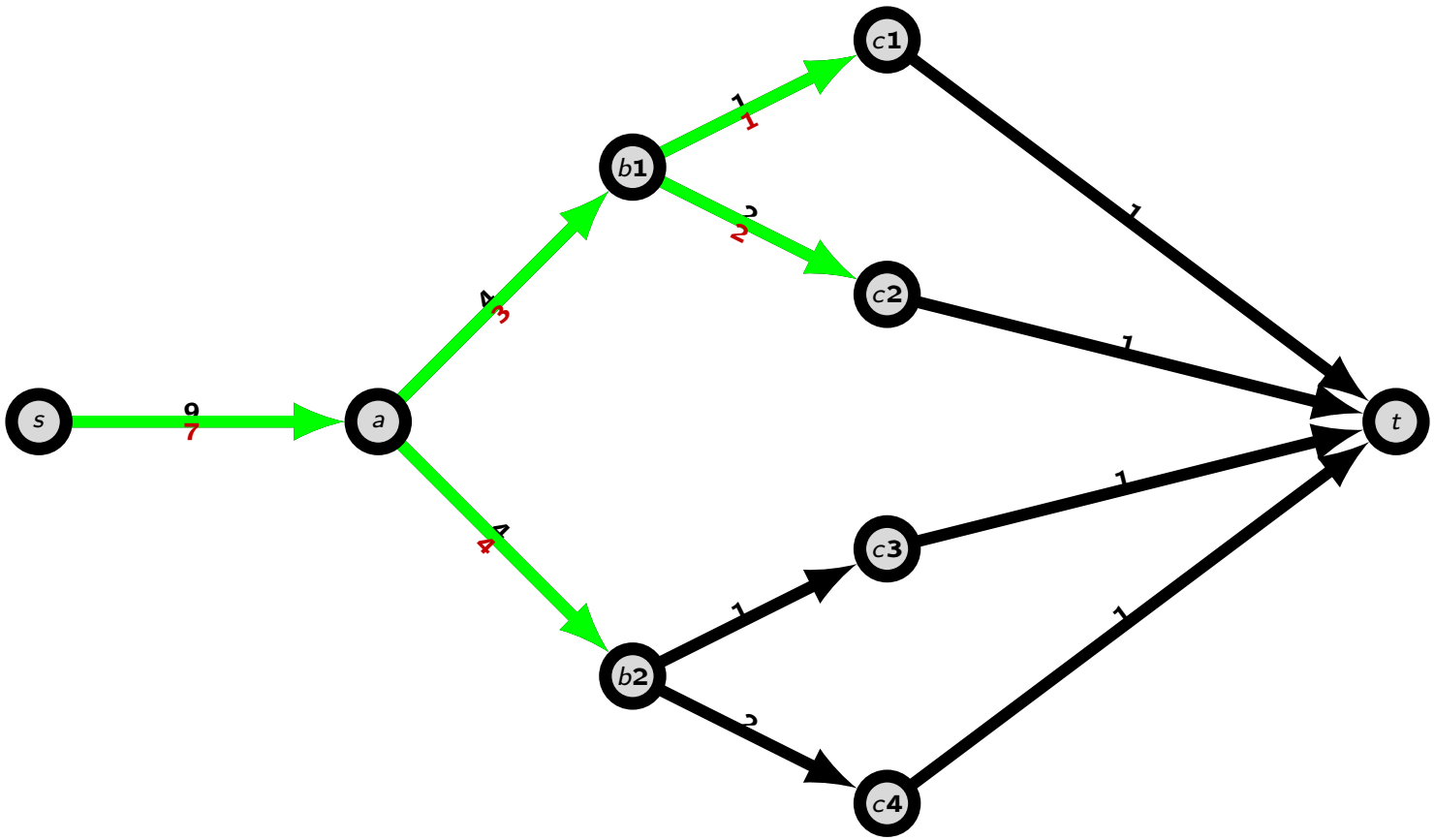
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

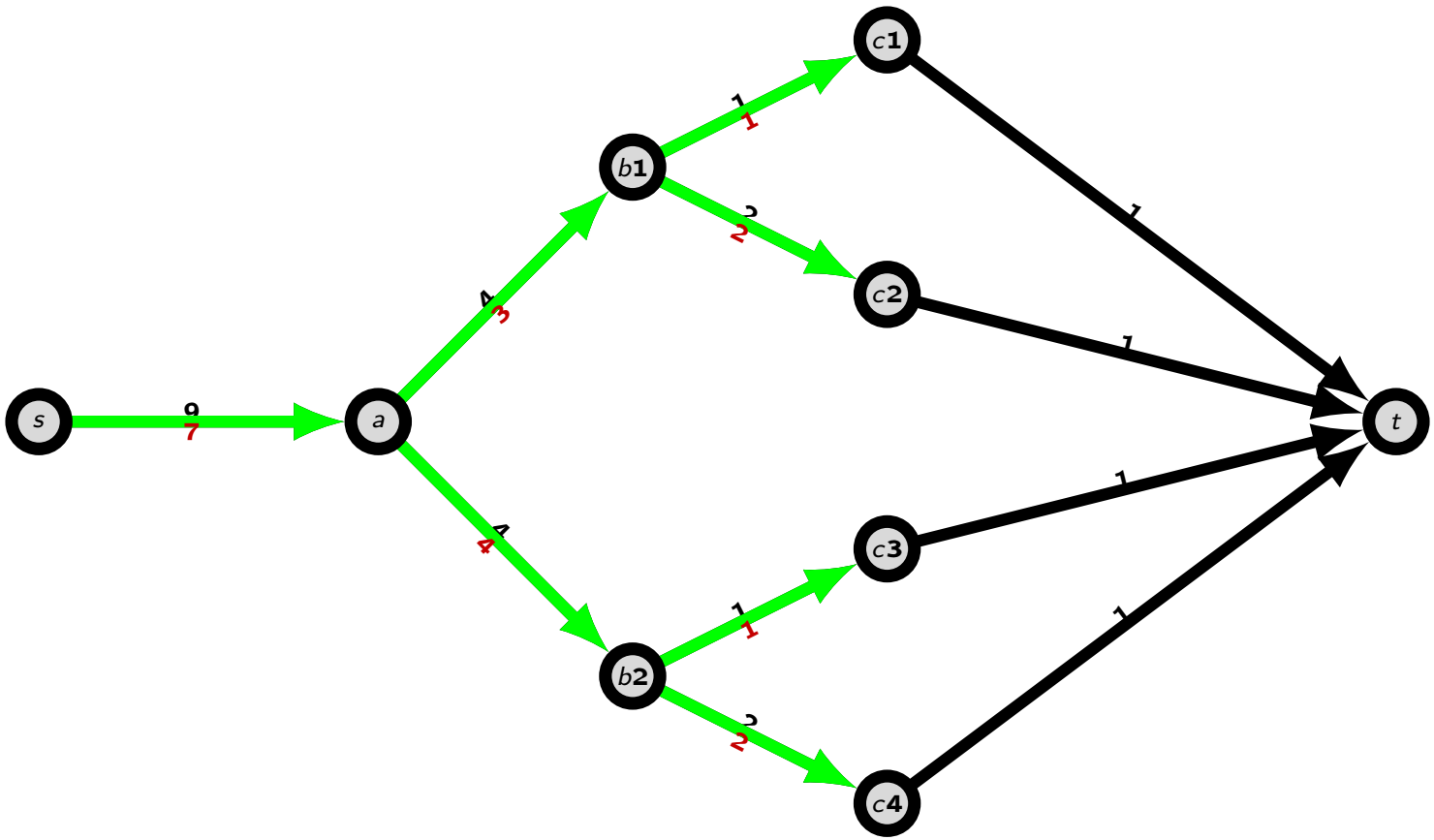
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

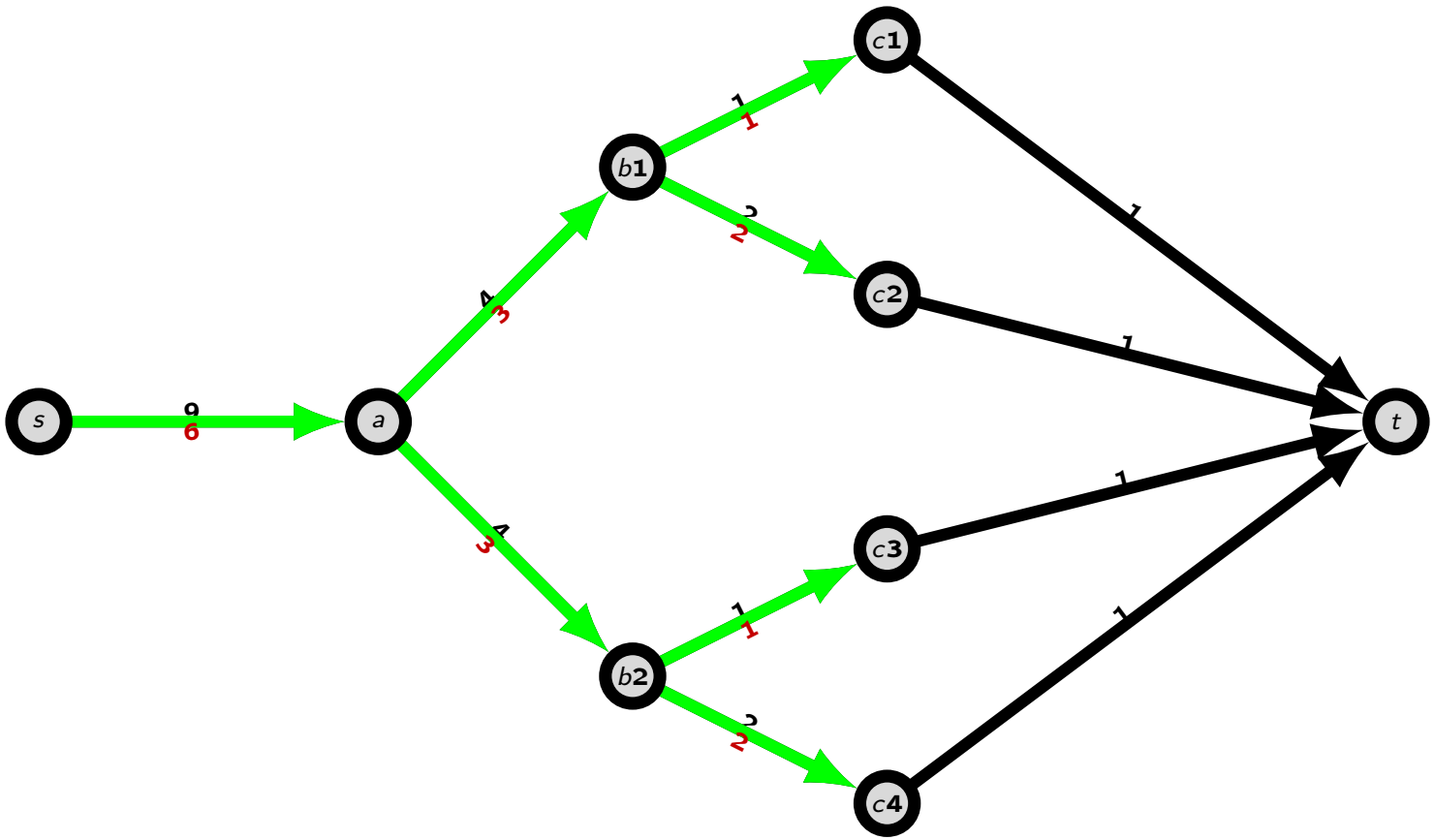
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

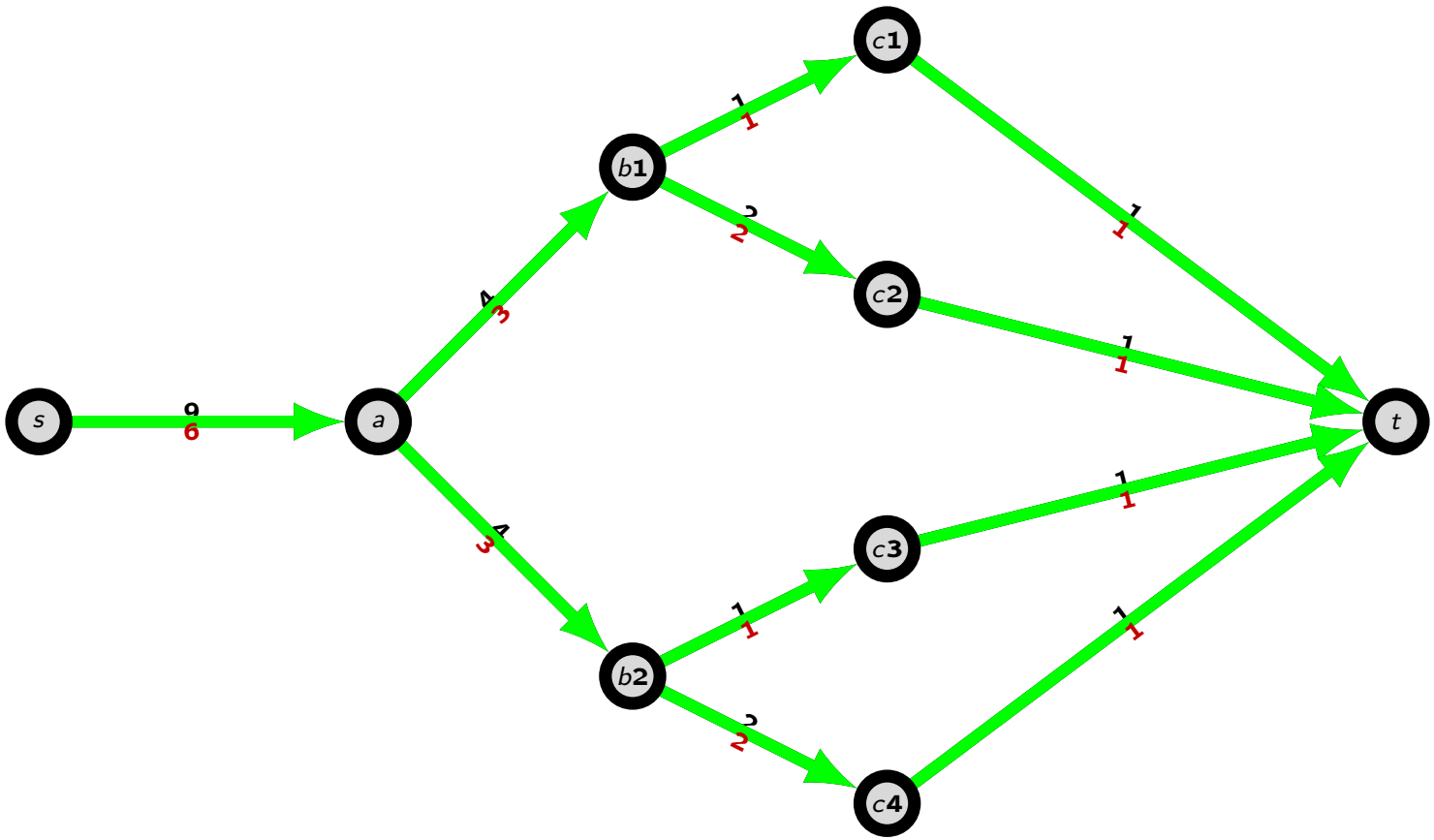
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

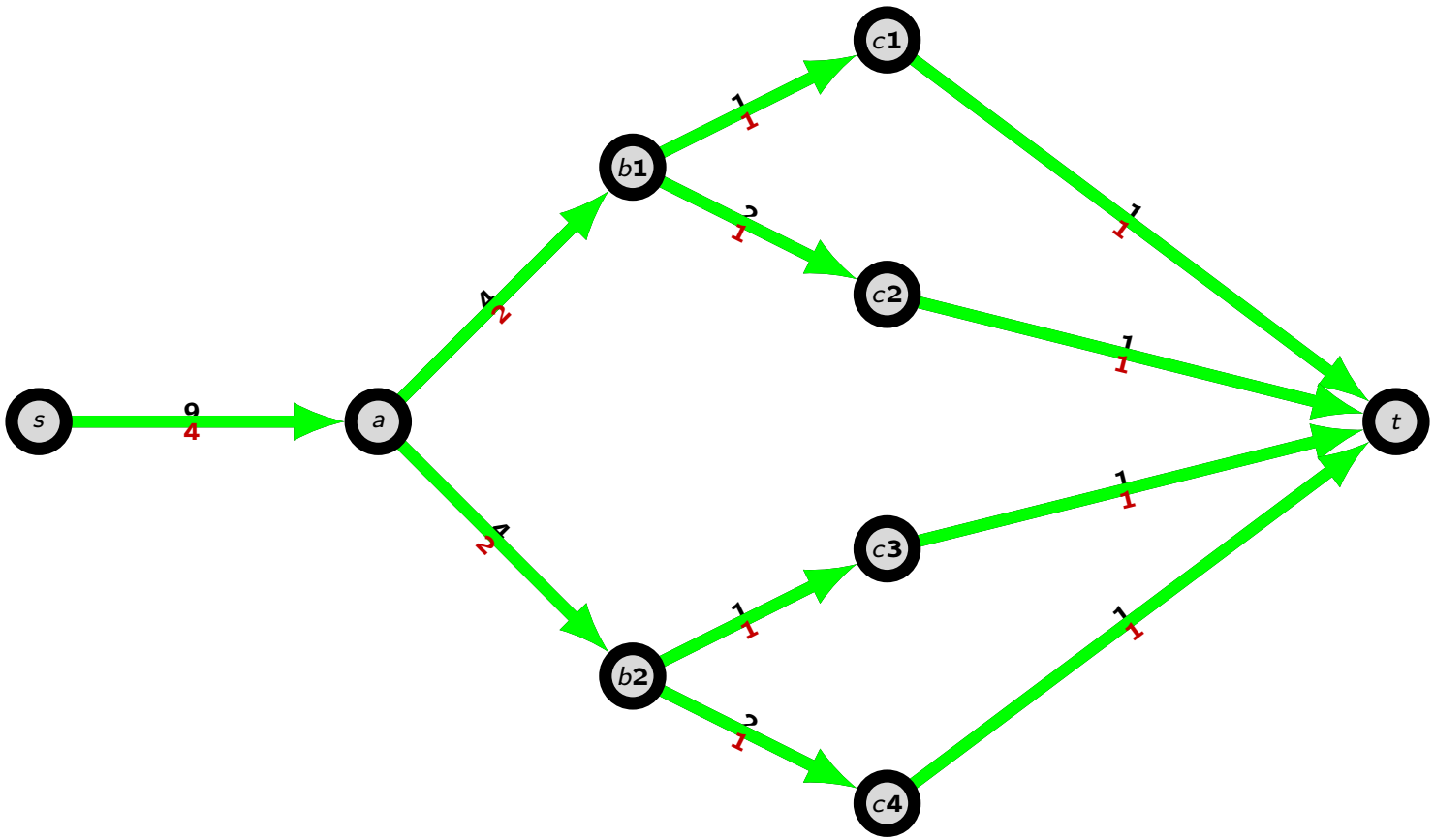
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am einfachen Beispiel

$n = |V|, m = |E|$

Idee: verteile möglichst großen Fluss von der Quelle aus.

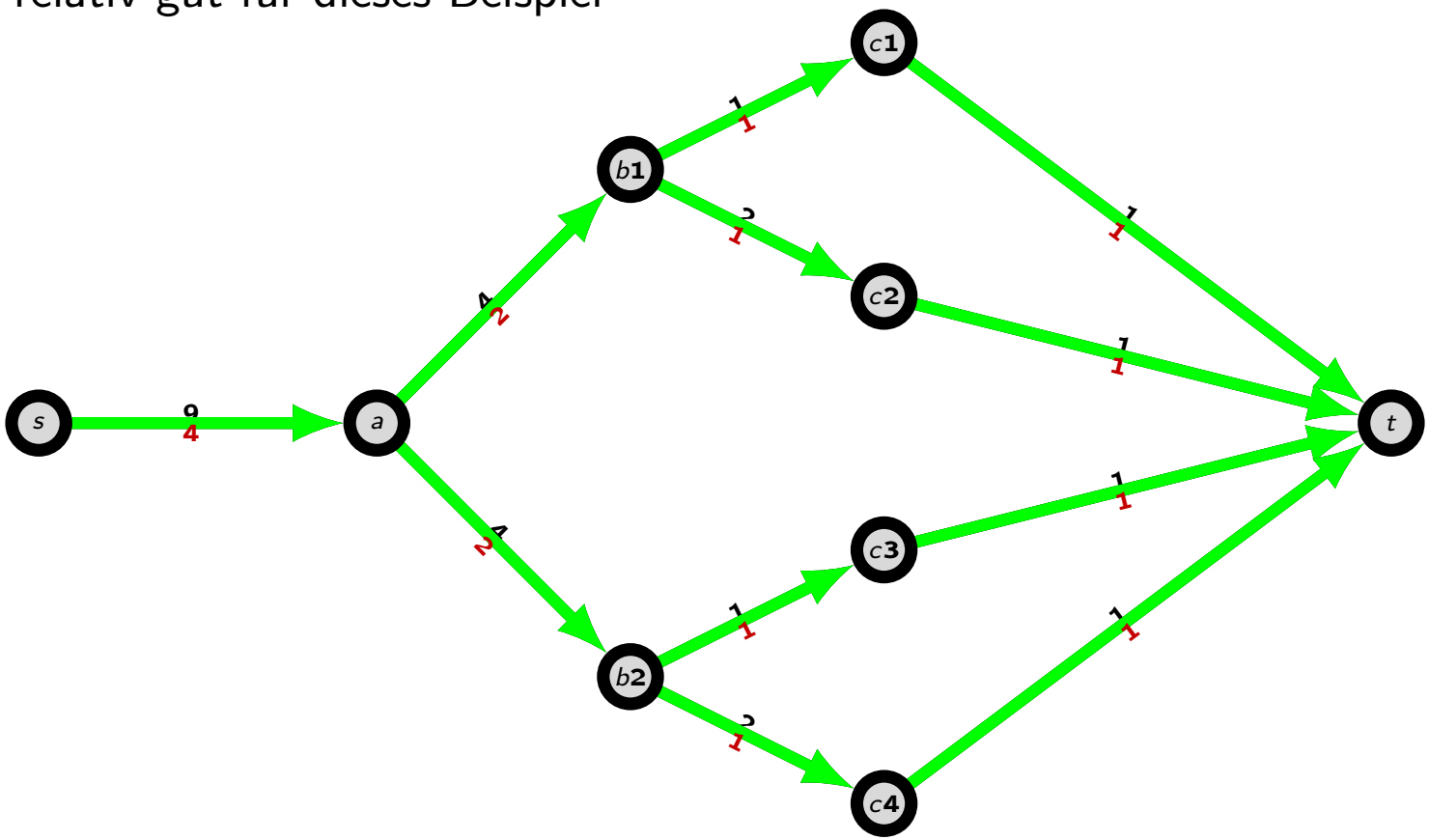


Versuch am einfachen Beispiel

$n = |V|, m = |E|$

Idee: verteile möglichst großen Fluss von der Quelle aus.

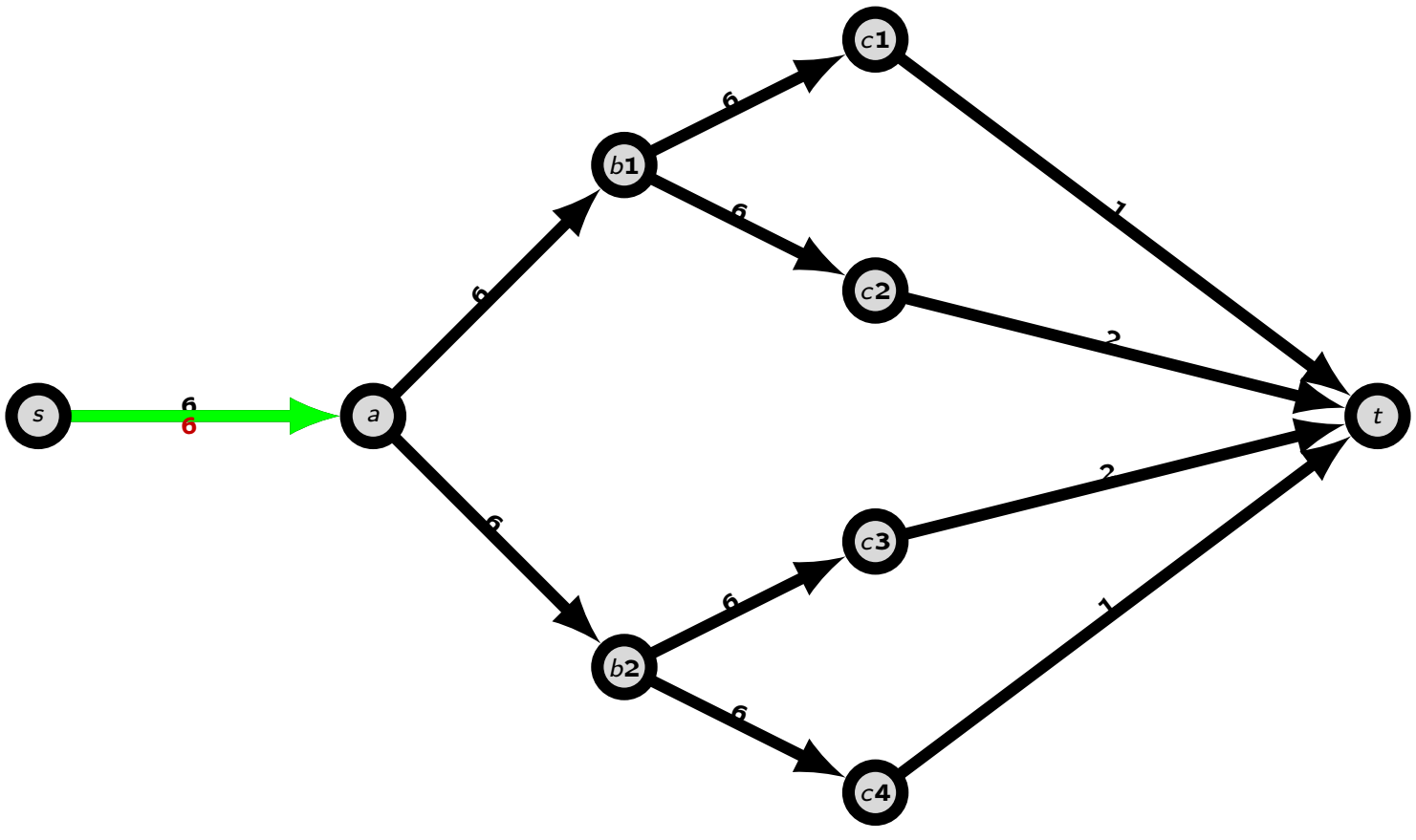
Geht relativ gut für dieses Beispiel



Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

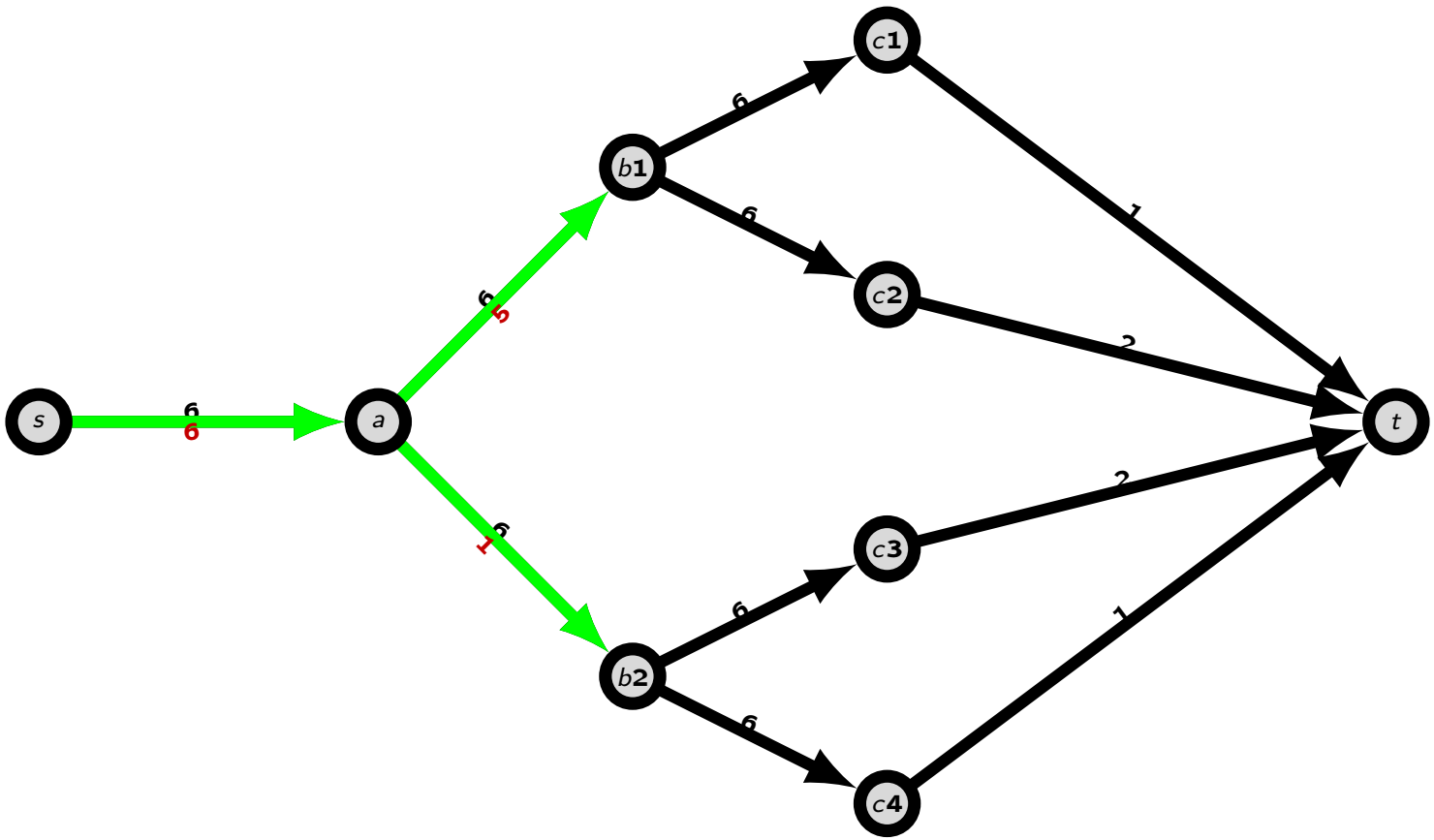
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

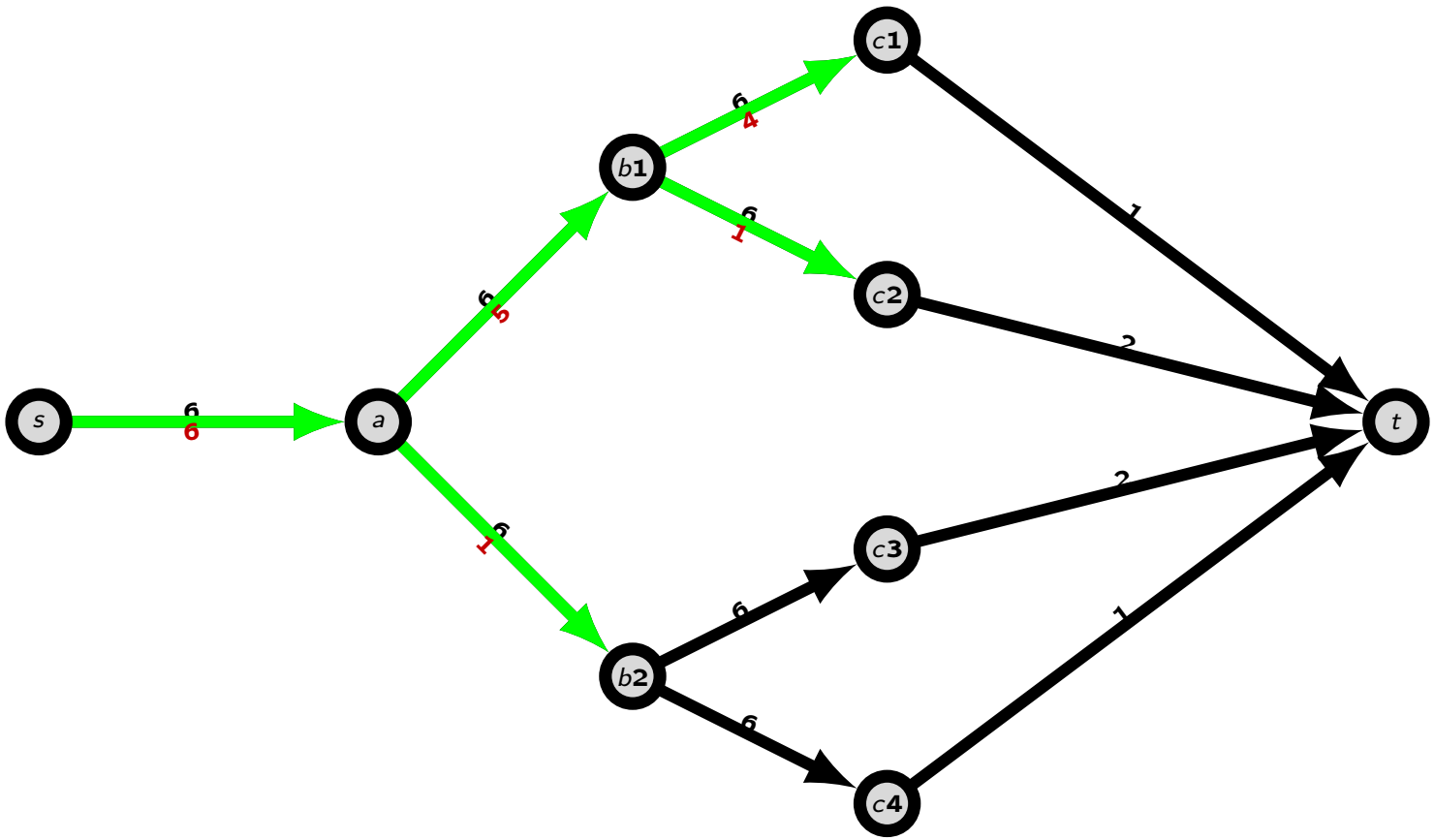
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

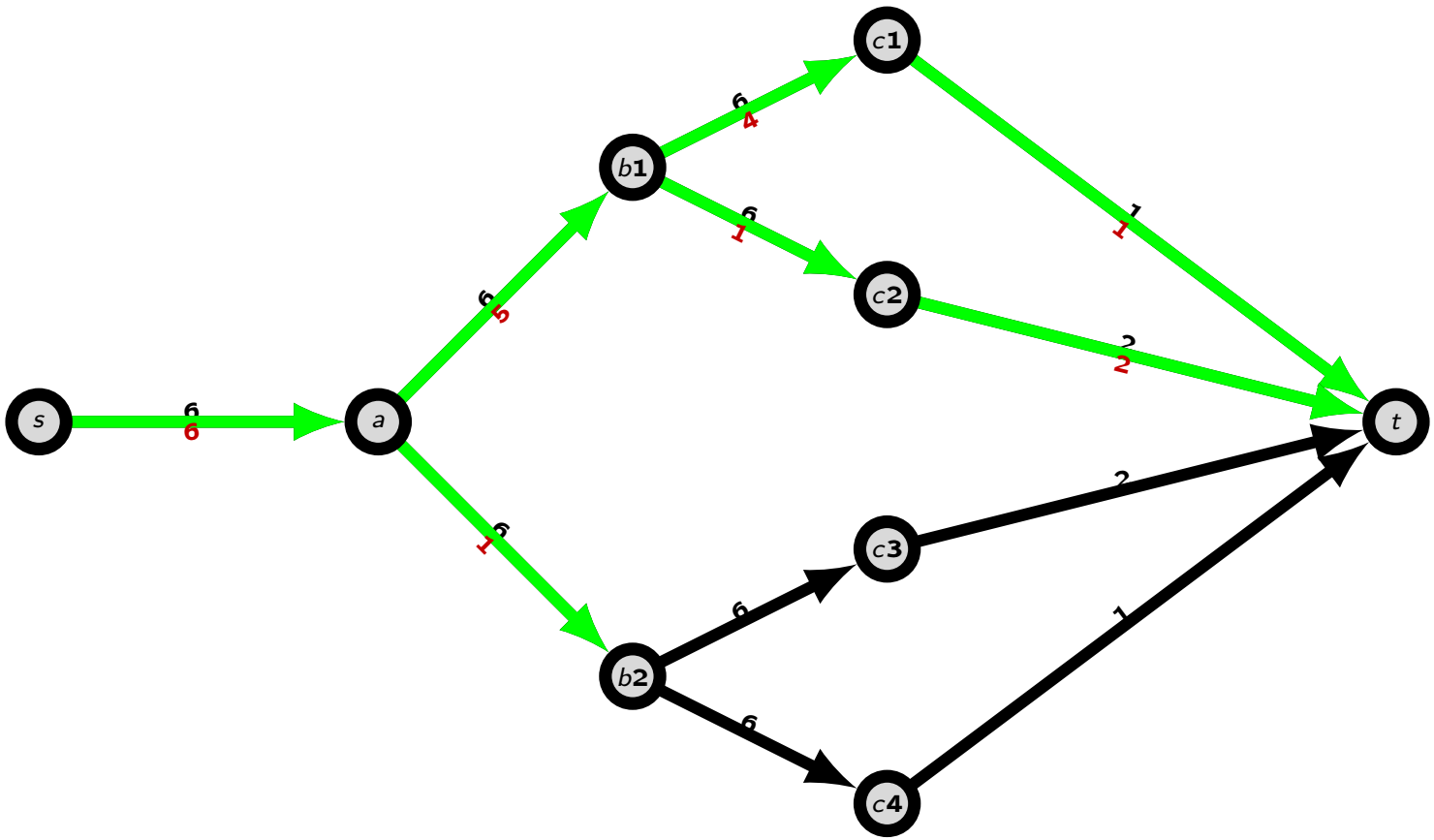
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

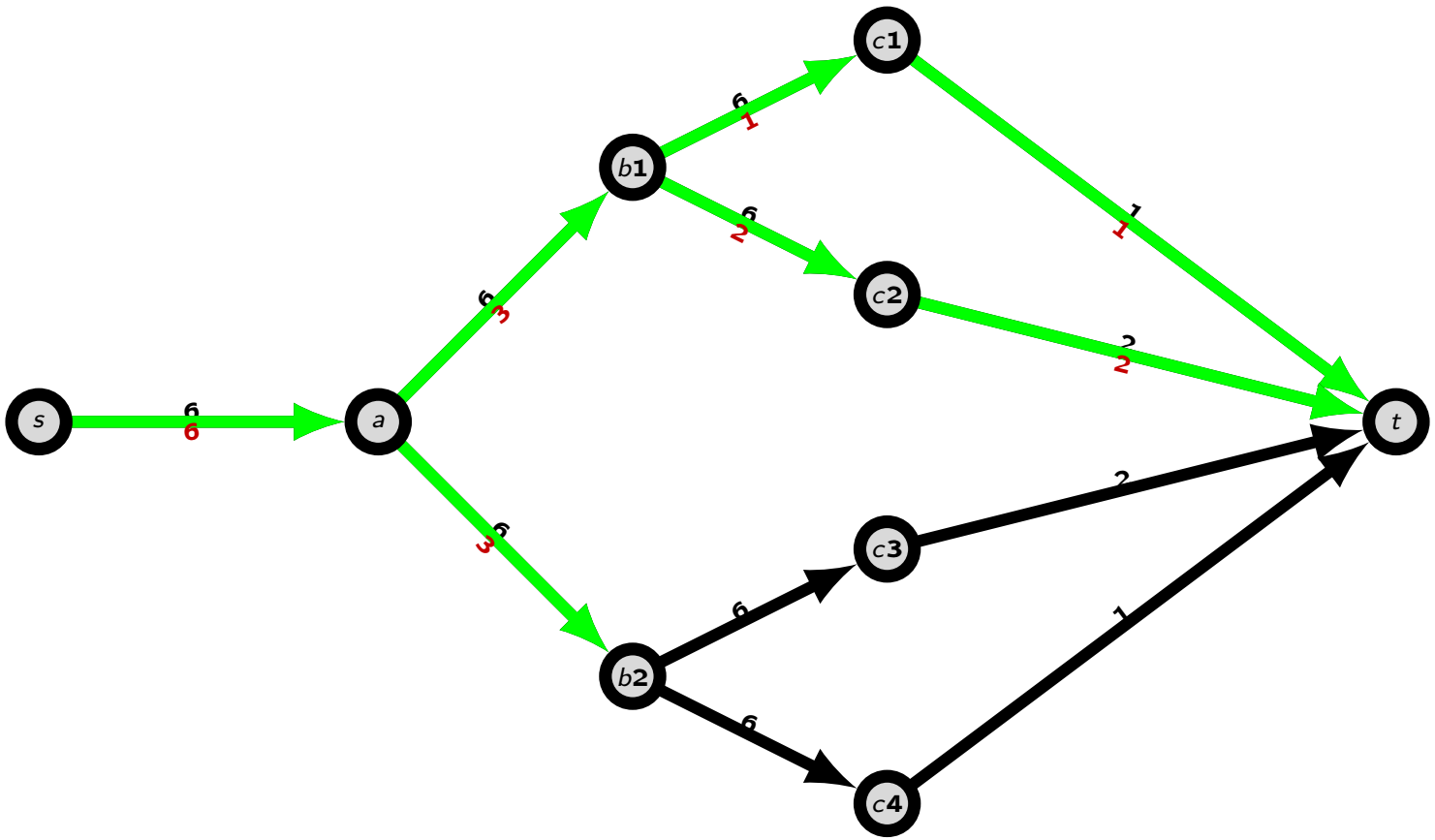
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

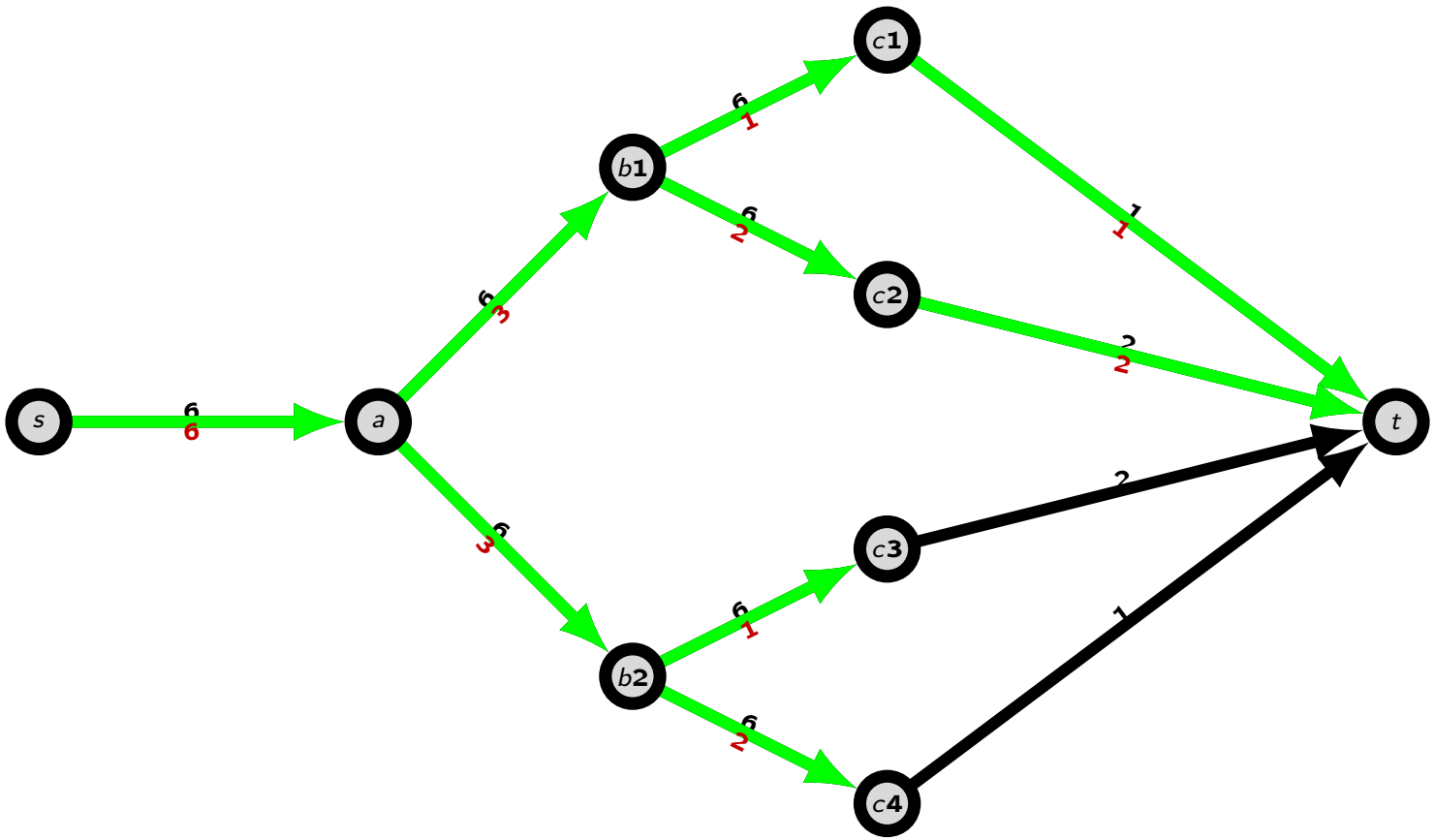
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

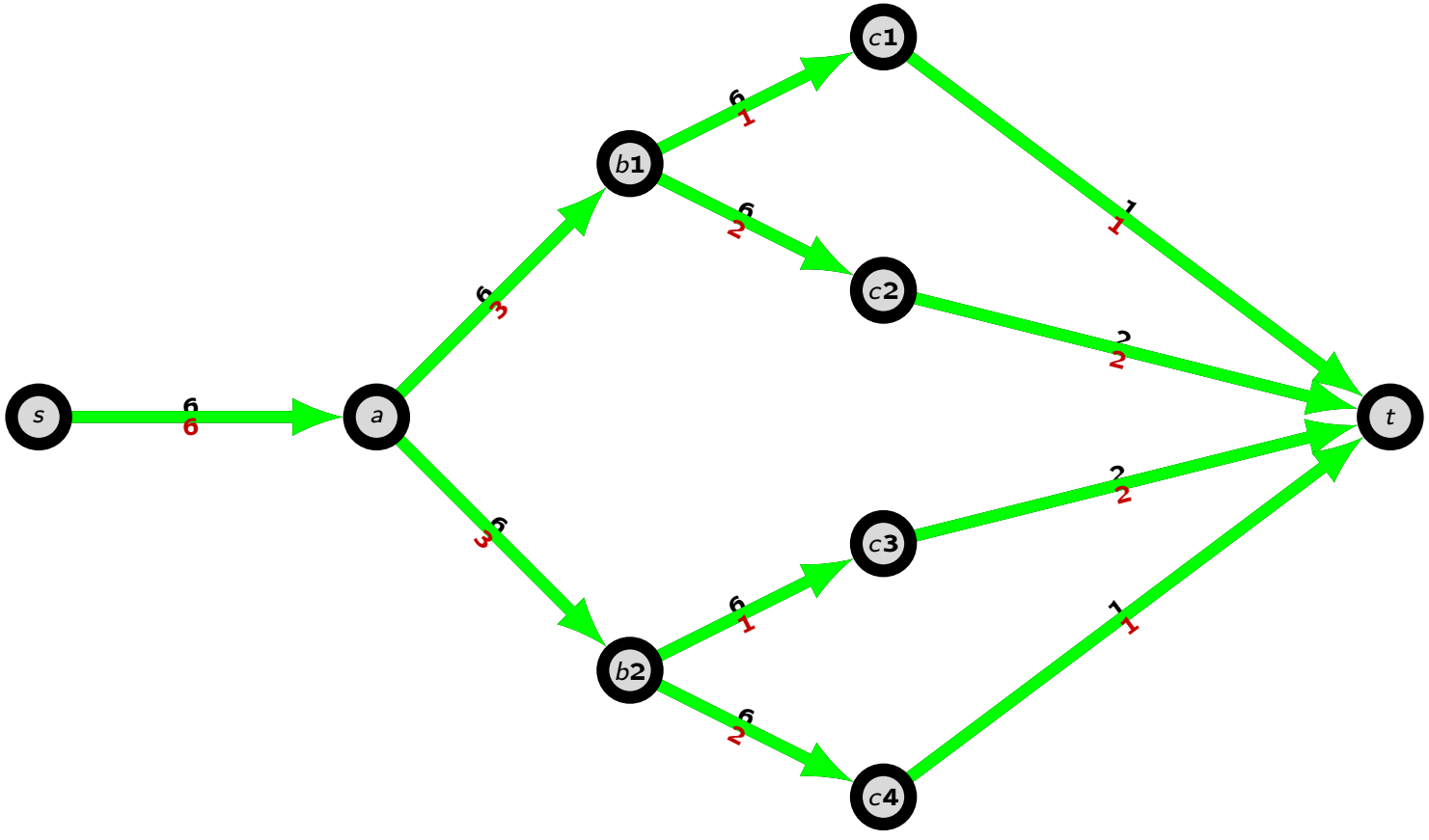
Idee: verteile möglichst großen Fluss von der Quelle aus.



Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

Idee: verteile möglichst großen Fluss von der Quelle aus.

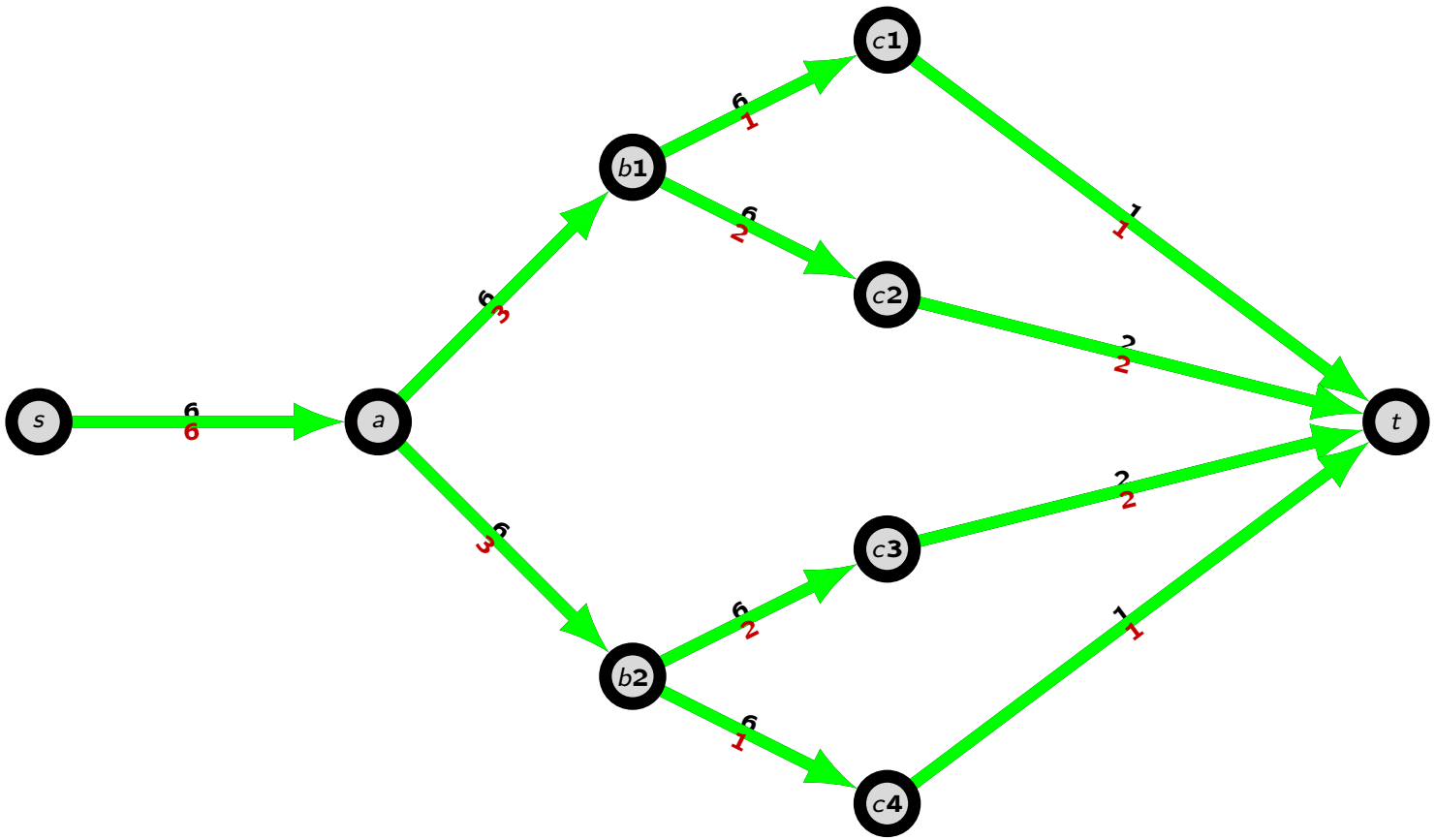


Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

Idee: verteile möglichst großen Fluss von der Quelle aus.

Geht nicht so gut für dieses Beispiel.

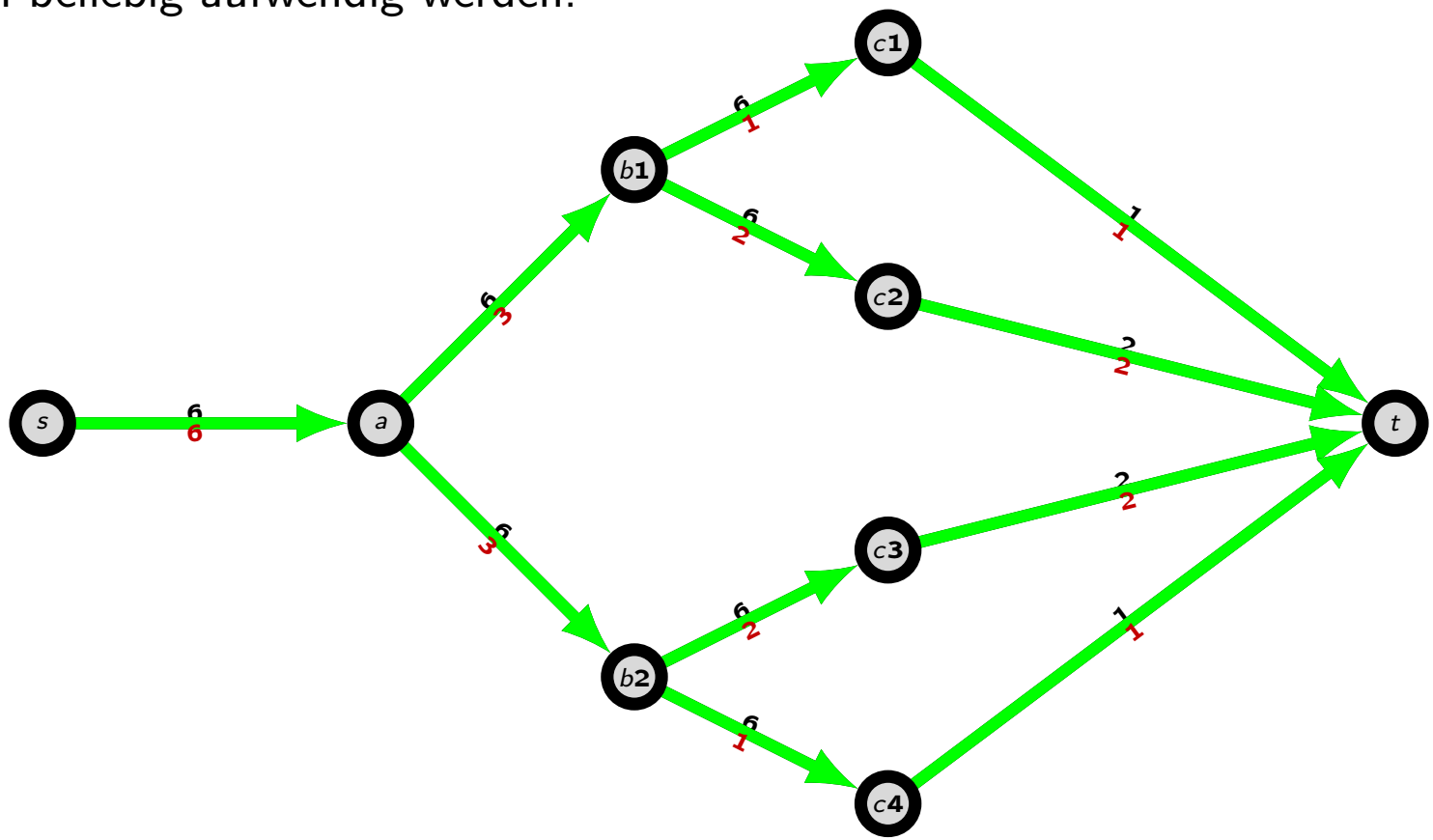


Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

Idee: verteile möglichst großen Fluss von der Quelle aus.

Geht nicht so gut für dieses Beispiel.
Kann beliebig aufwendig werden.

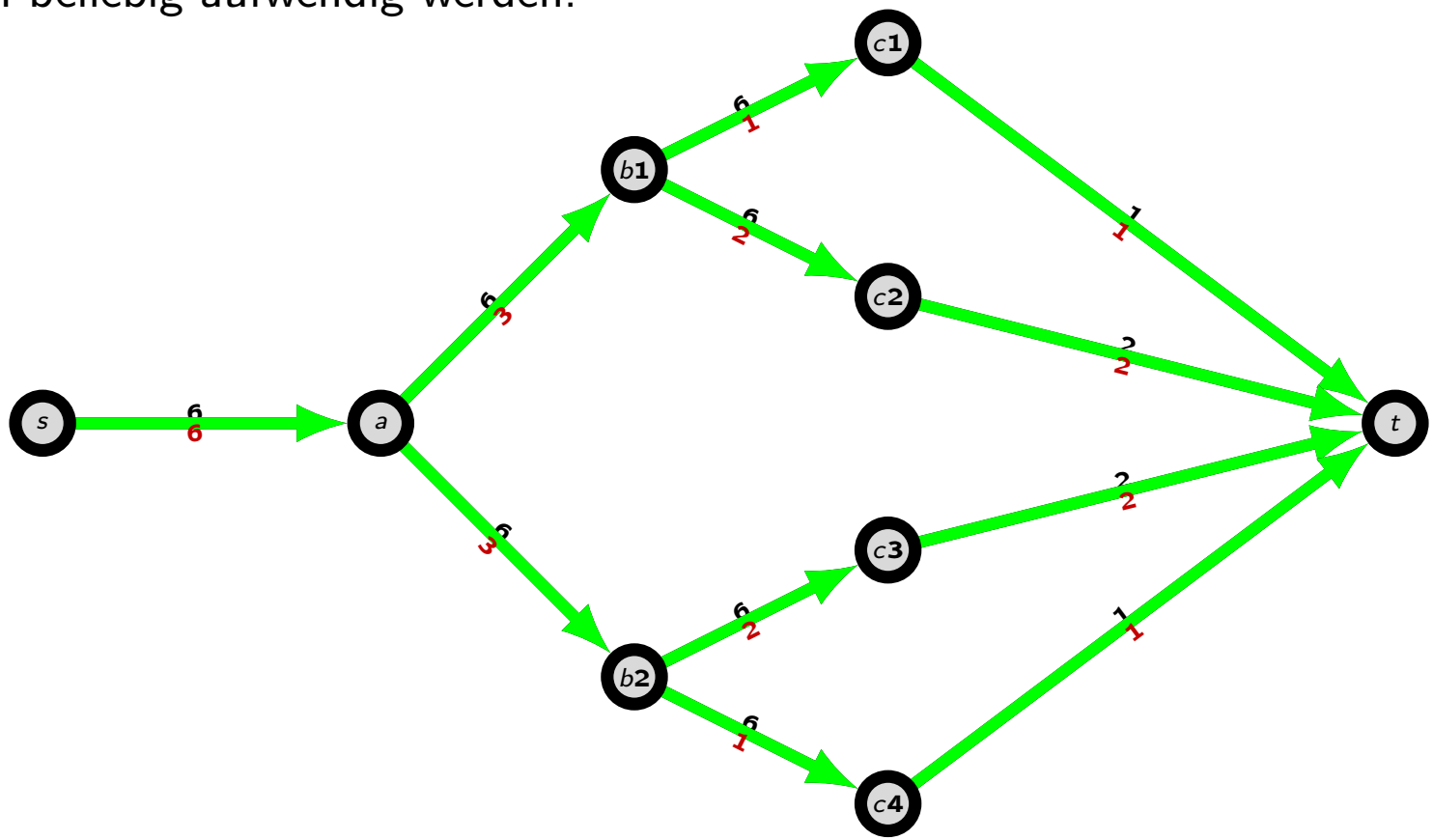


Versuch am aufwendigeren Beispiel

$n = |V|, m = |E|$

Idee: verteile möglichst großen Fluss von der Quelle aus.

Geht nicht so gut für dieses Beispiel.
Kann beliebig aufwendig werden.



Situation

$n = |V|, m = |E|$

- Ein “direktes” Bestimmen des Flusses von der Quelle aus scheitert.

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).
- Algorithmus:

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).
- Algorithmus:
  1. Starte mit dem leeren Fluss.

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).
- Algorithmus:
  1. Starte mit dem leeren Fluss.
  2. Bestimme einen Pfad P, auf dem der Fluss von s nach t vergrößerbar ist.

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).
- Algorithmus:
  1. Starte mit dem leeren Fluss.
  2. Bestimme einen Pfad P, auf dem der Fluss von s nach t vergrößerbar ist.
  3. Bestimme den maximalen vergrößernden Fluss p auf P.

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).
- Algorithmus:
  1. Starte mit dem leeren Fluss.
  2. Bestimme einen Pfad P, auf dem der Fluss von s nach t vergrößerbar ist.
  3. Bestimme den maximalen vergrößernden Fluss p auf P.
  4. Erweitere die Lösung um p auf dem Pfad P.

Situation

$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).
- Algorithmus:
  1. Starte mit dem leeren Fluss.
  2. Bestimme einen Pfad P, auf dem der Fluss von s nach t vergrößerbar ist.
  3. Bestimme den maximalen vergrößernden Fluss p auf P.
  4. Erweitere die Lösung um p auf dem Pfad P.
  5. Wiederhole so oft, wie es einen passenden Pfad P gibt.

Situation

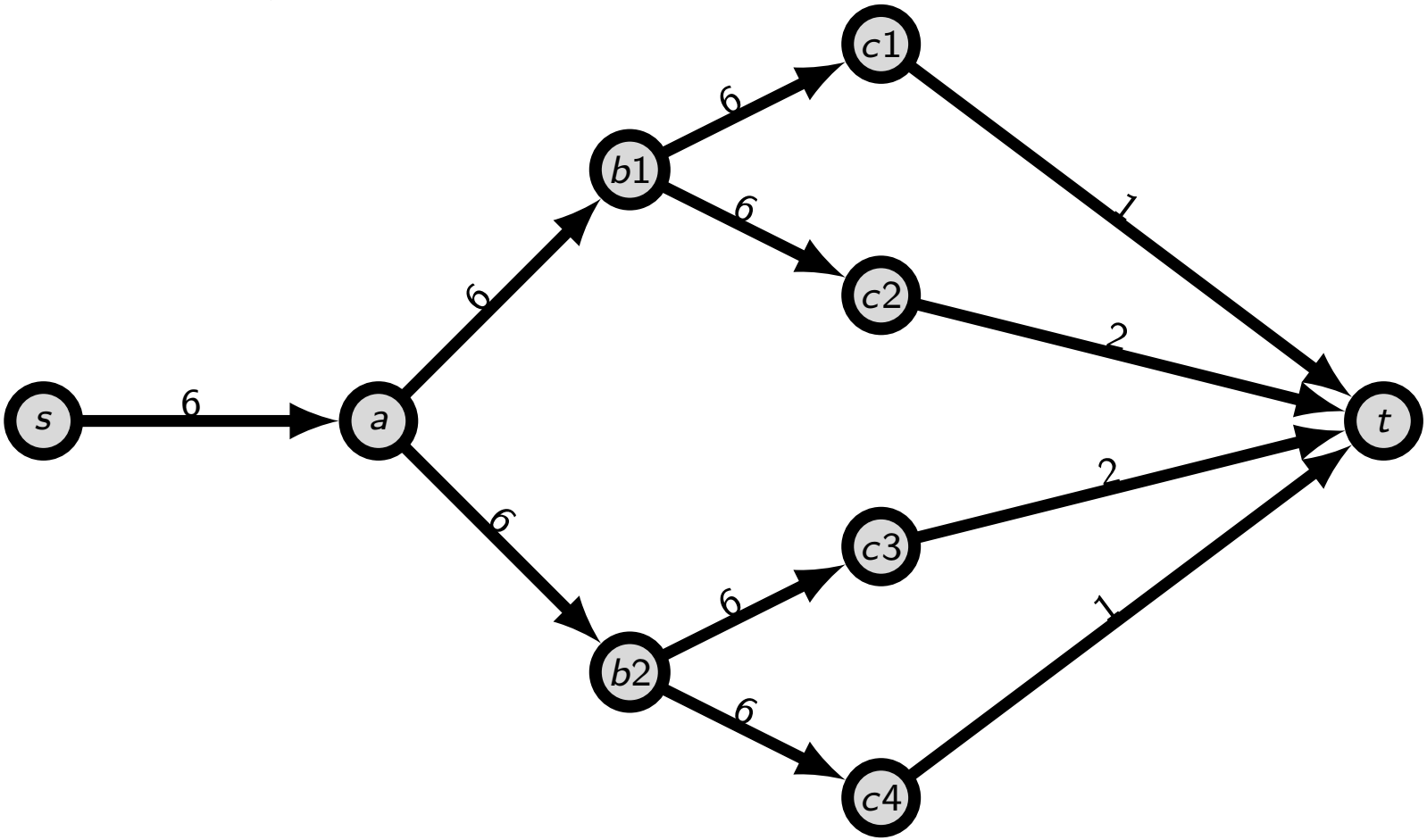
$n = |V|, m = |E|$

- Ein "direktes" Bestimmen des Flusses von der Quelle aus scheitert.
- Es muss eine einfachere Struktur für einen Fluss gesucht werden.
- Die Lösung ergibt sich dann aus mehreren dieser einfachen Strukturen.
- Idee: Wähle als einfache Struktur den Pfad (Weg).
- Algorithmus:
  1. Starte mit dem leeren Fluss.
  2. Bestimme einen Pfad P, auf dem der Fluss von s nach t vergrößerbar ist.
  3. Bestimme den maximalen vergrößernden Fluss p auf P.
  4. Erweitere die Lösung um p auf dem Pfad P.
  5. Wiederhole so oft, wie es einen passenden Pfad P gibt.

Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

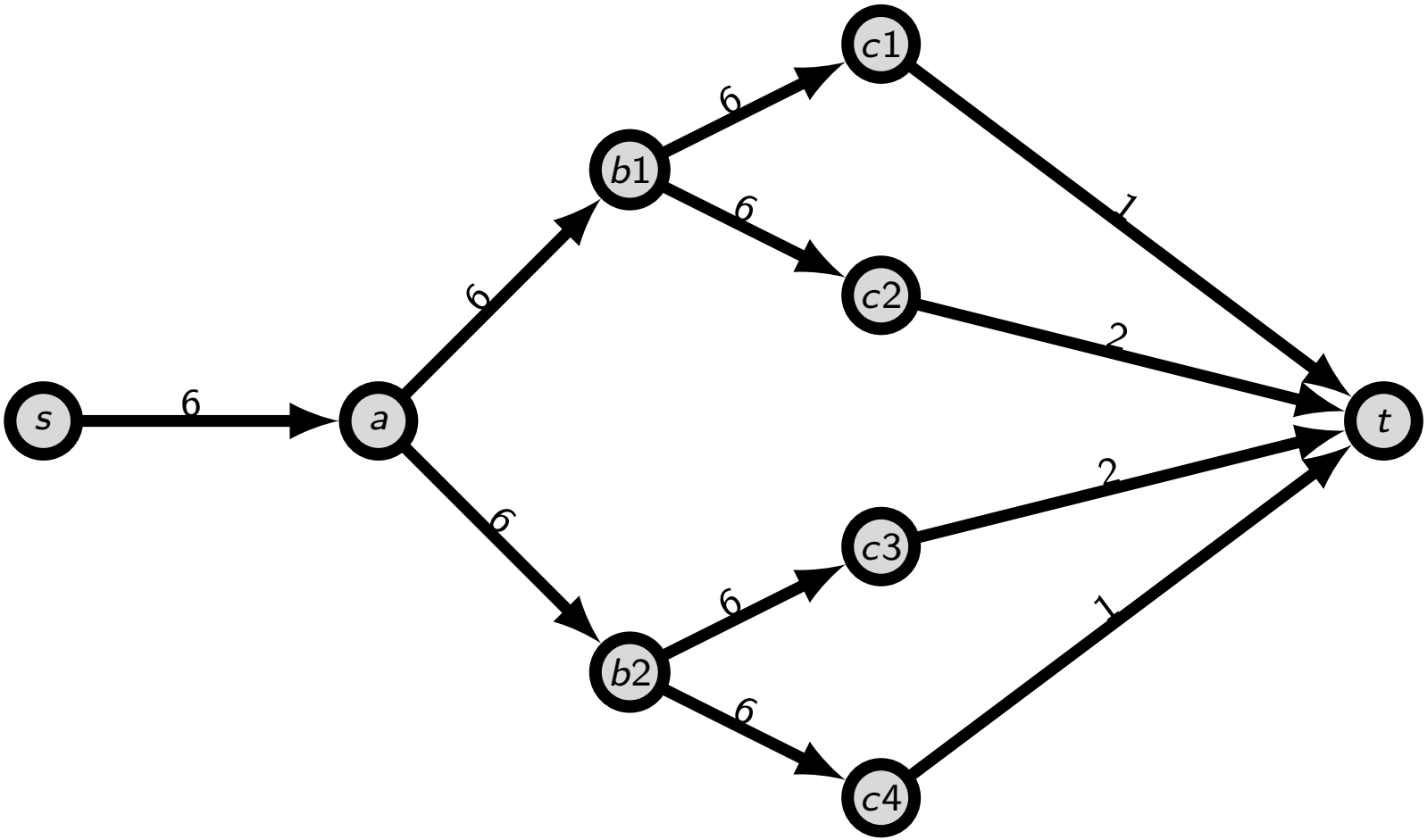
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

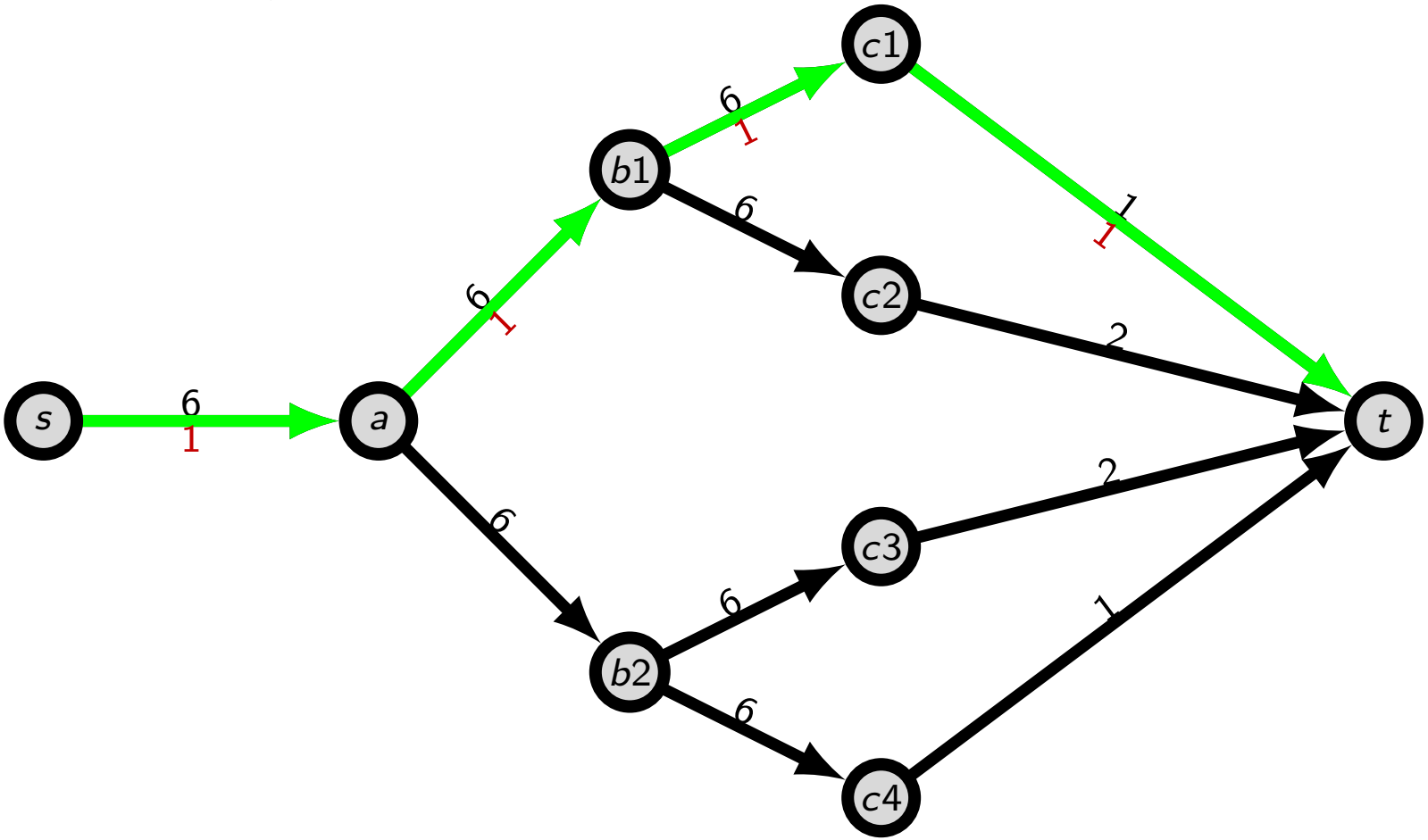
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

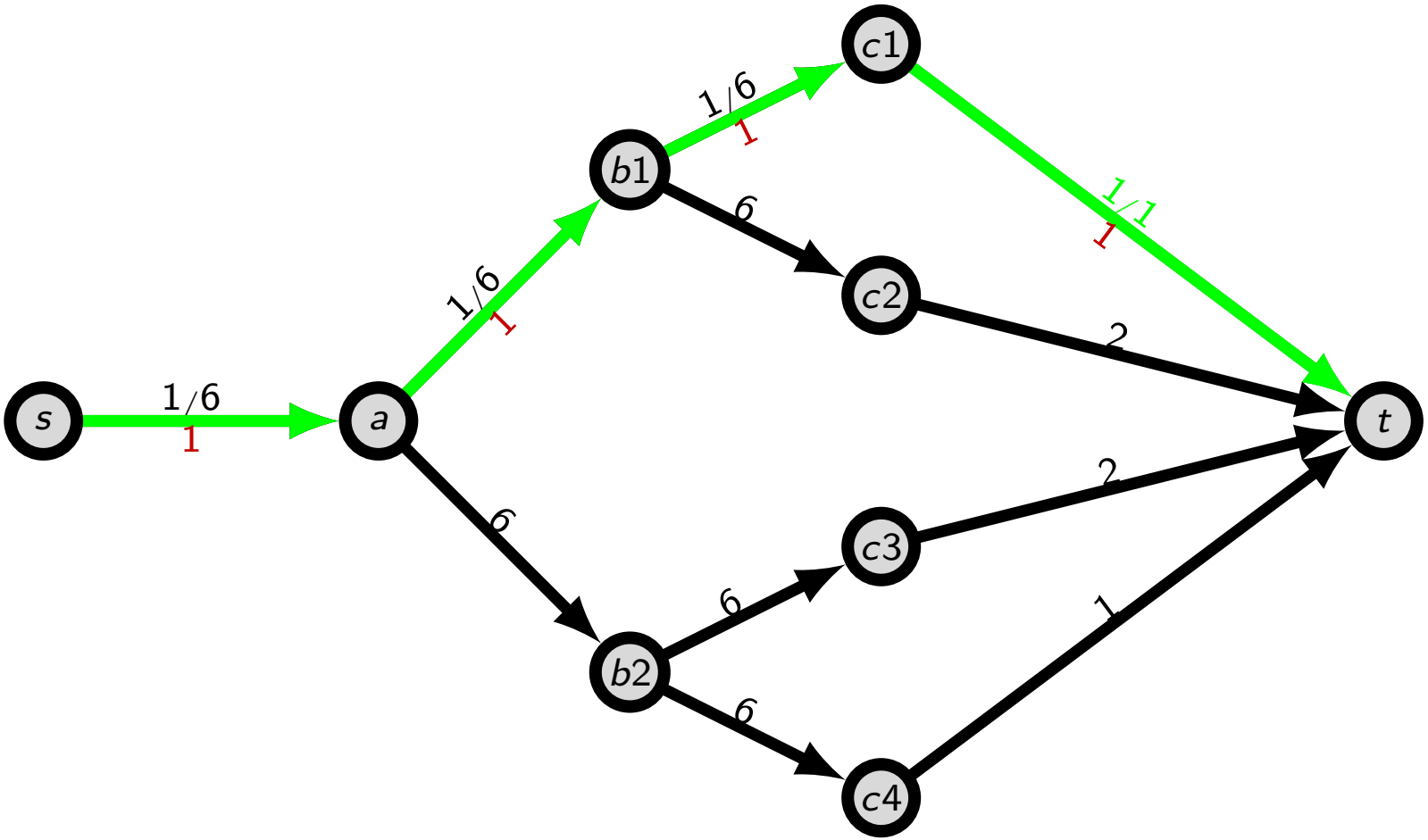
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

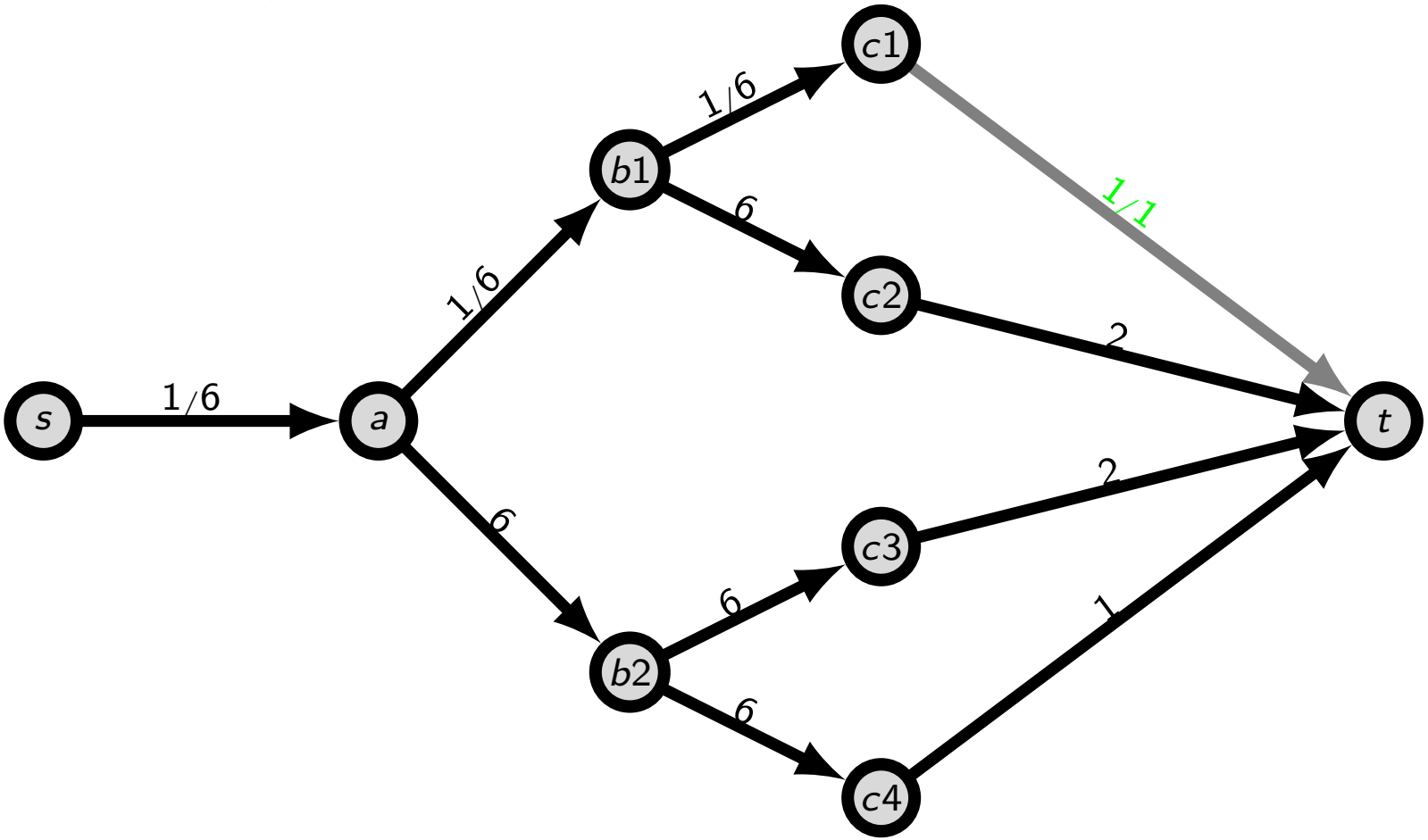
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

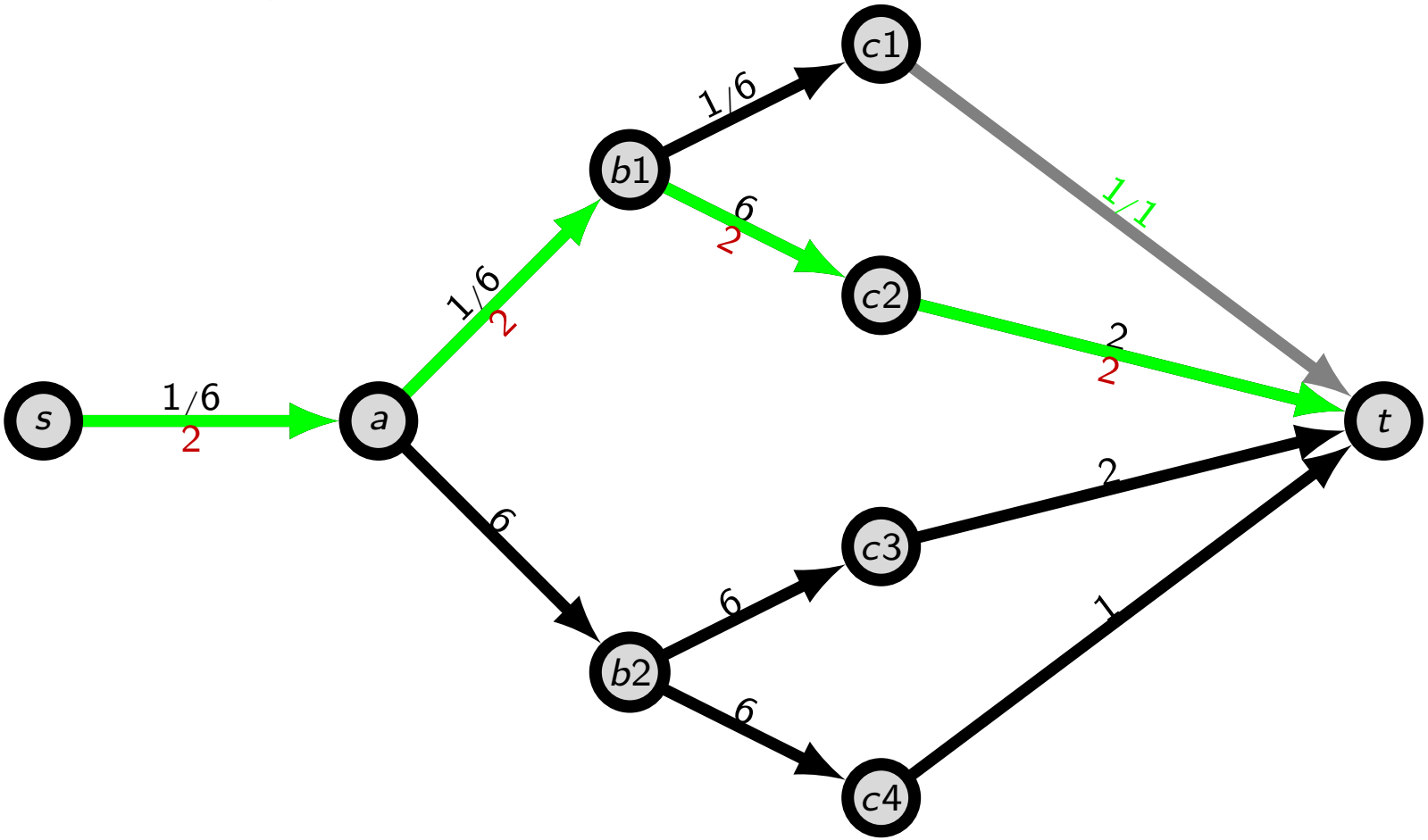
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

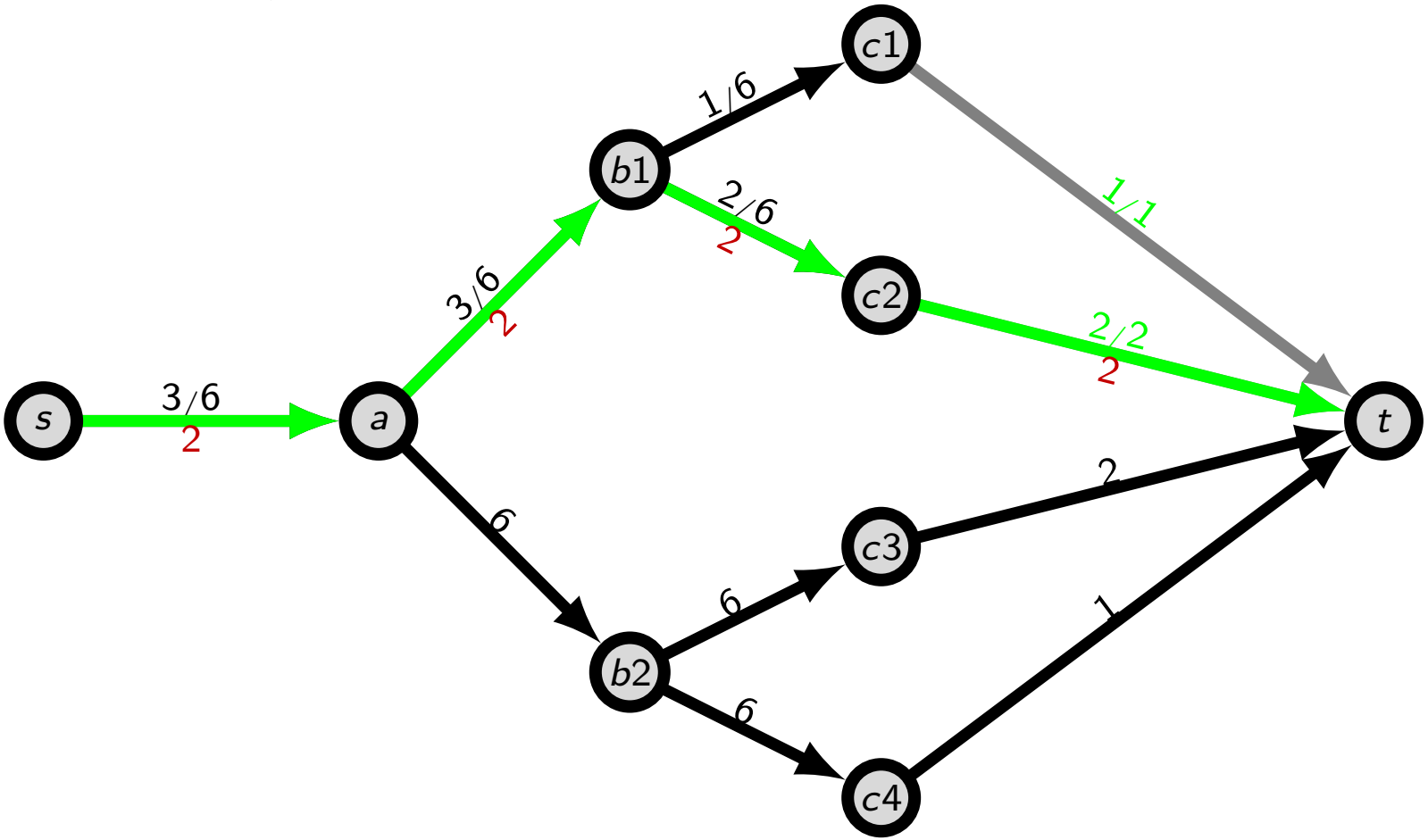
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

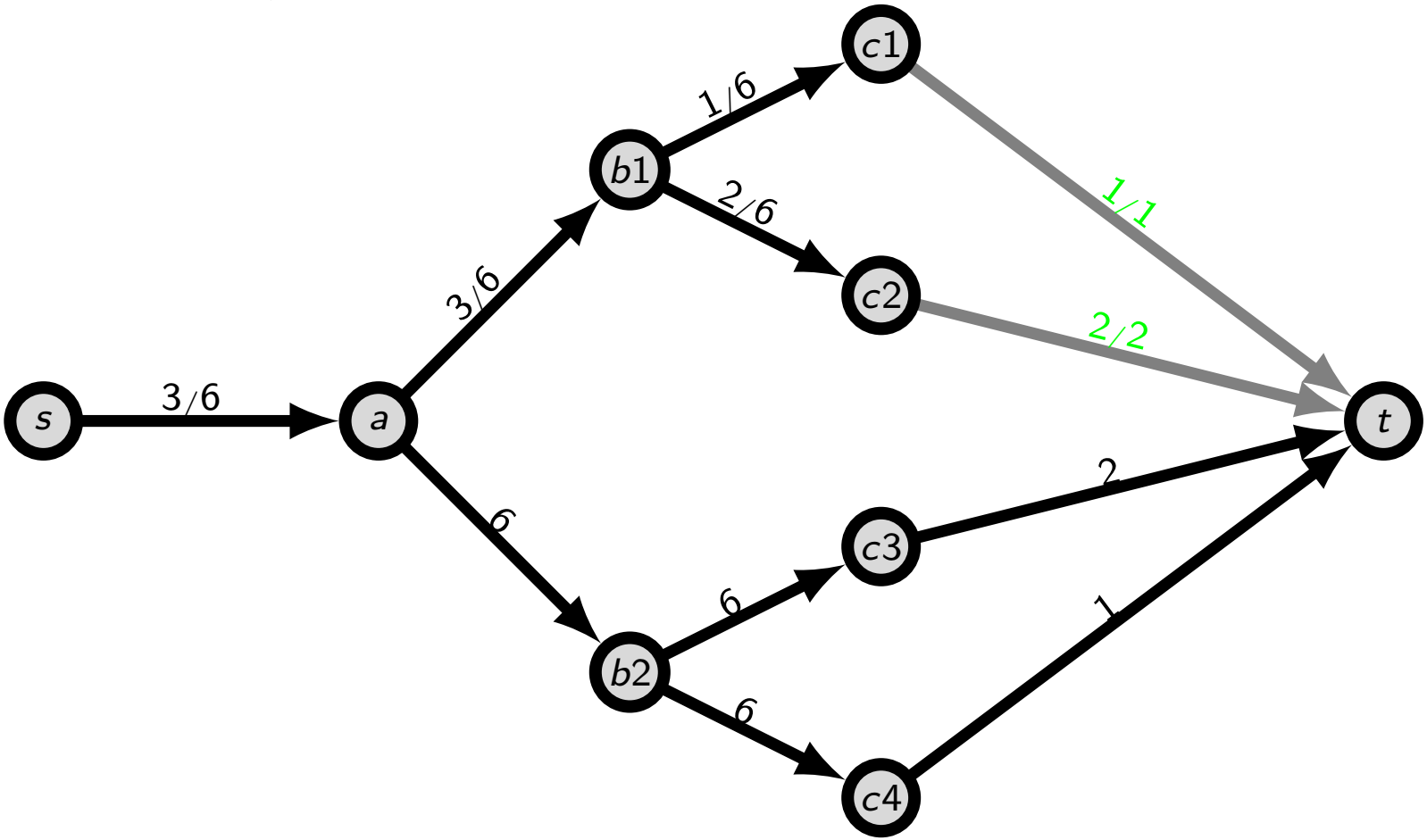
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

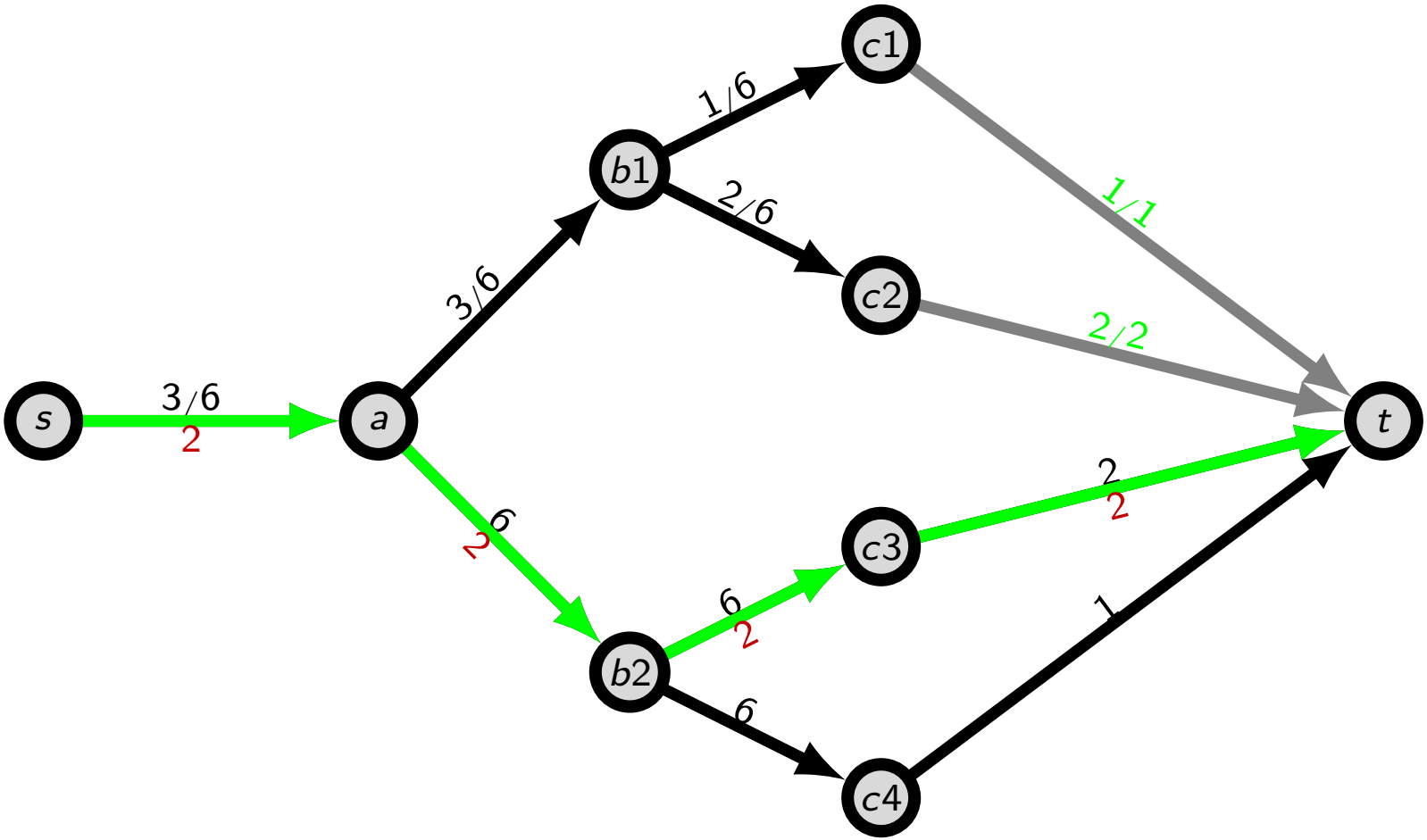
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

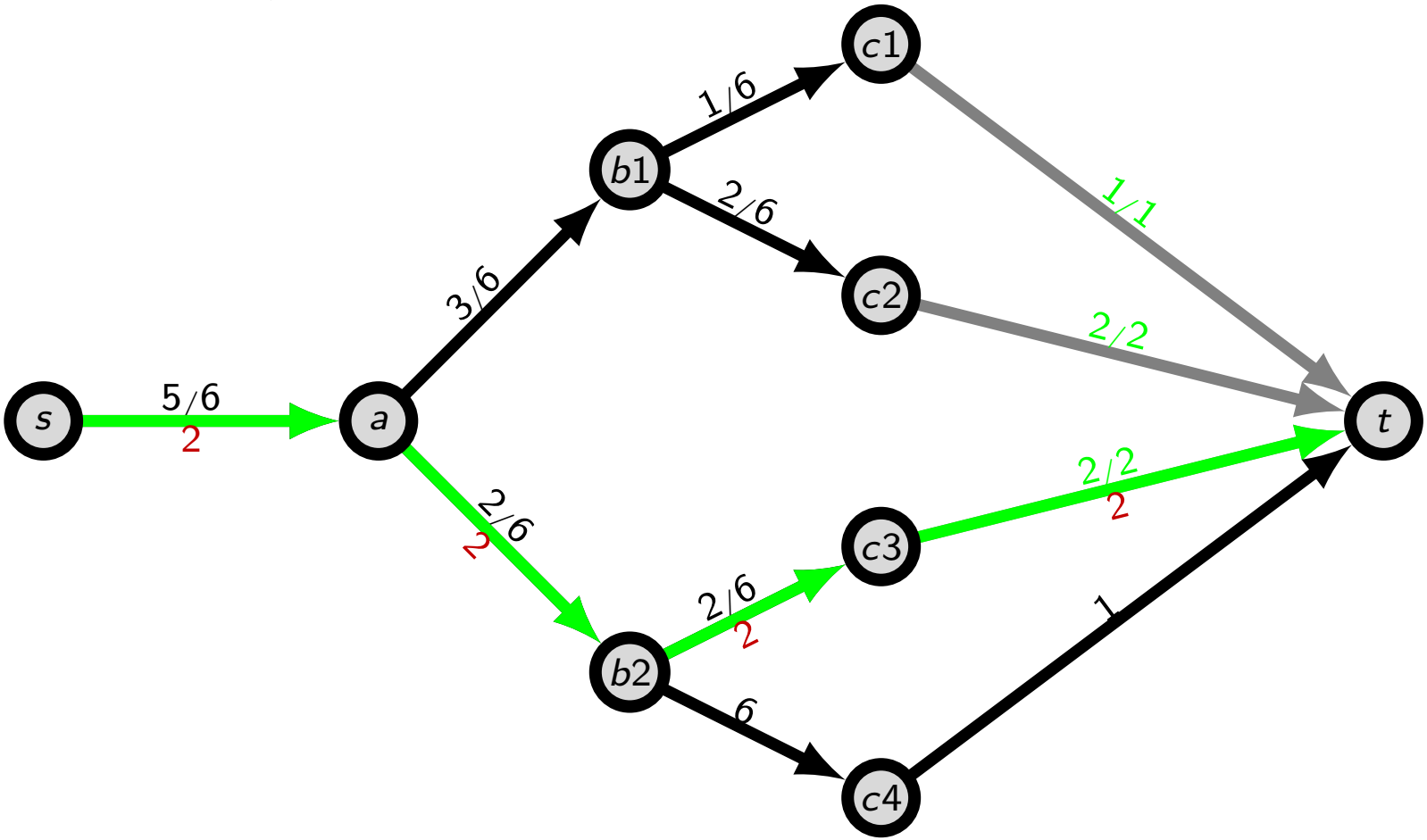
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

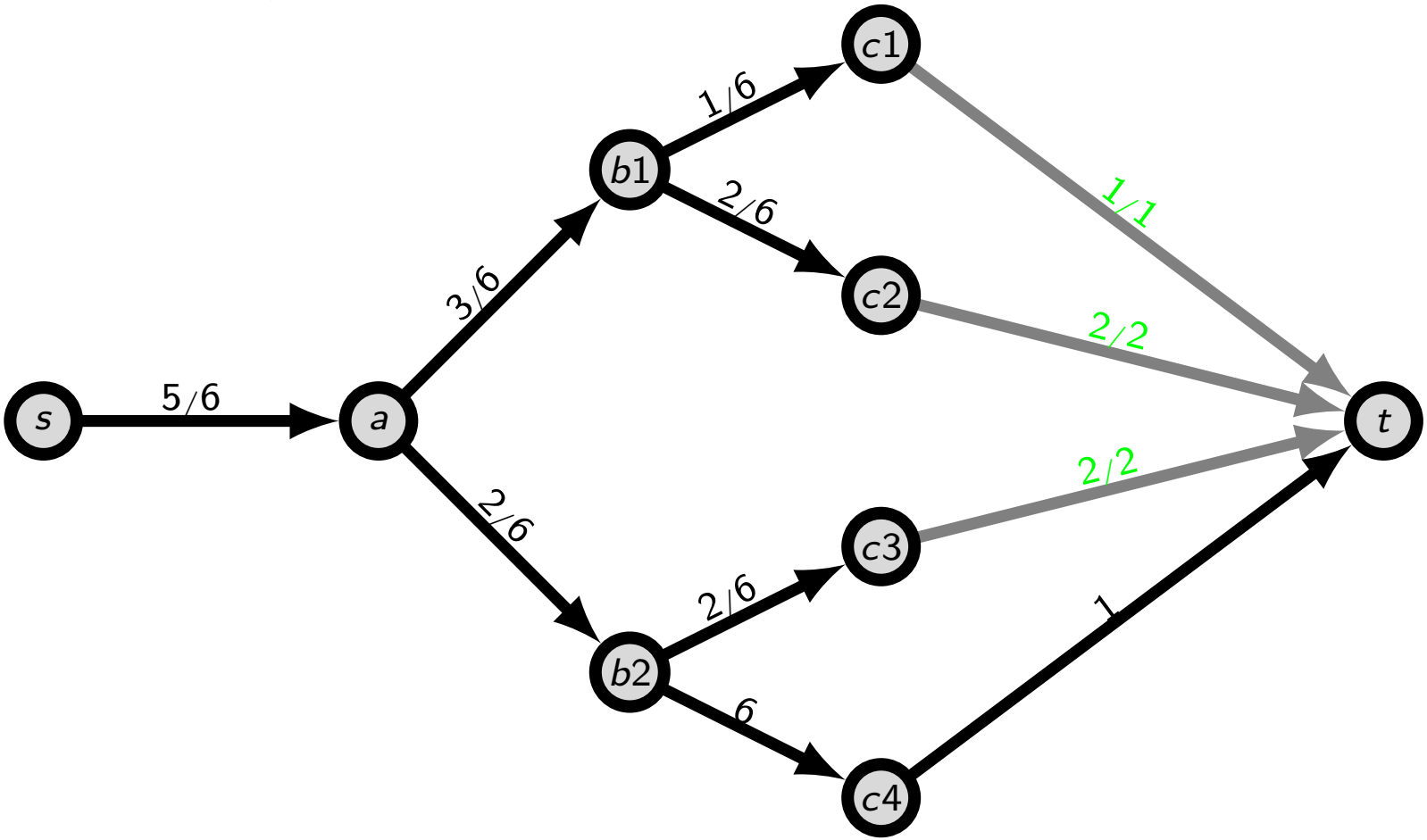
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

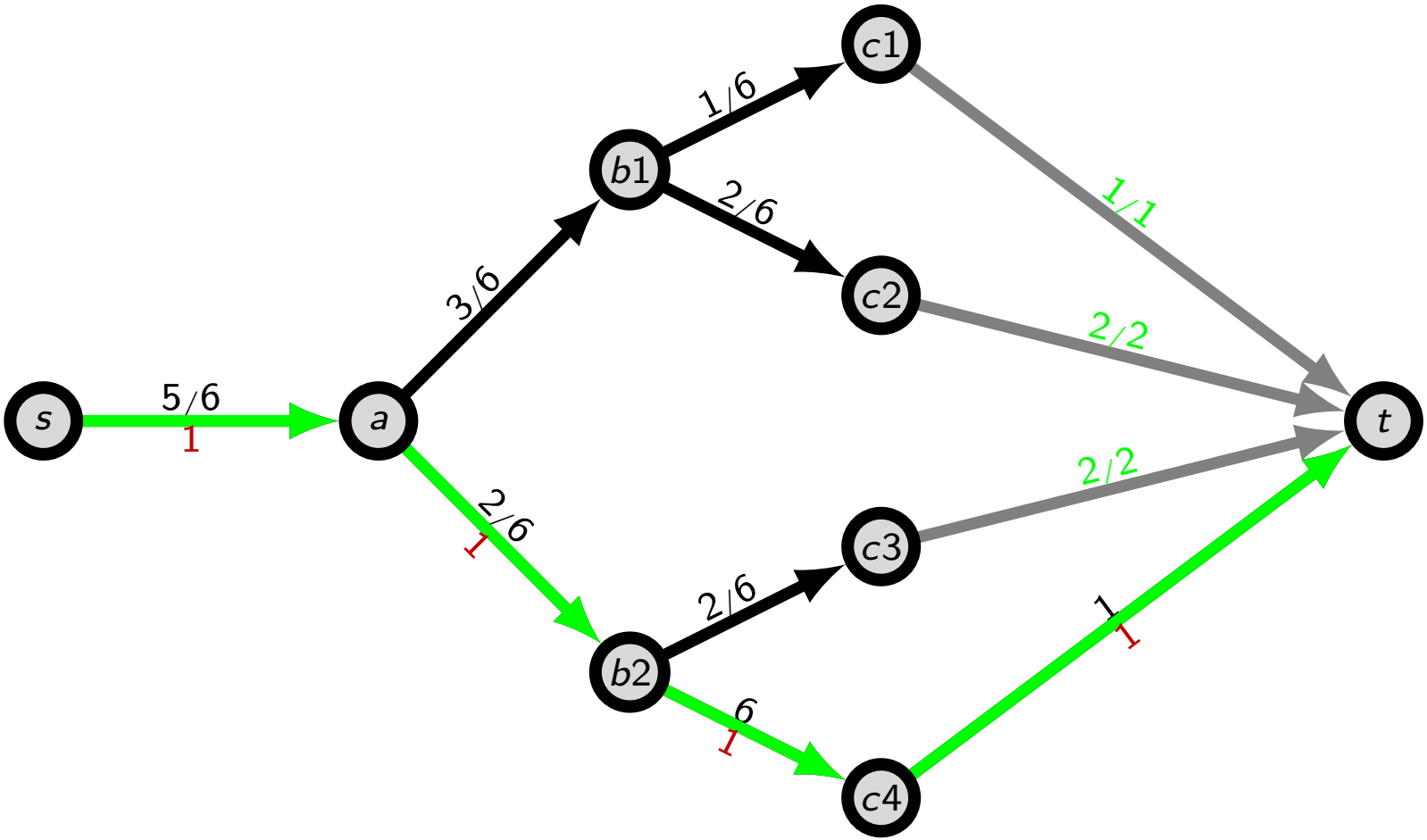
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

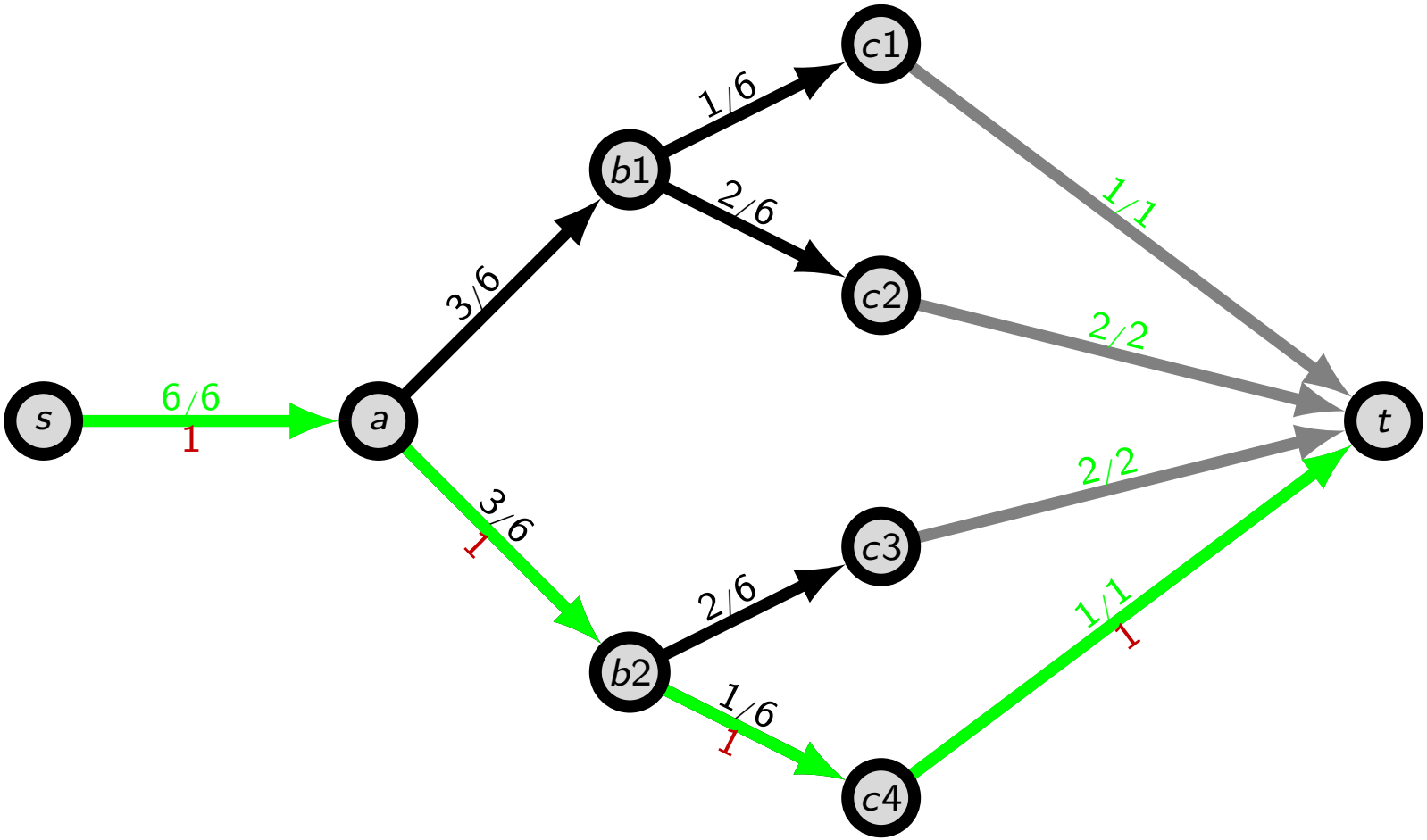
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Zweiter Versuch am einfachen Beispiel

$n = |V|, m = |E|$

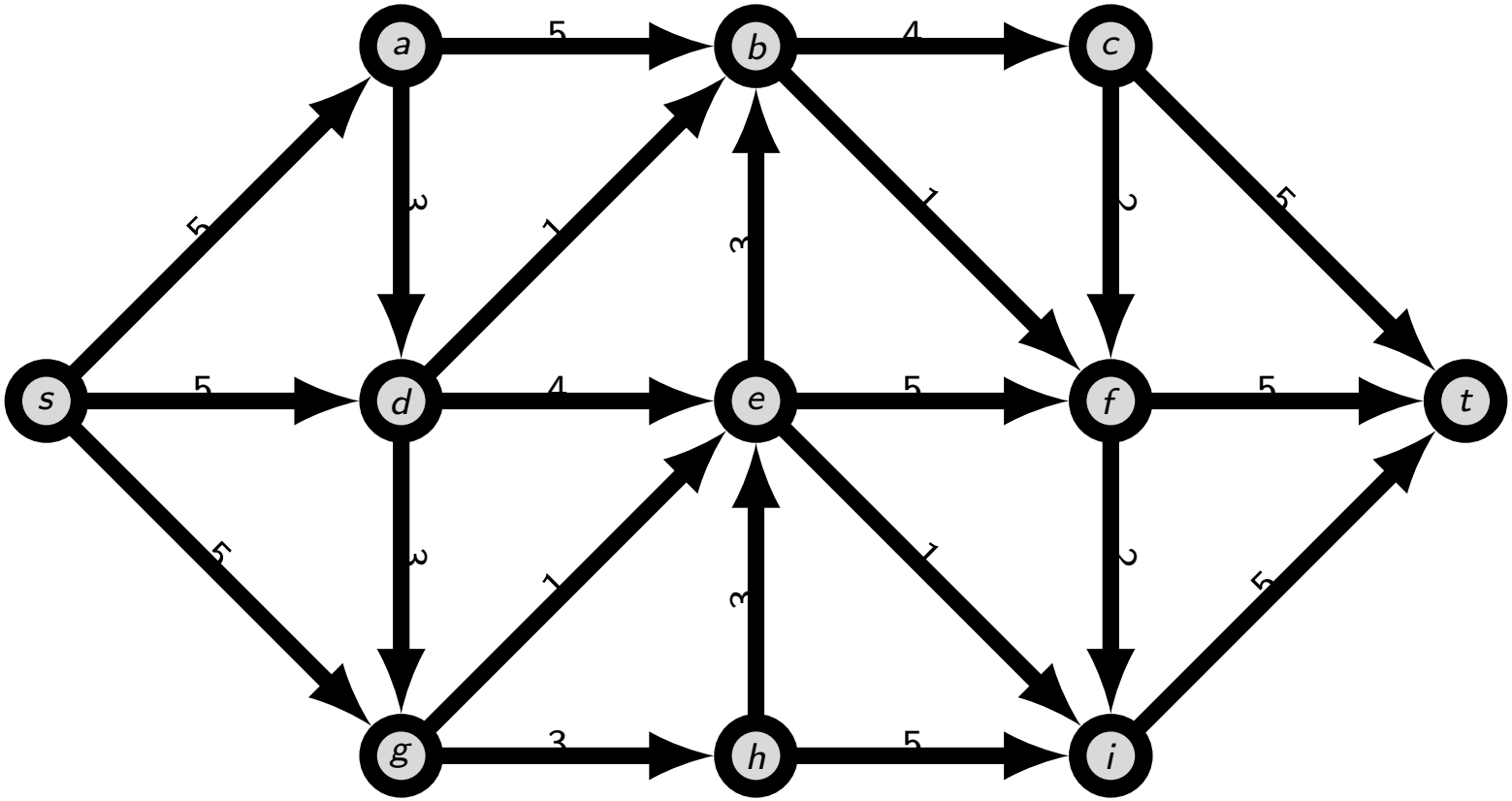
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

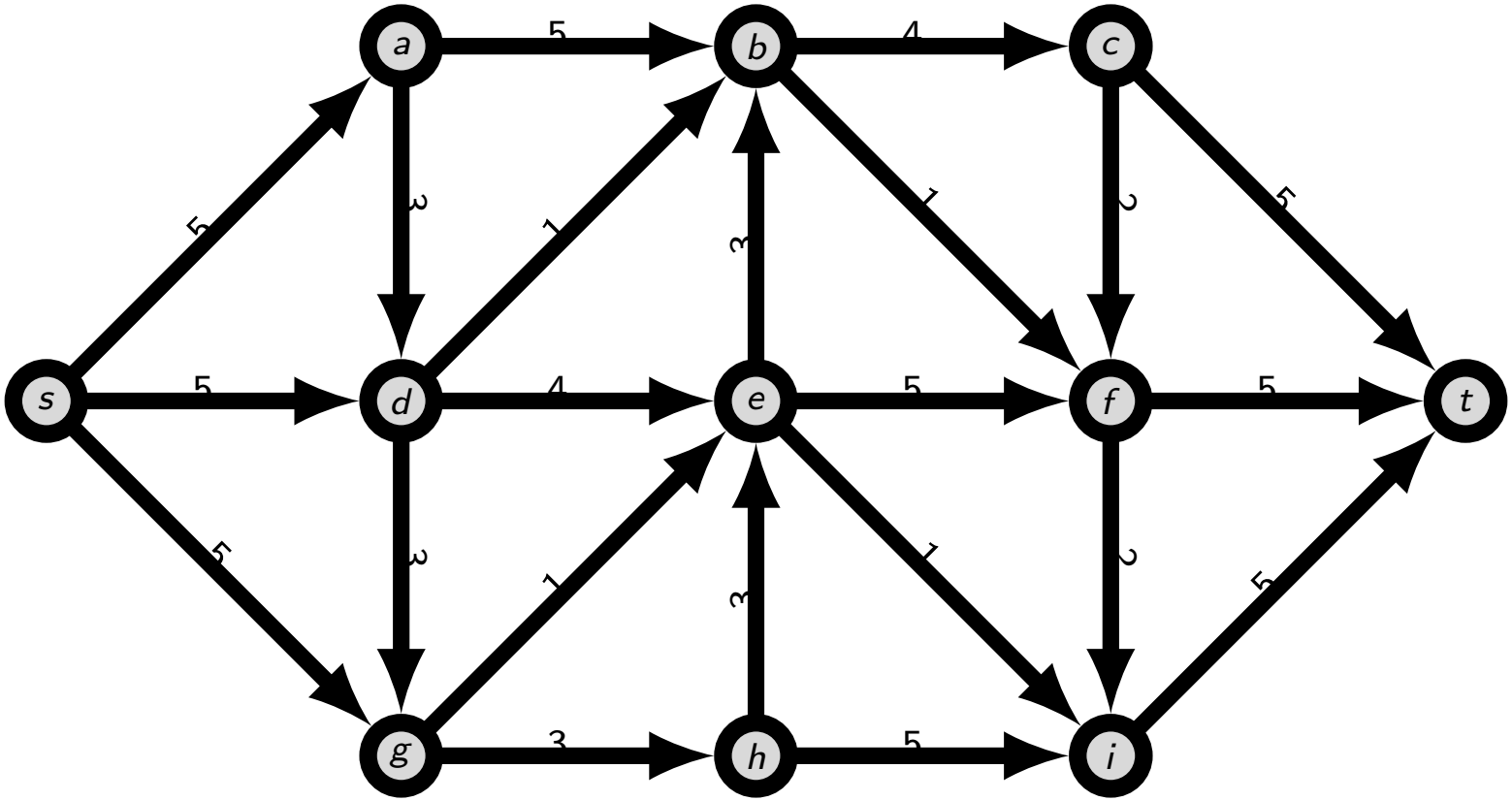
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

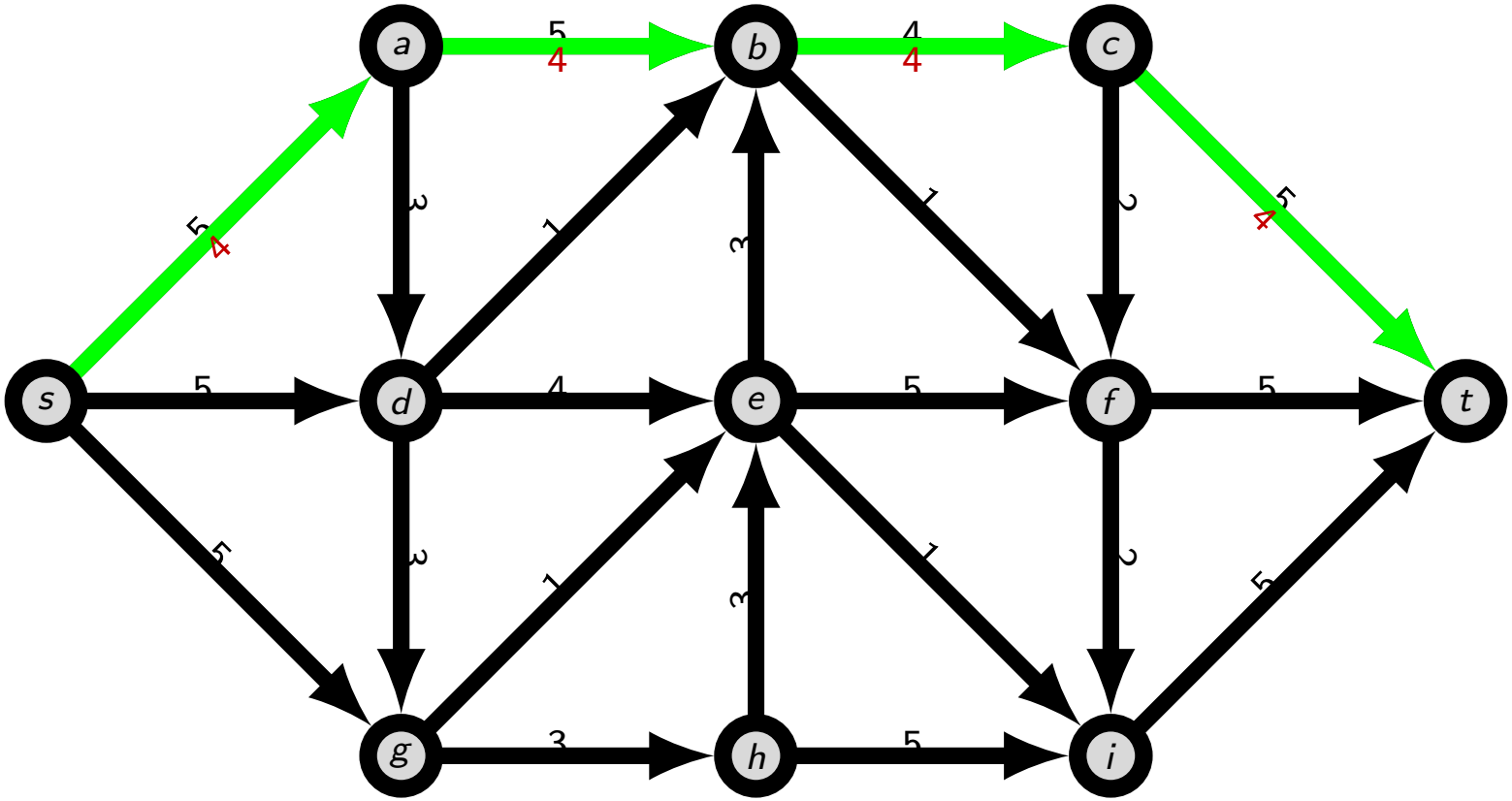
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

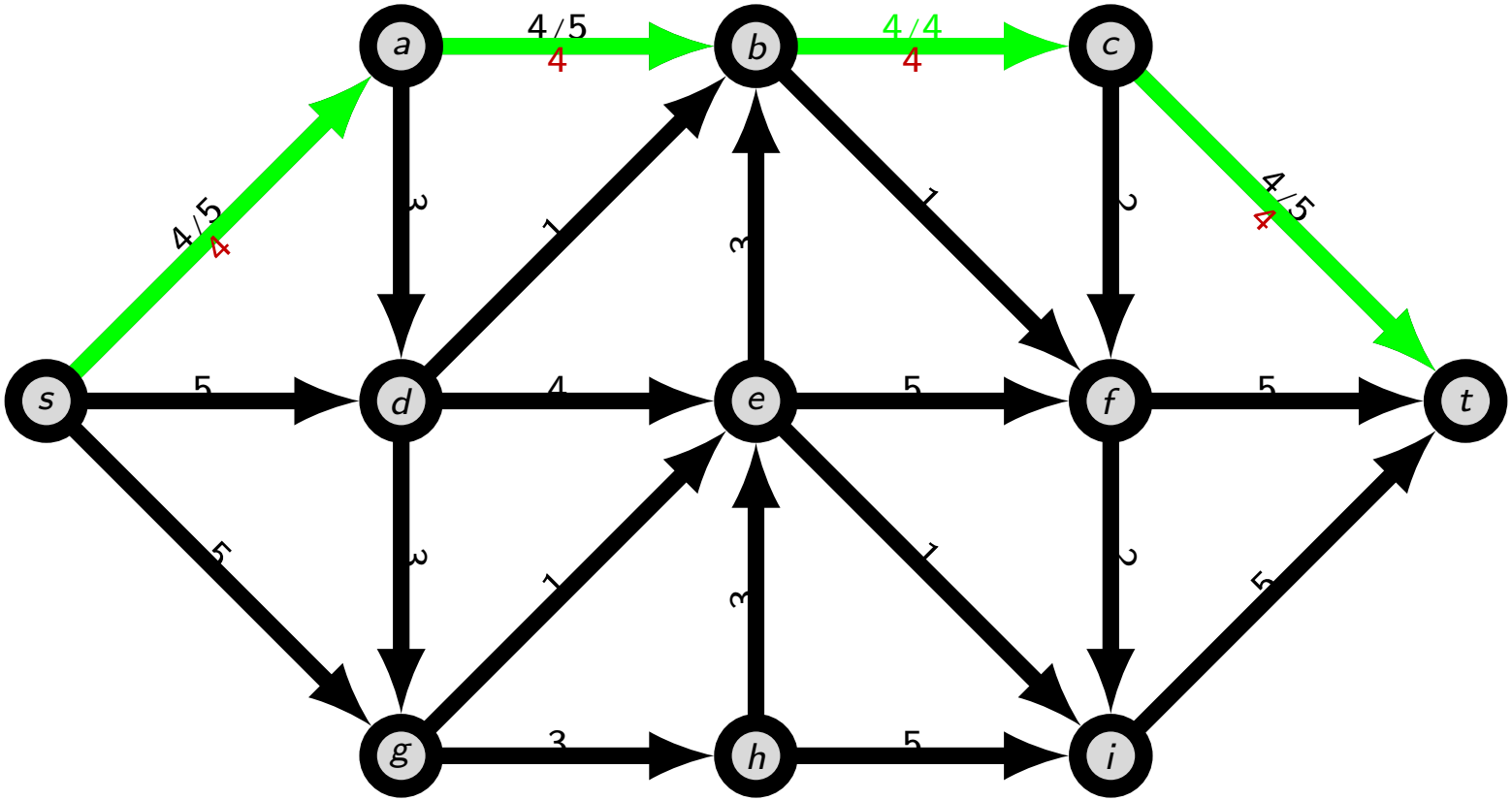
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

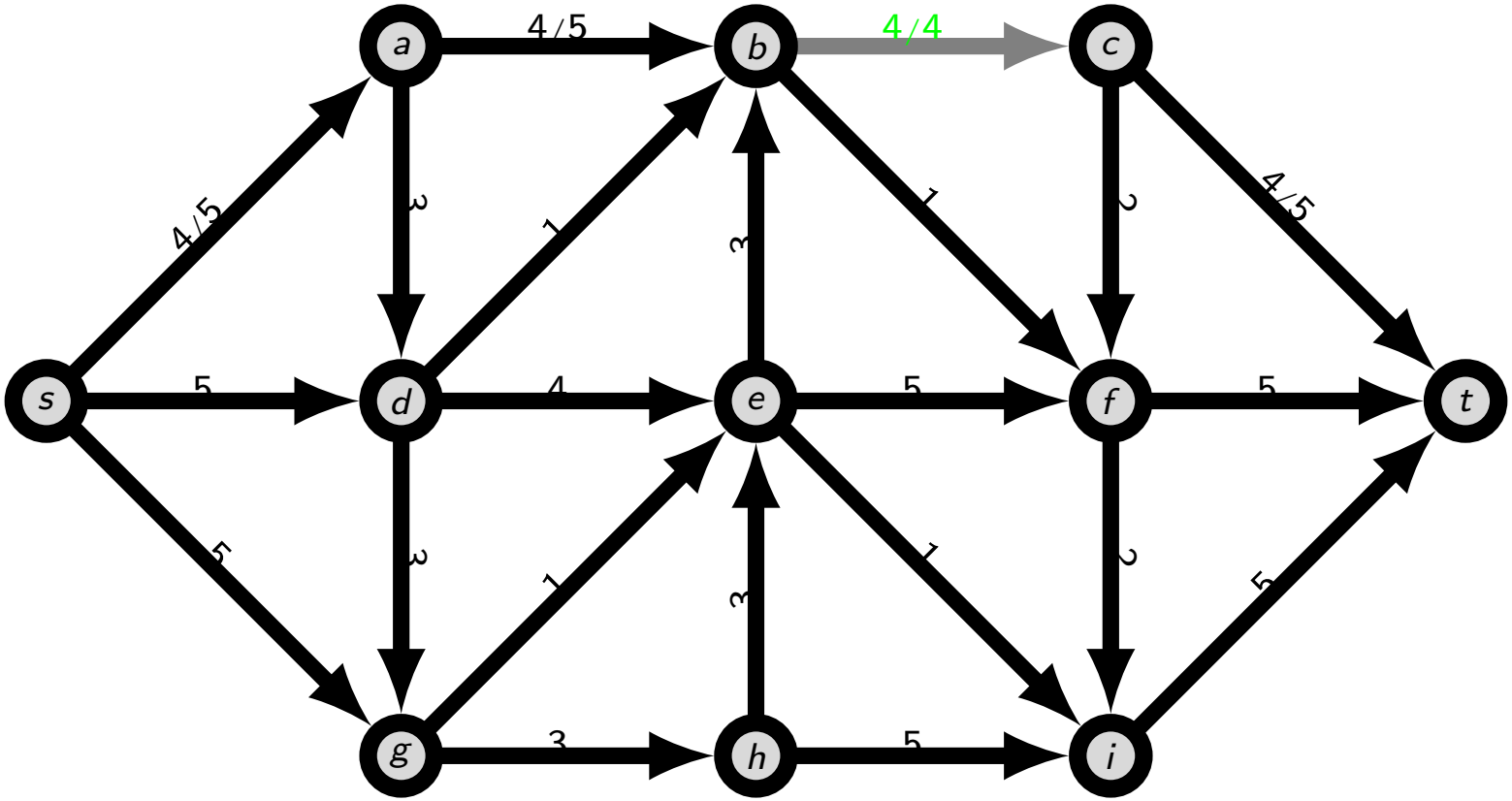
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

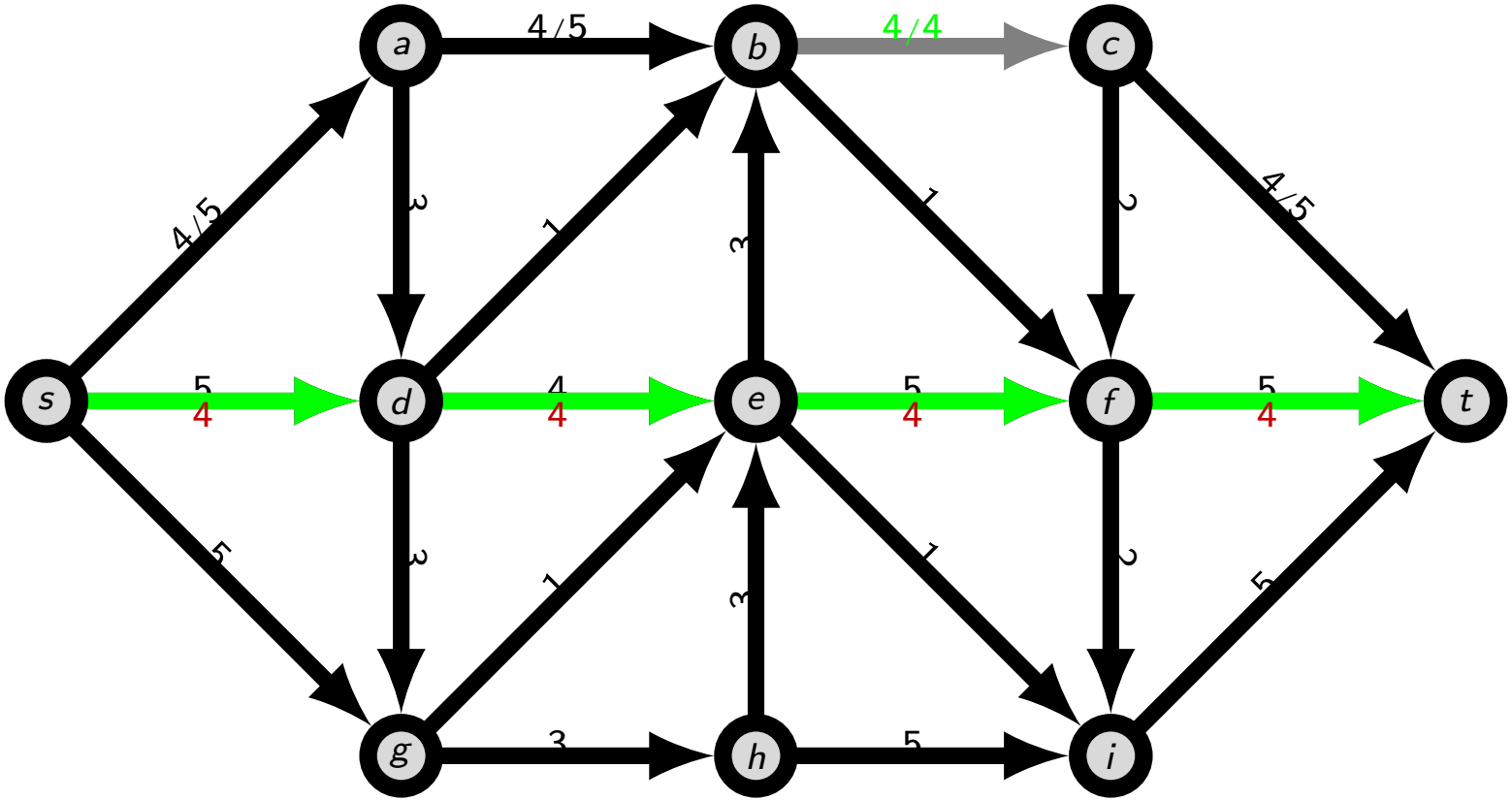
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

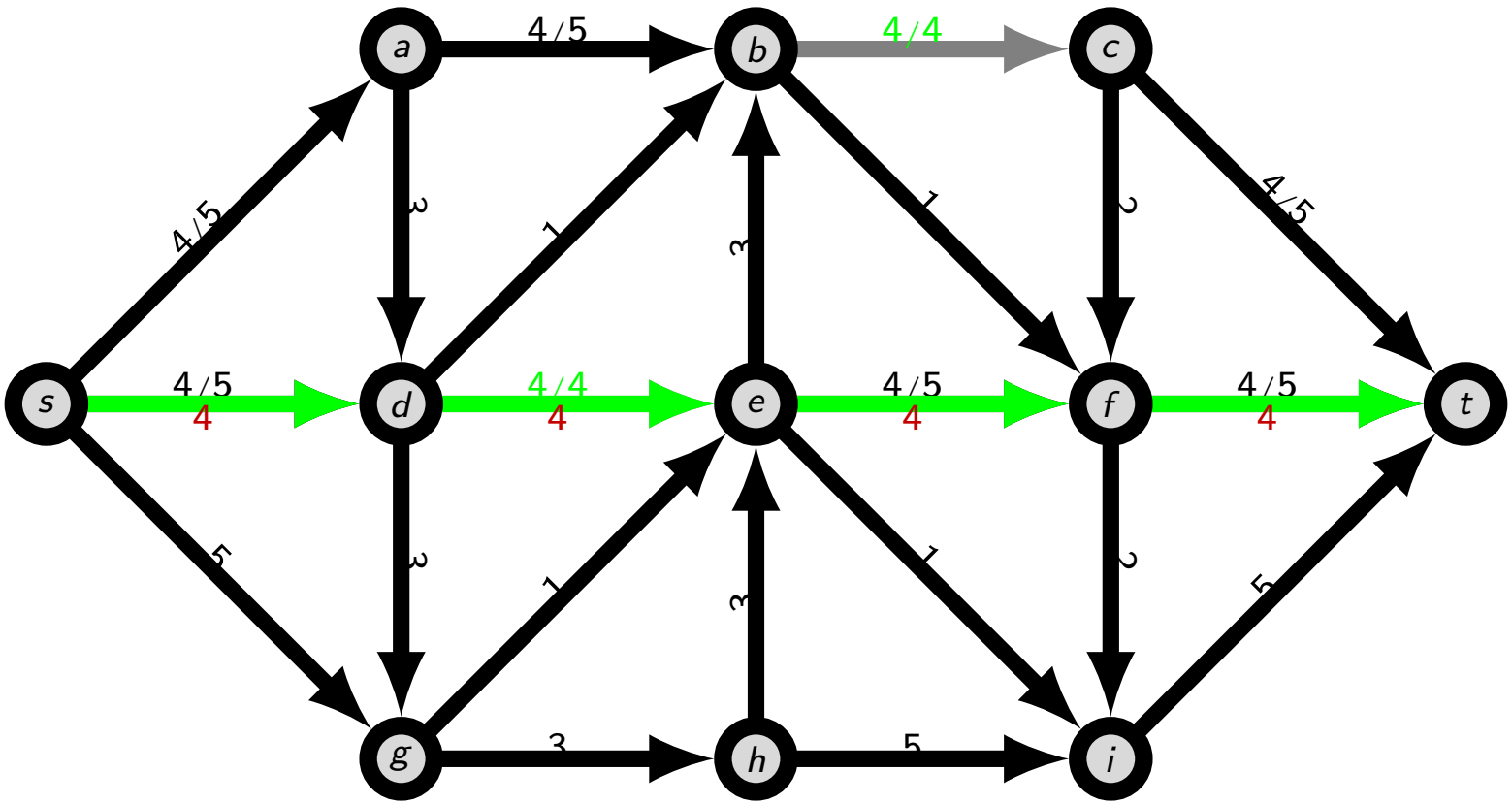
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

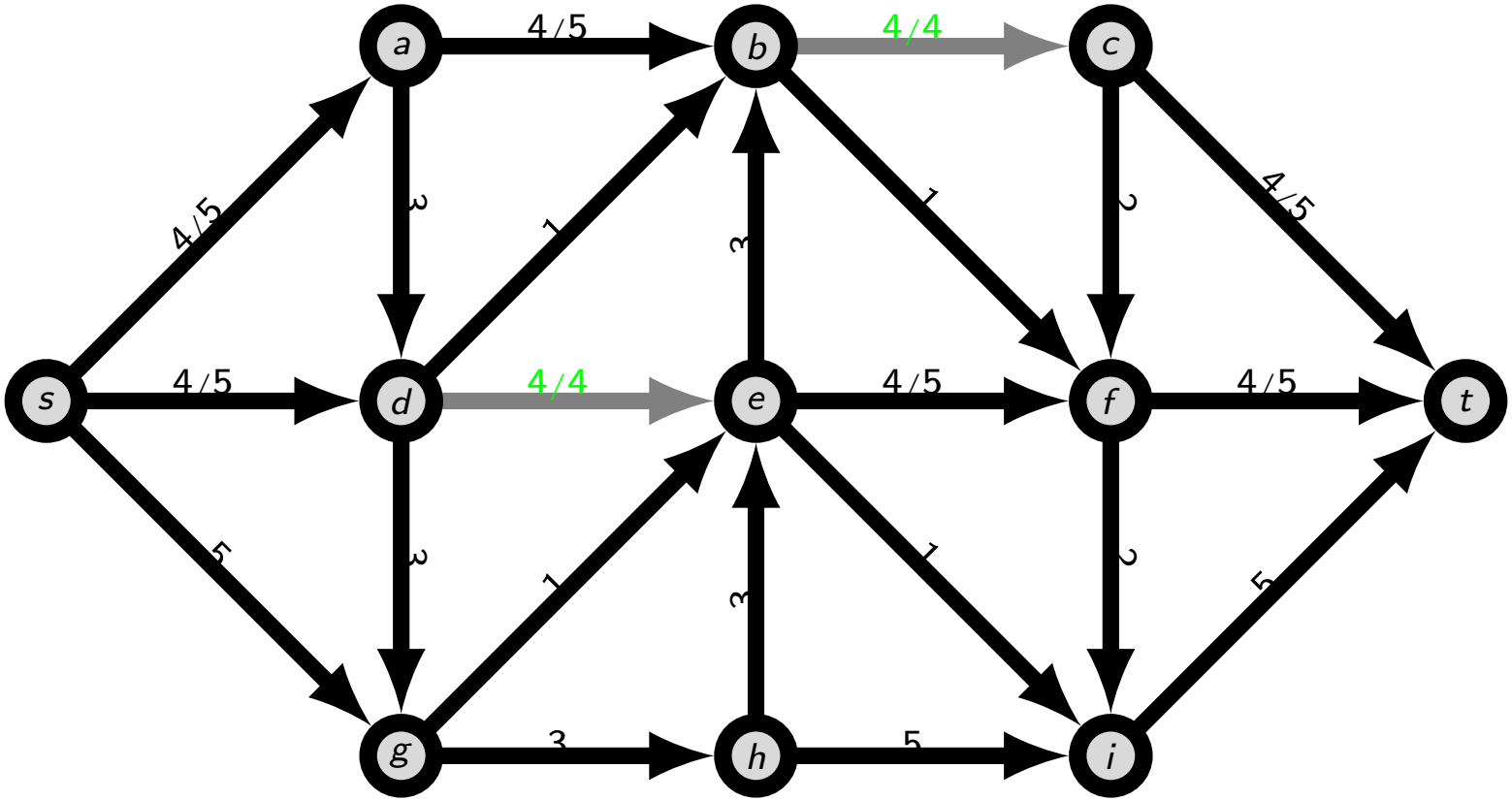
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

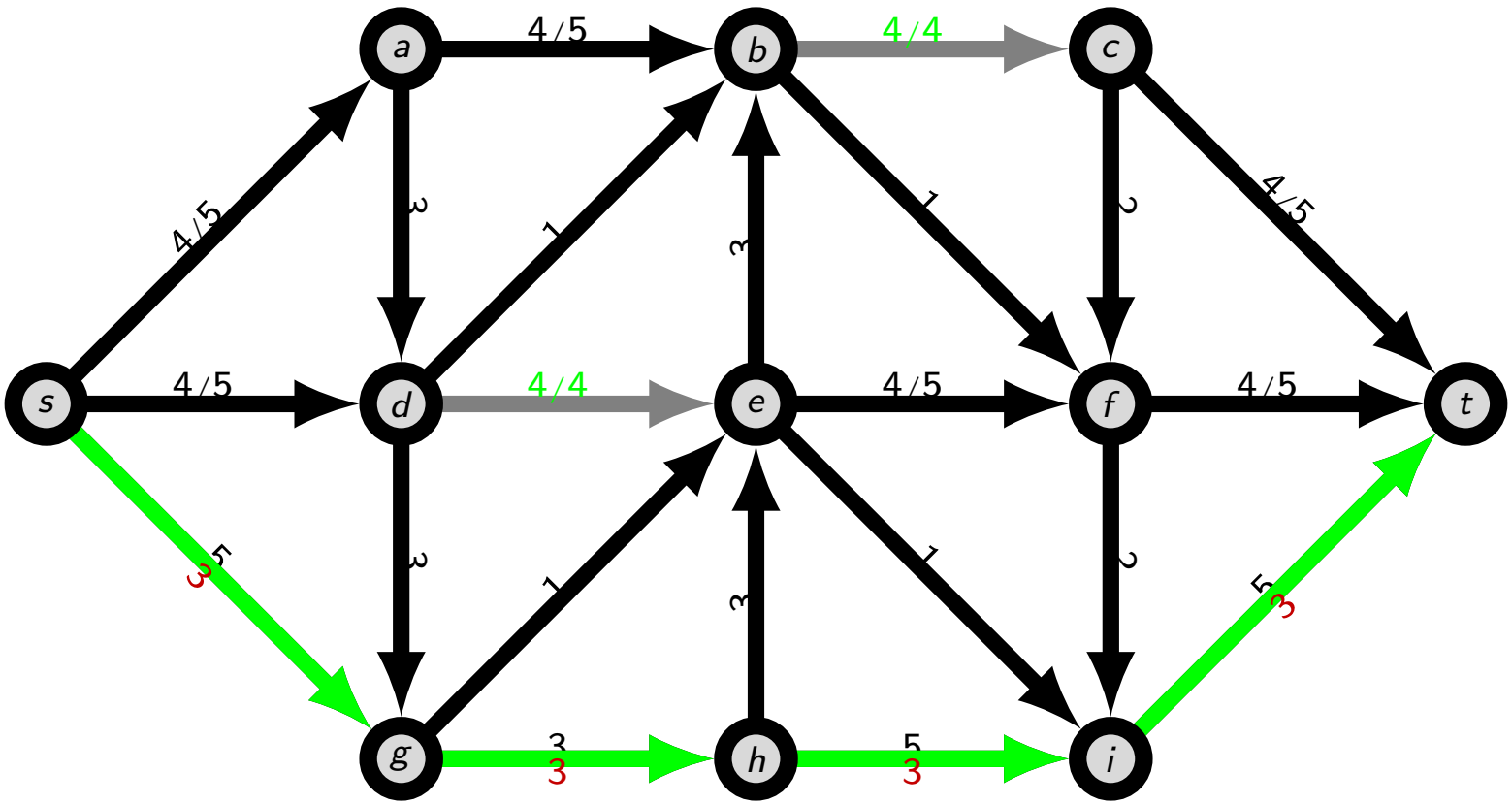
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

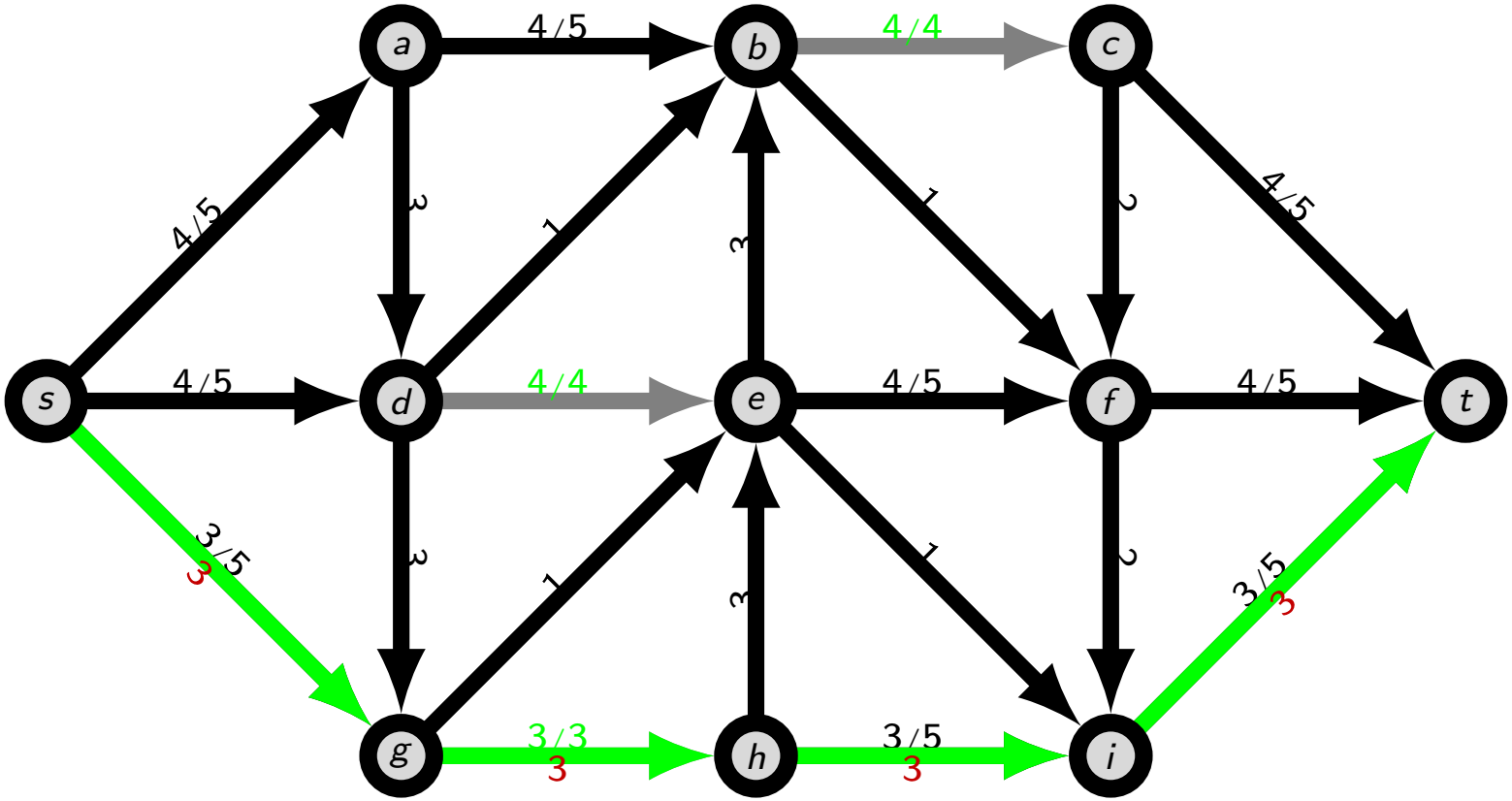
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

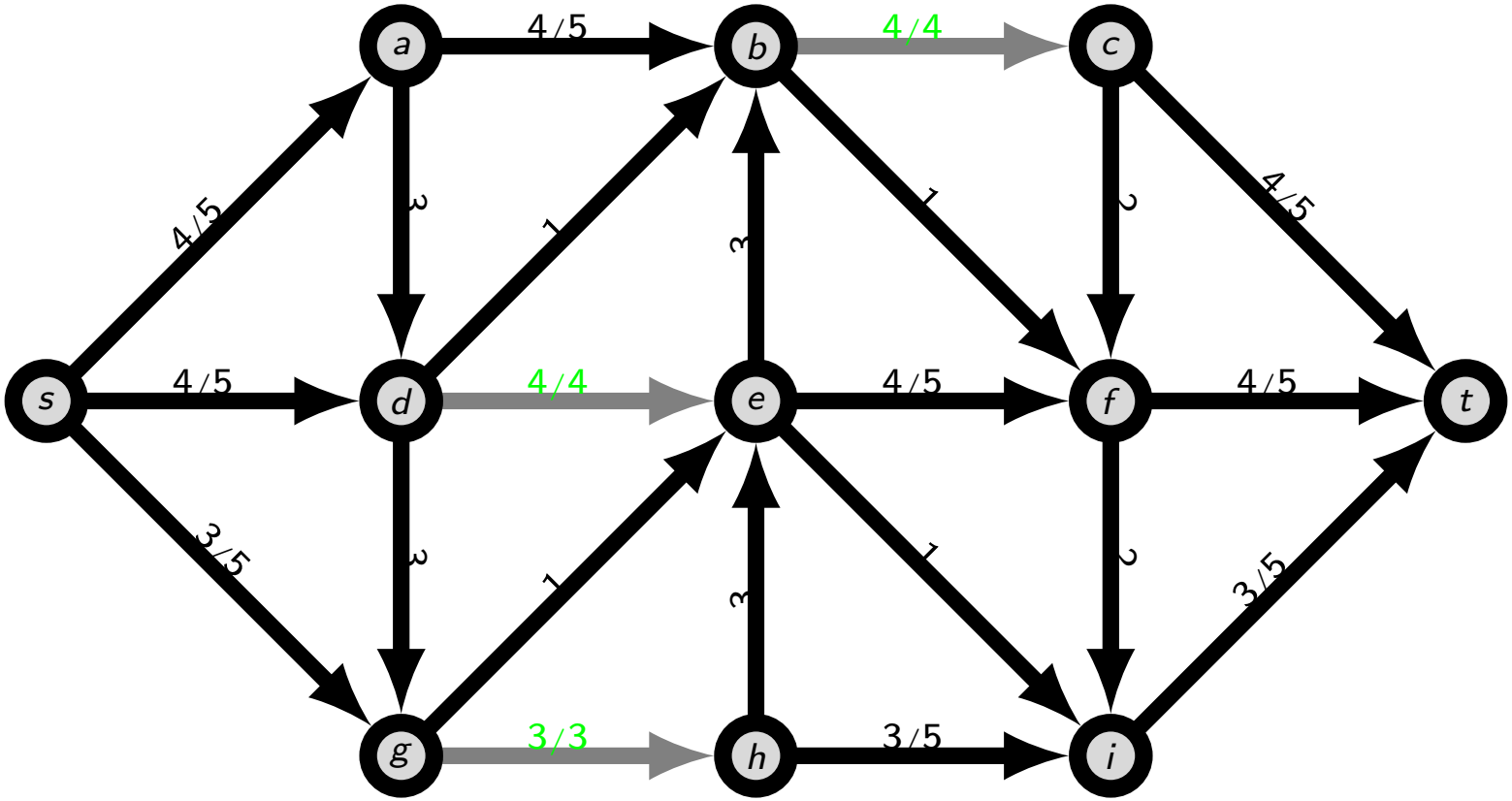
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.

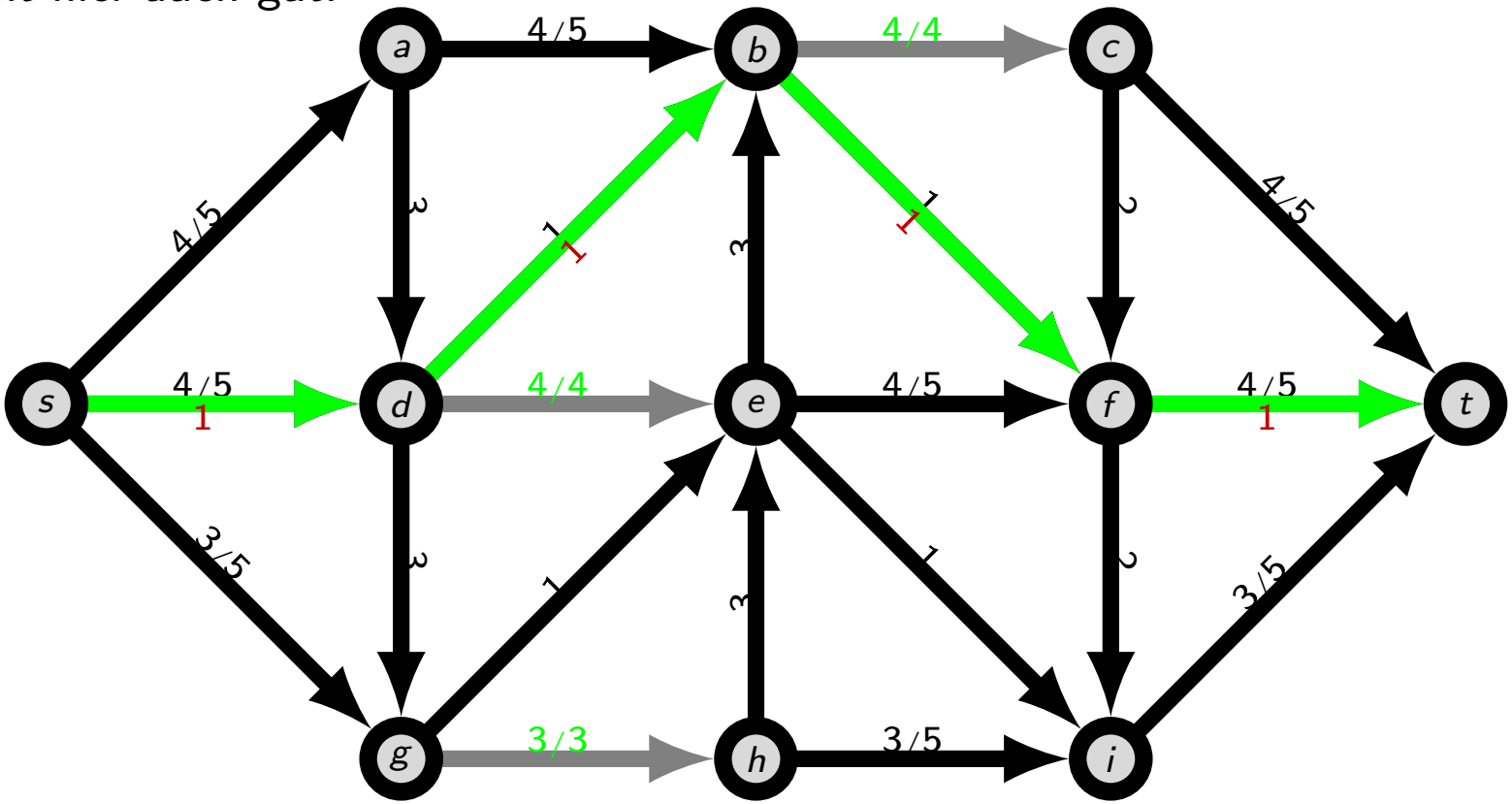


Kleines Beispiel

$n = |V|, m = |E|$

Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.

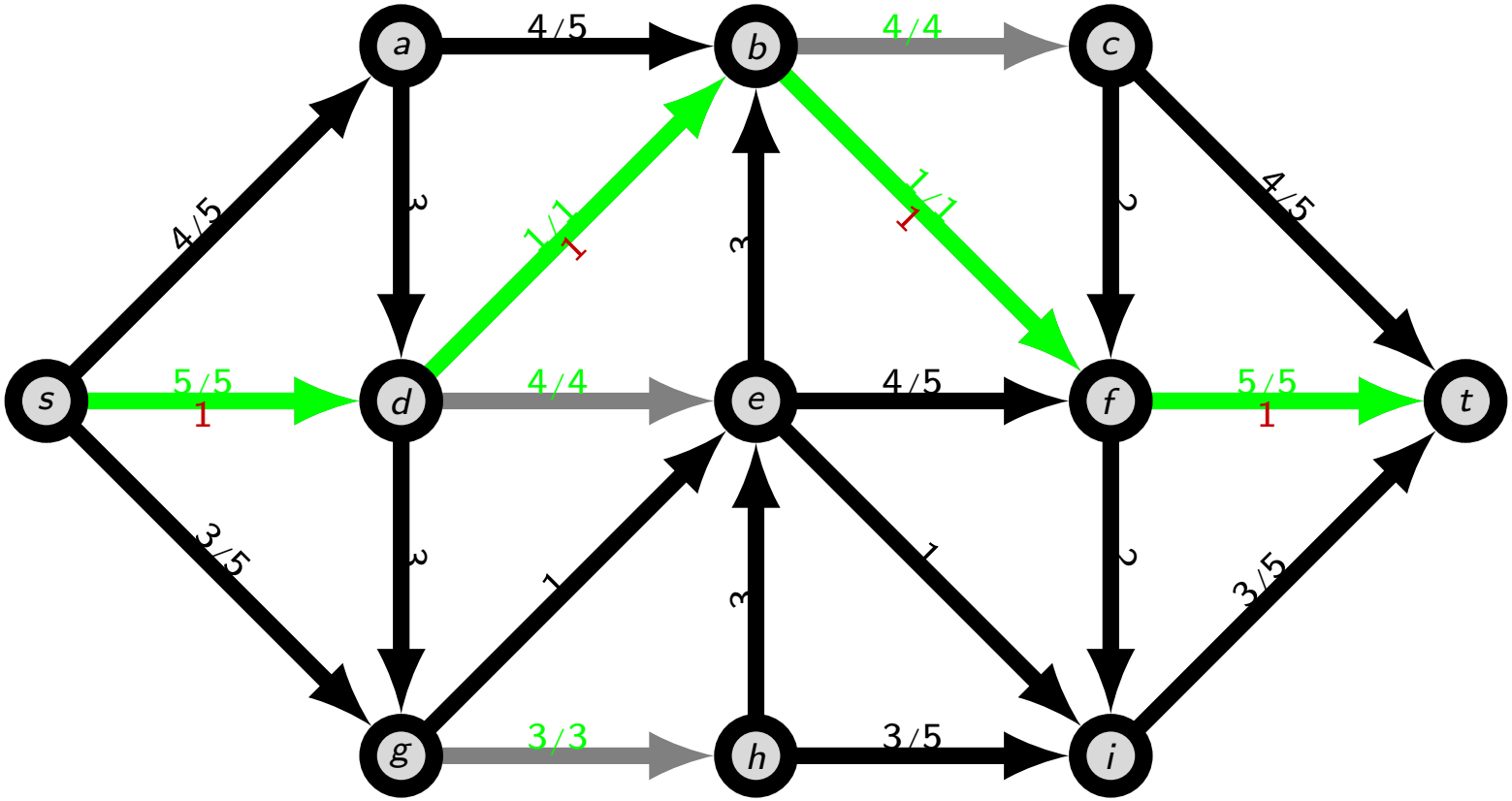
Geht hier auch gut.



Kleines Beispiel

$n = |V|, m = |E|$

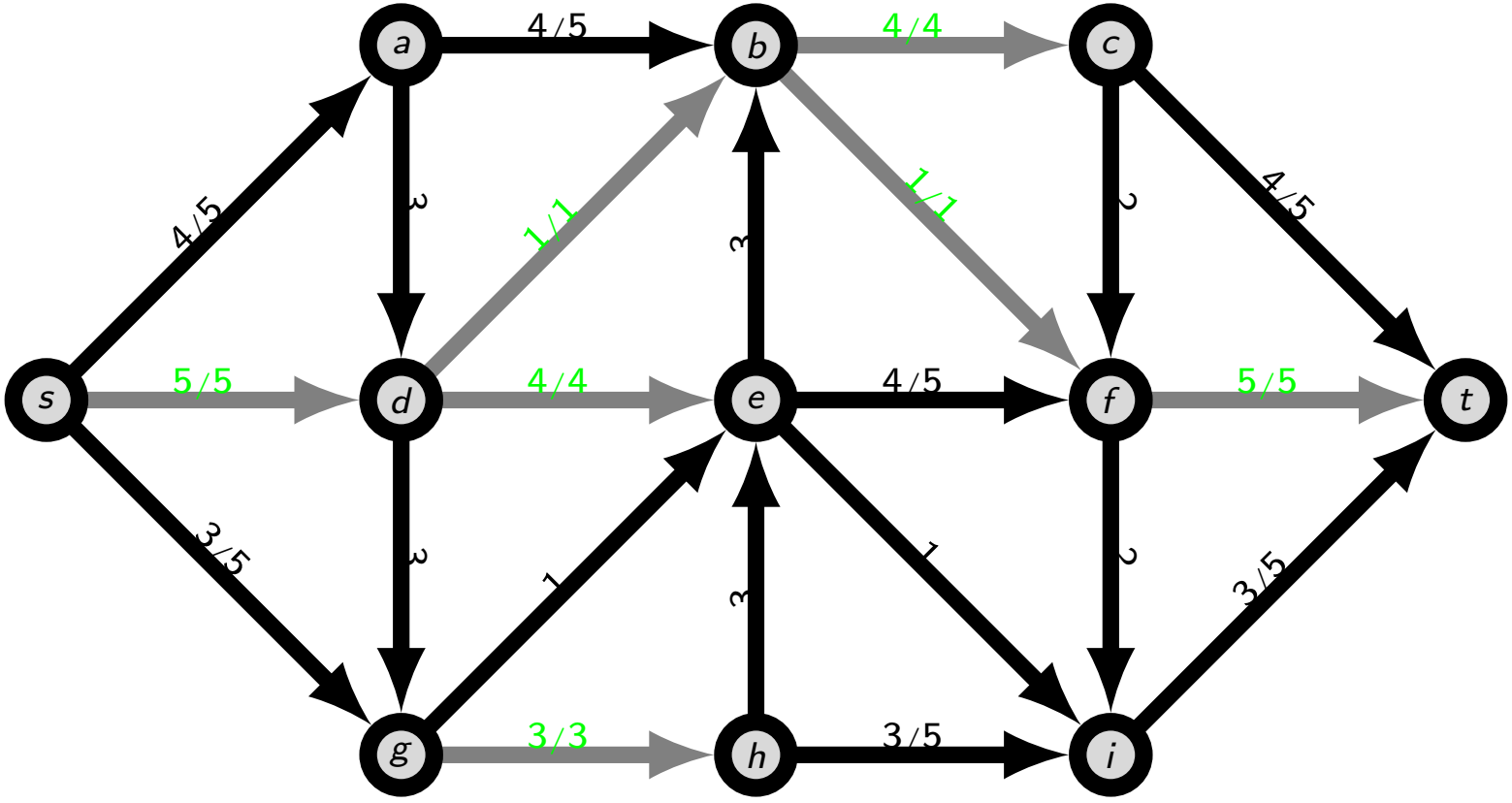
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

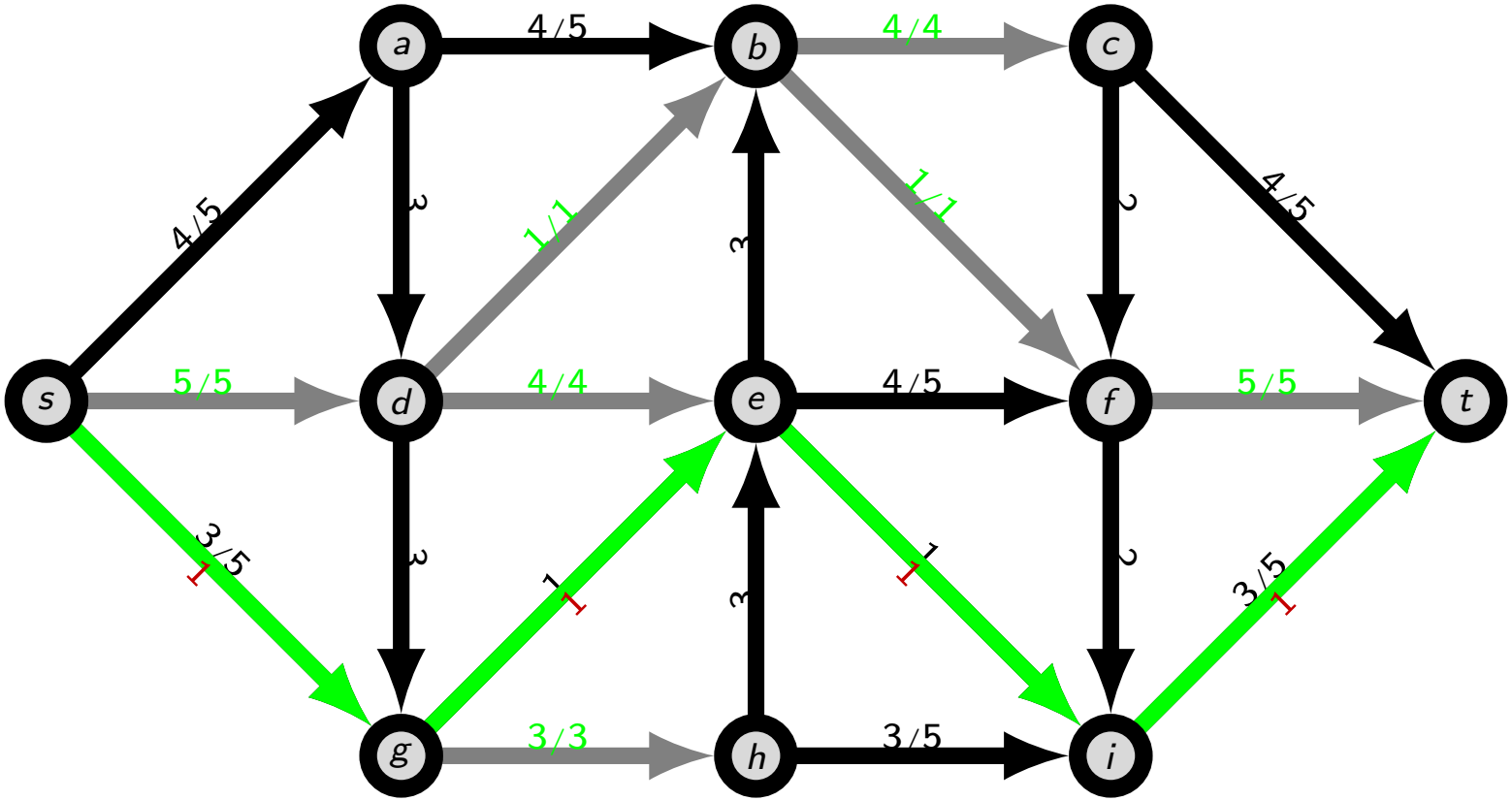
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

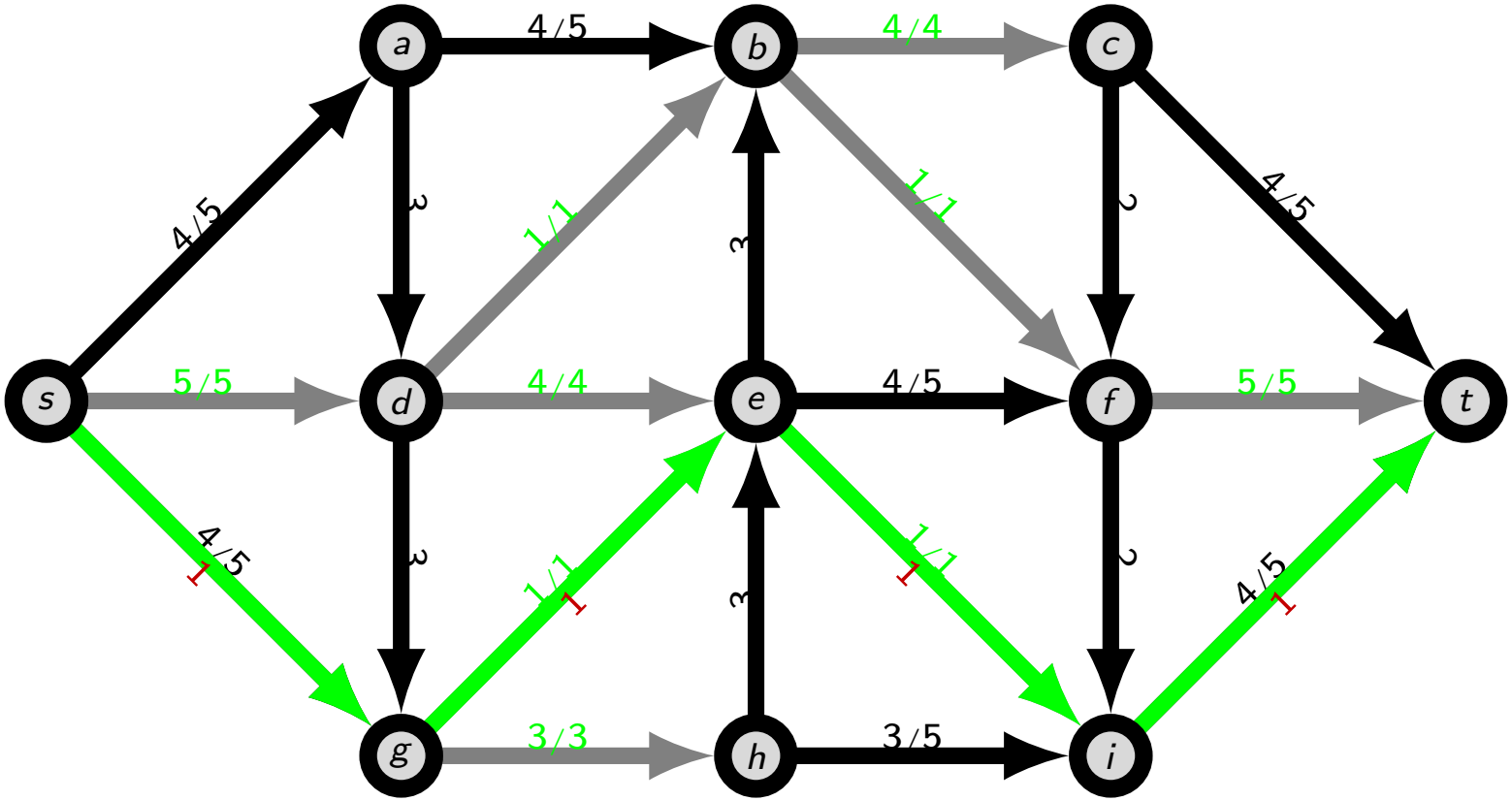
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



Kleines Beispiel

$n = |V|, m = |E|$

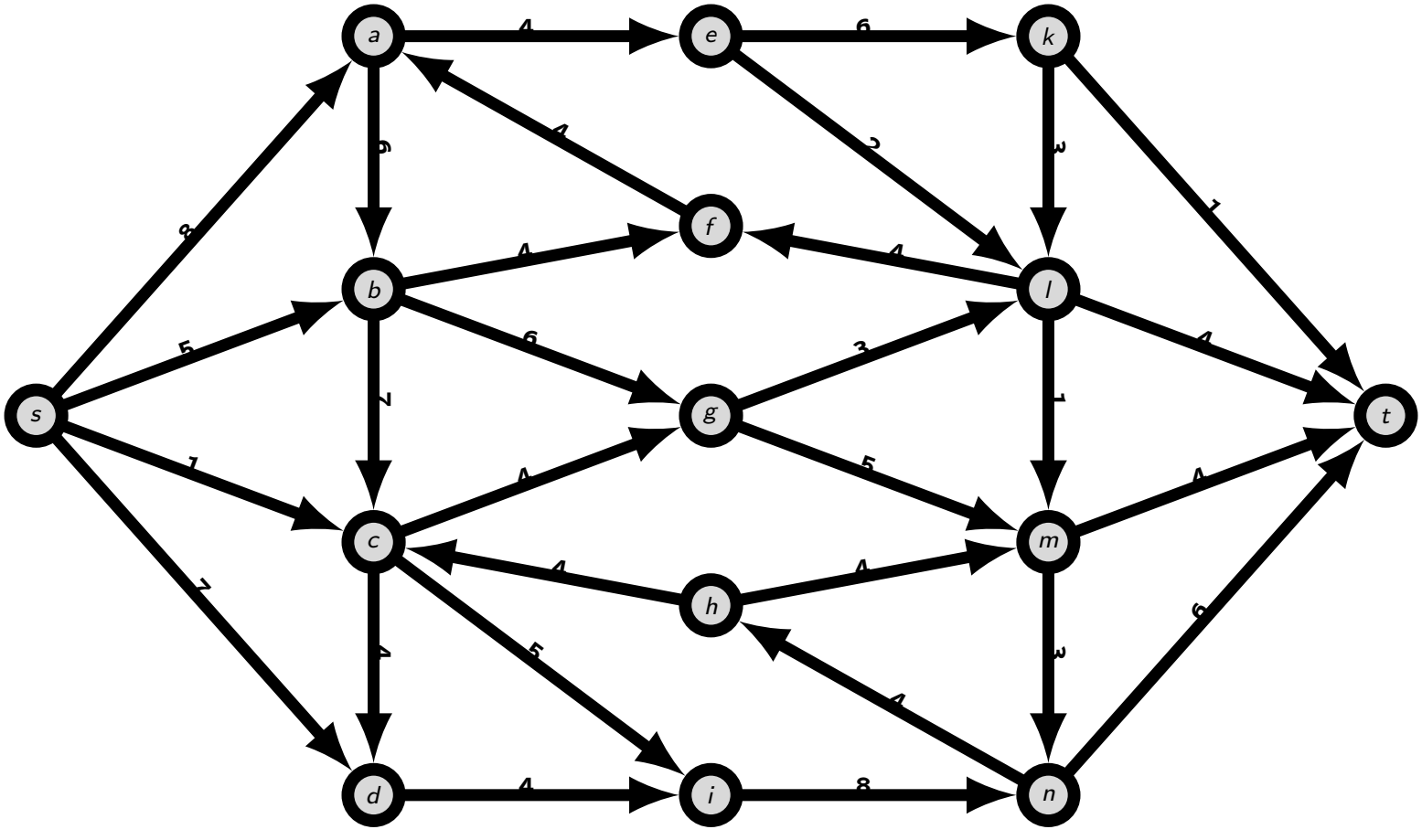
Idee: finde in jedem Schritt einen flussvergrößernden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

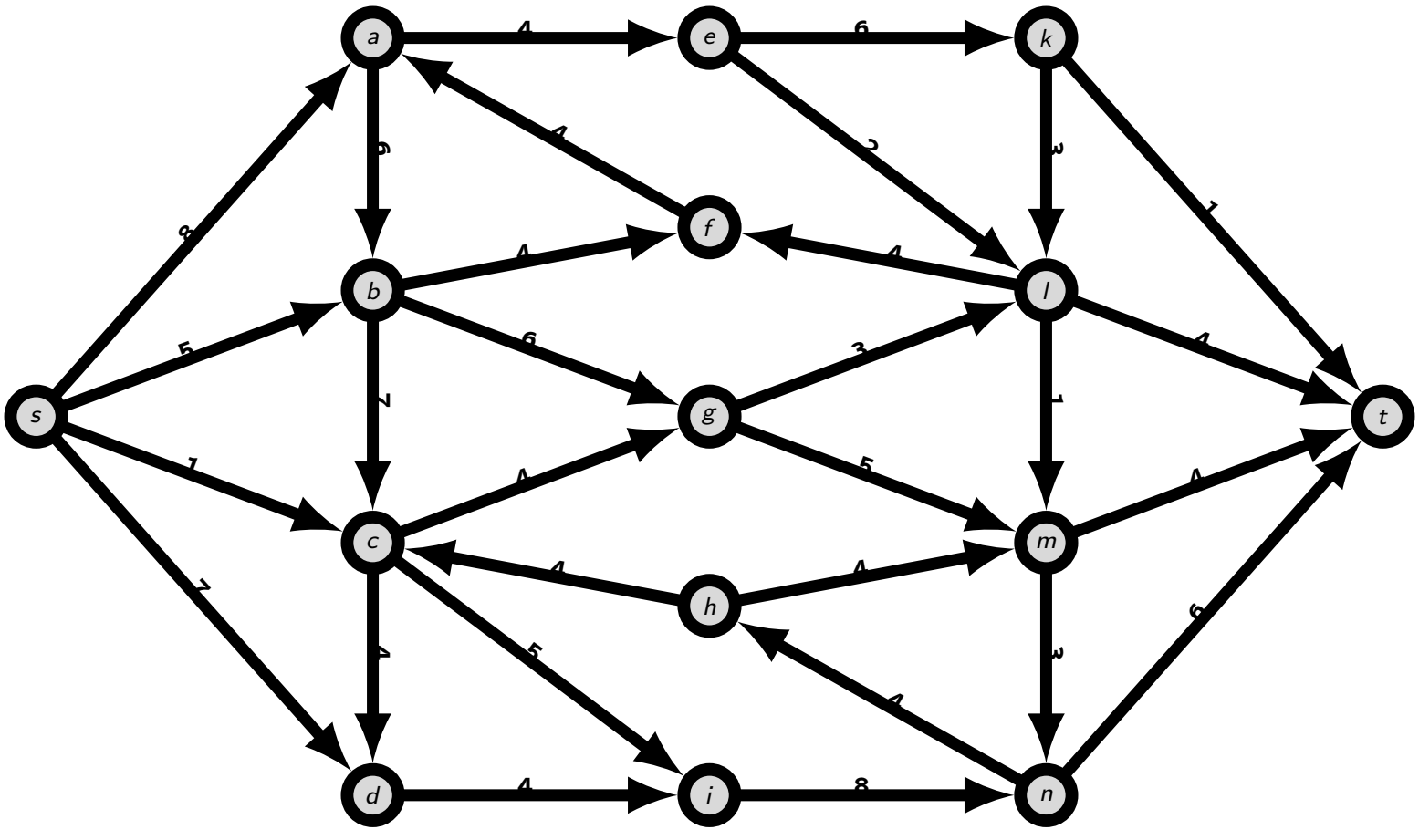
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

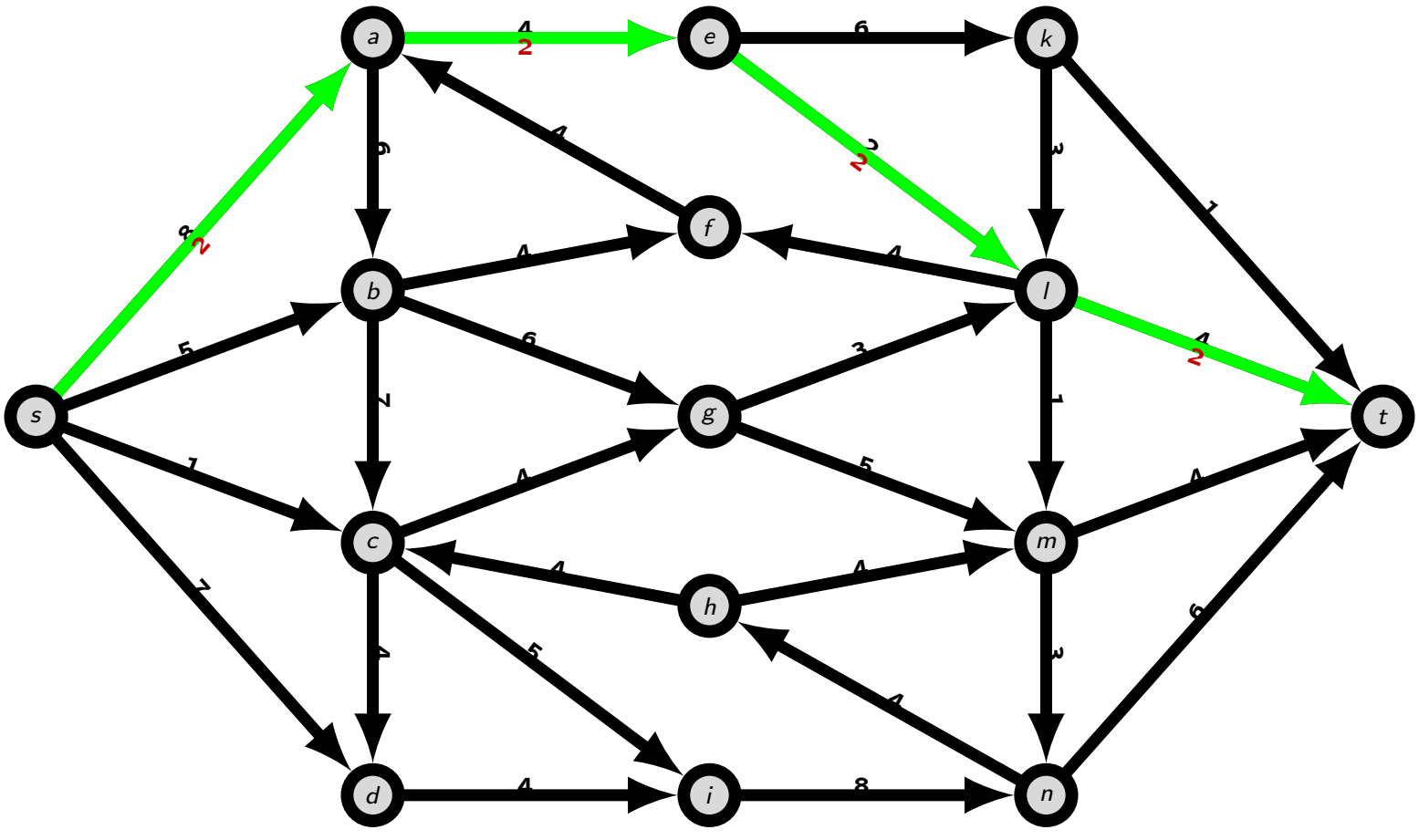
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

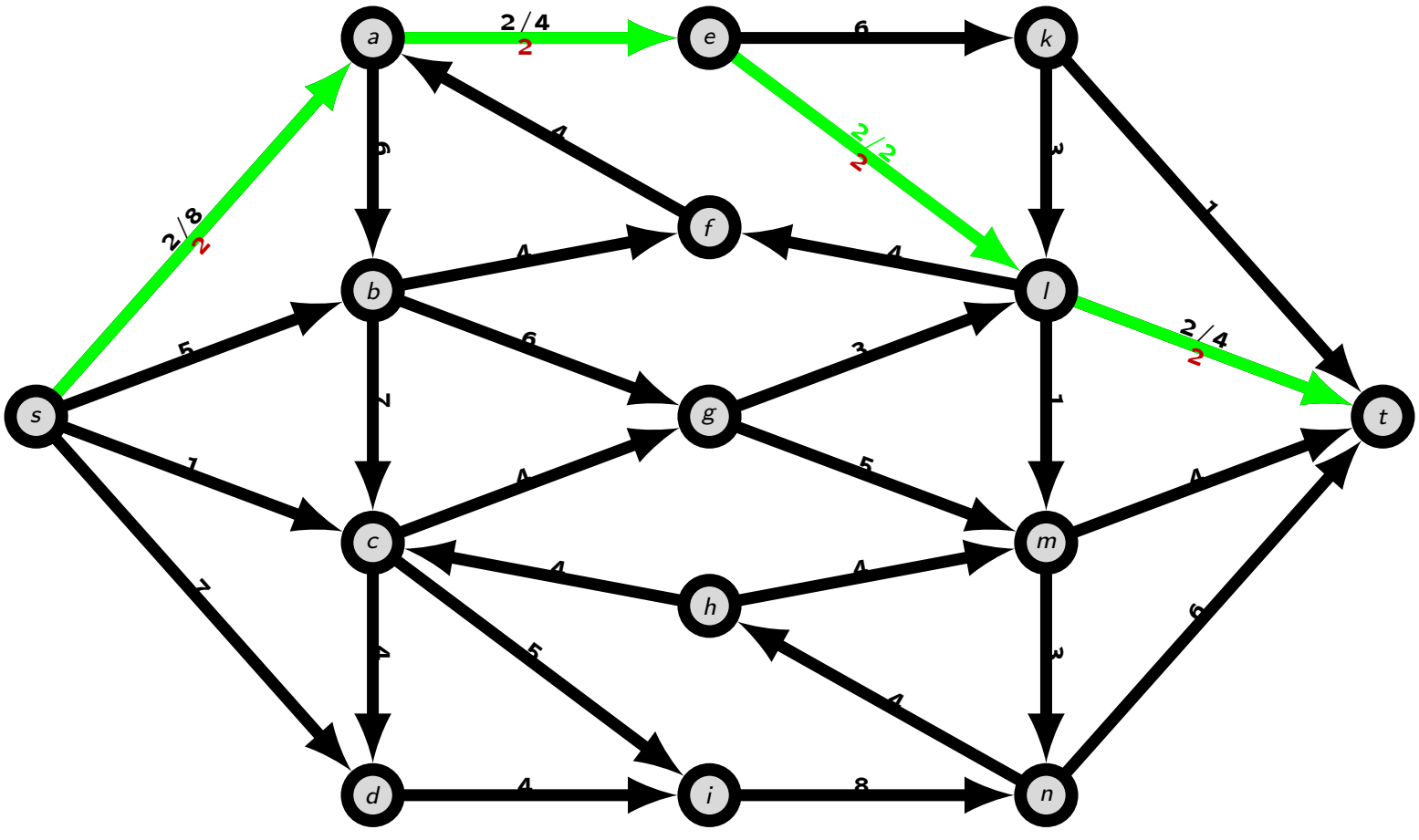
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

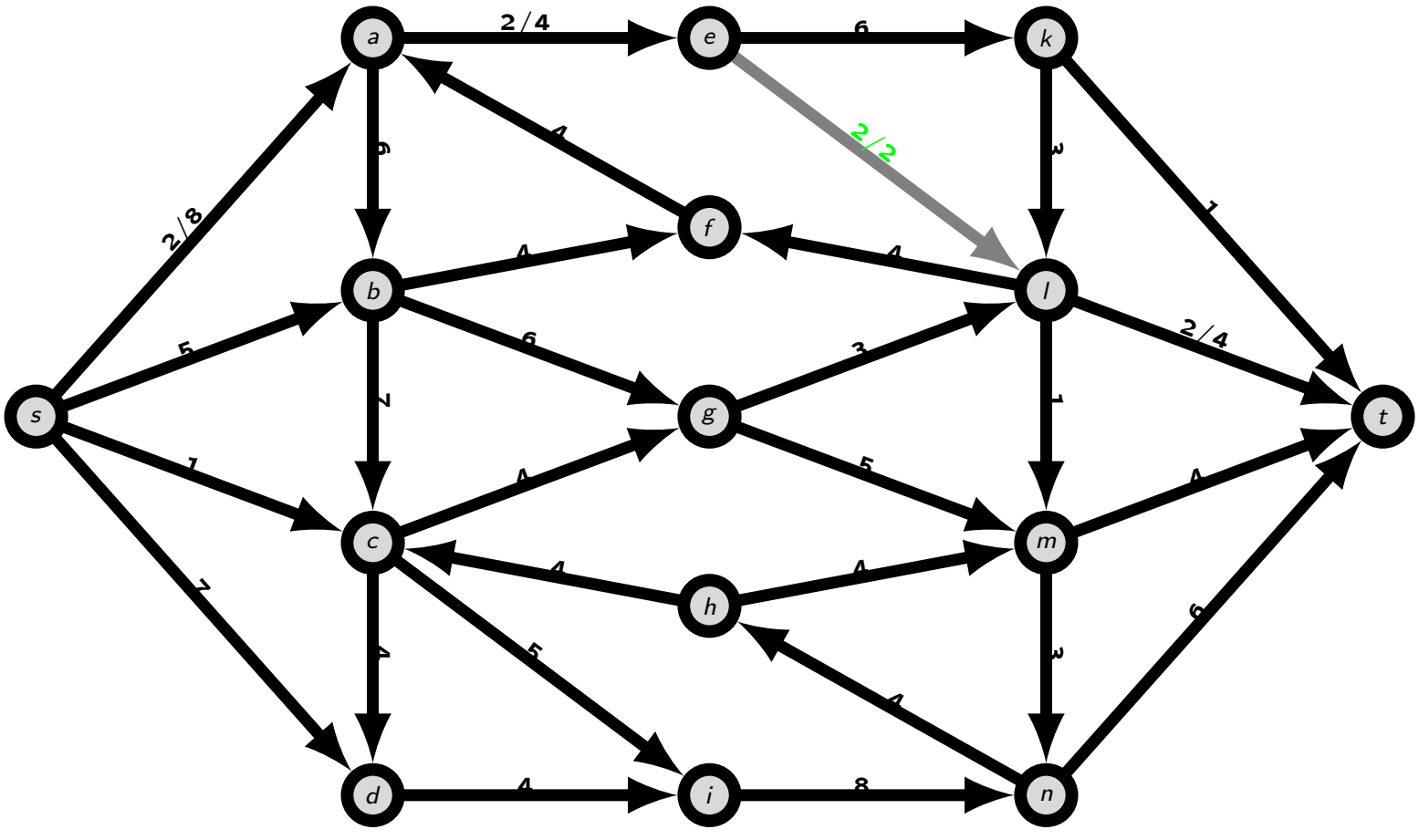
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

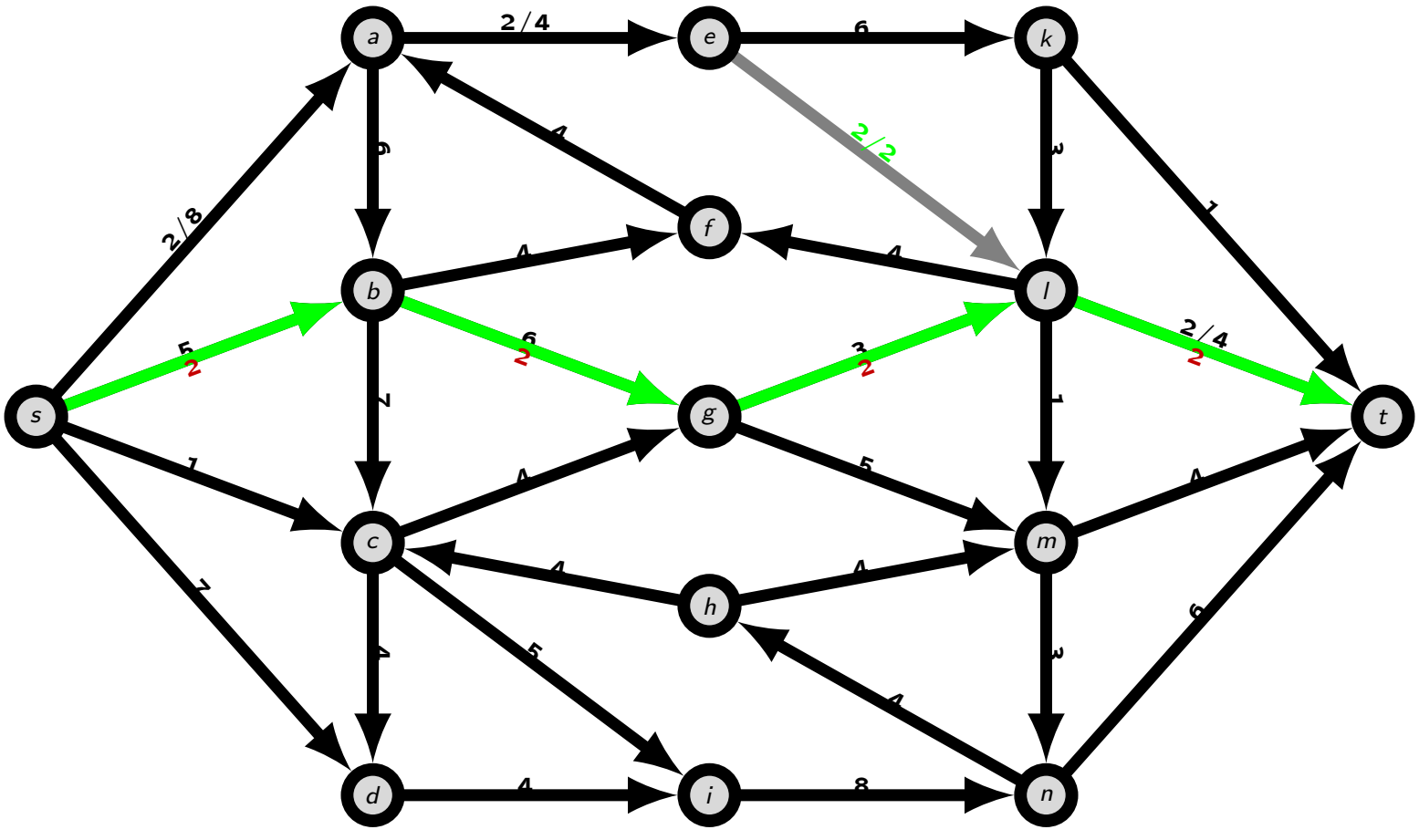
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

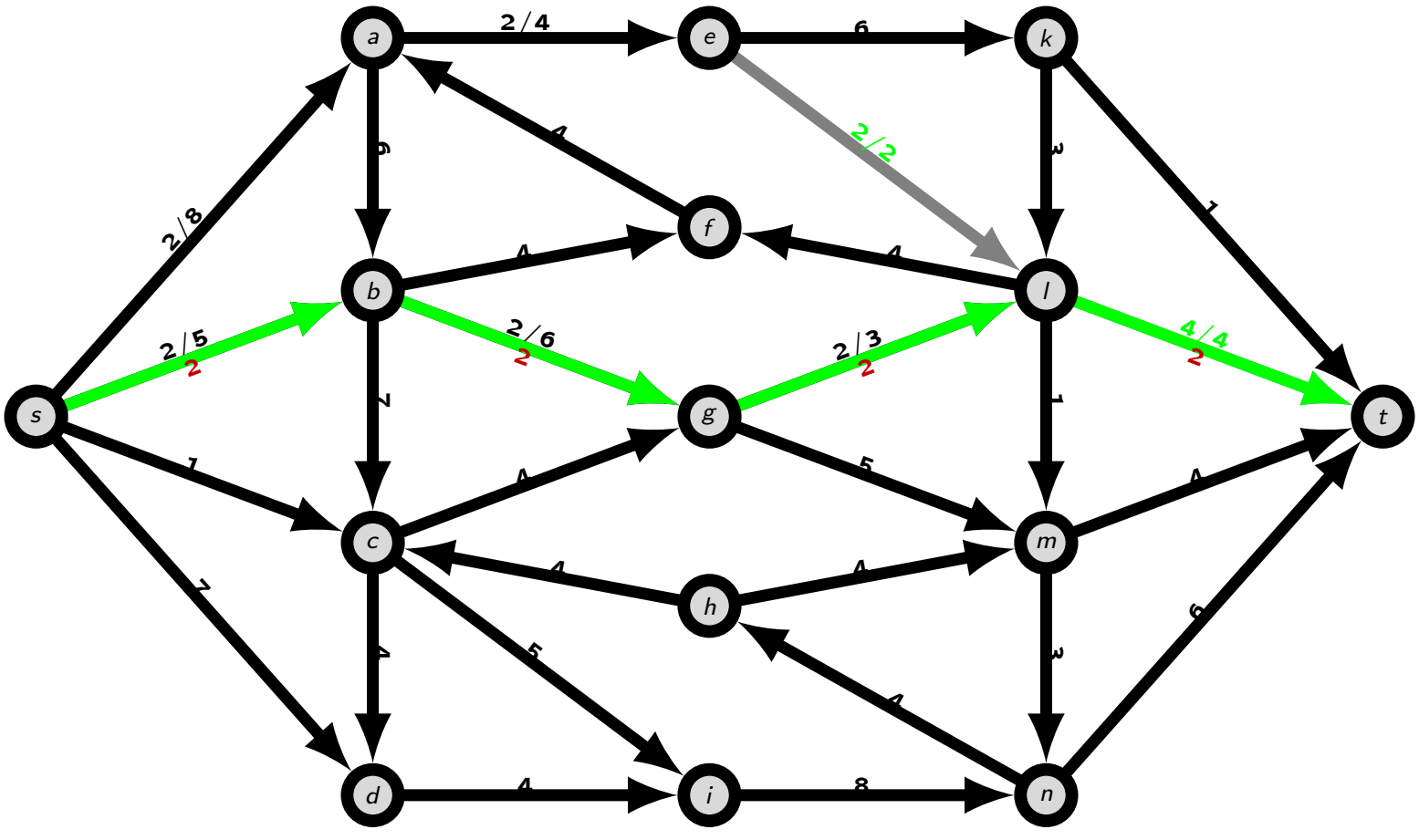
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

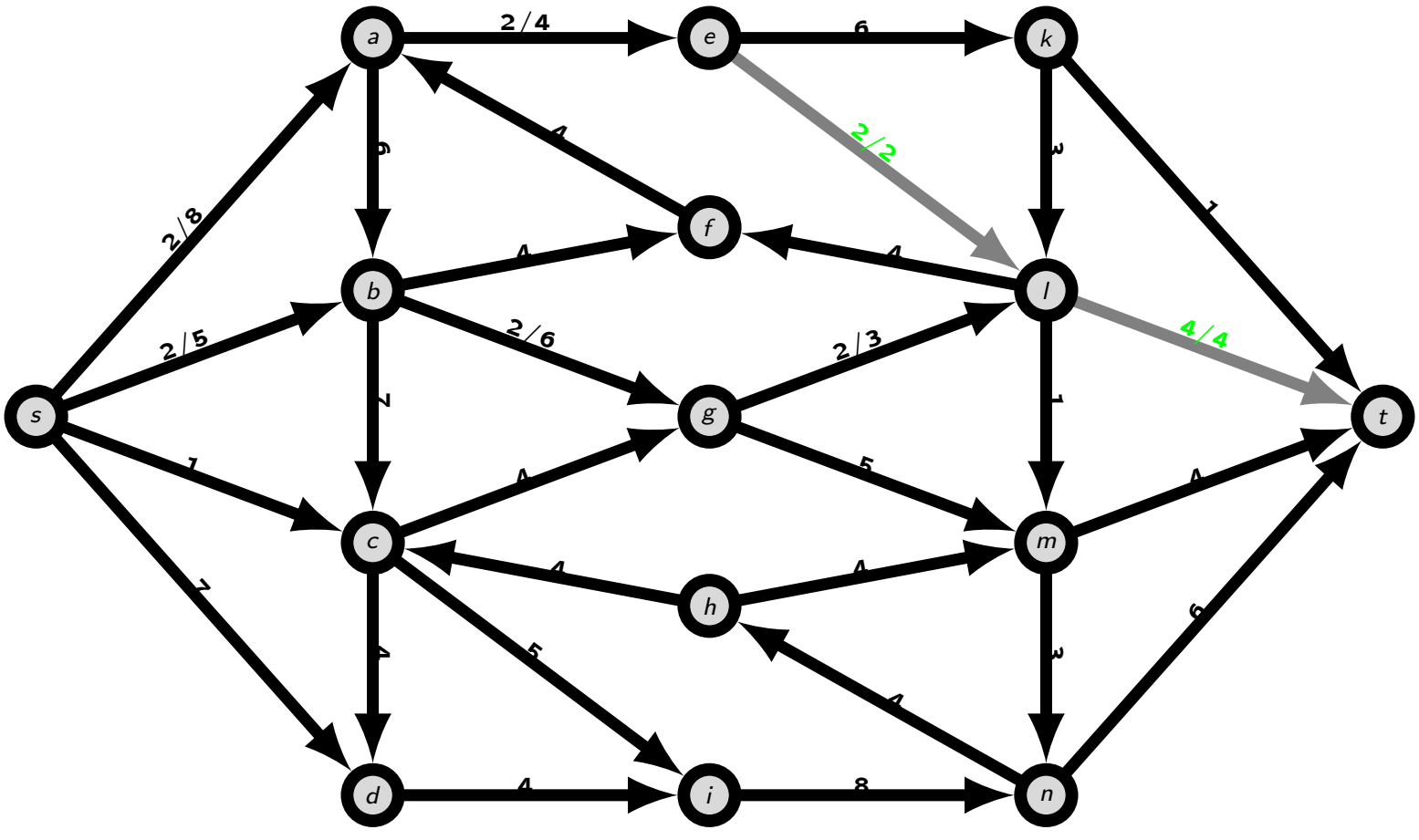
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

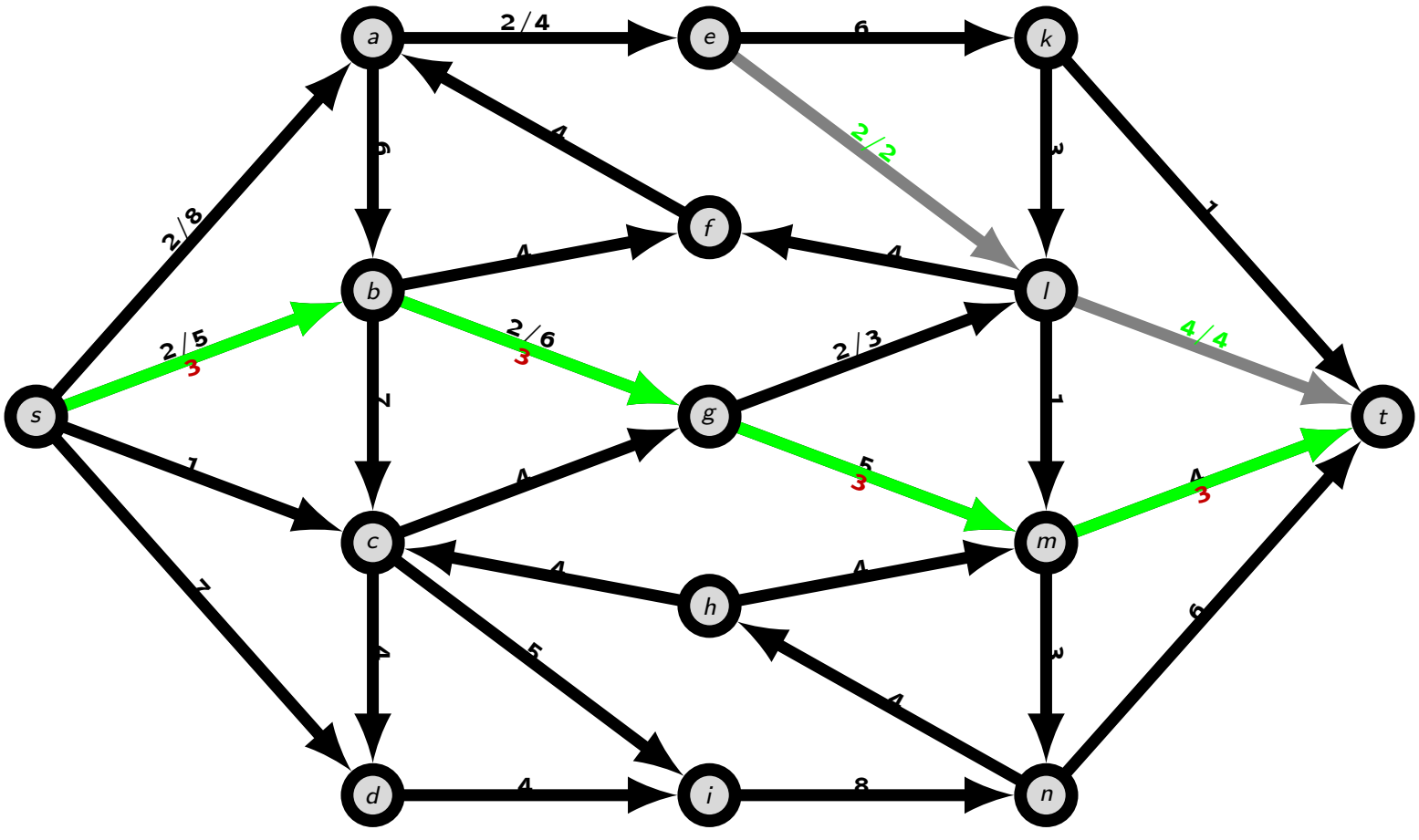
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

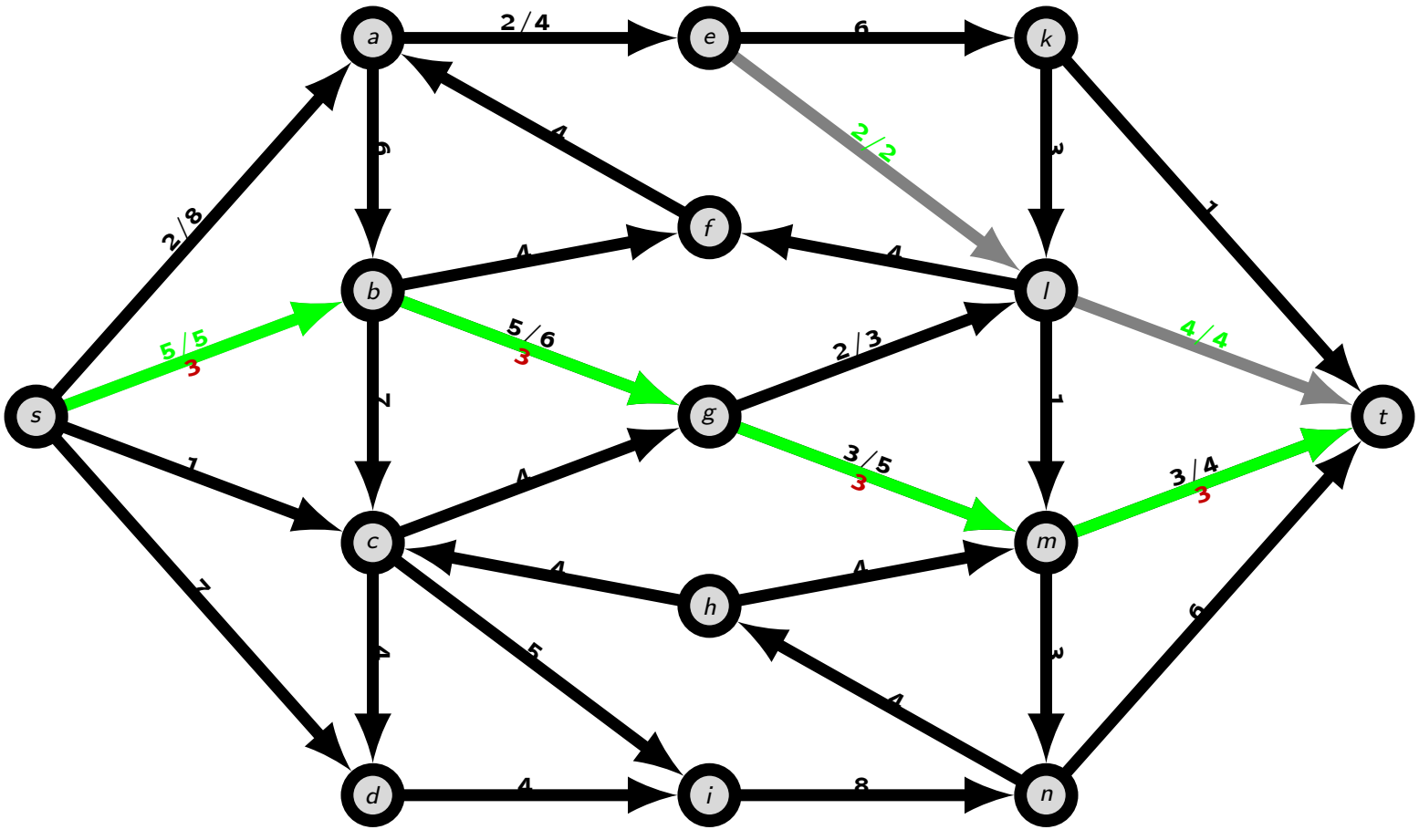
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

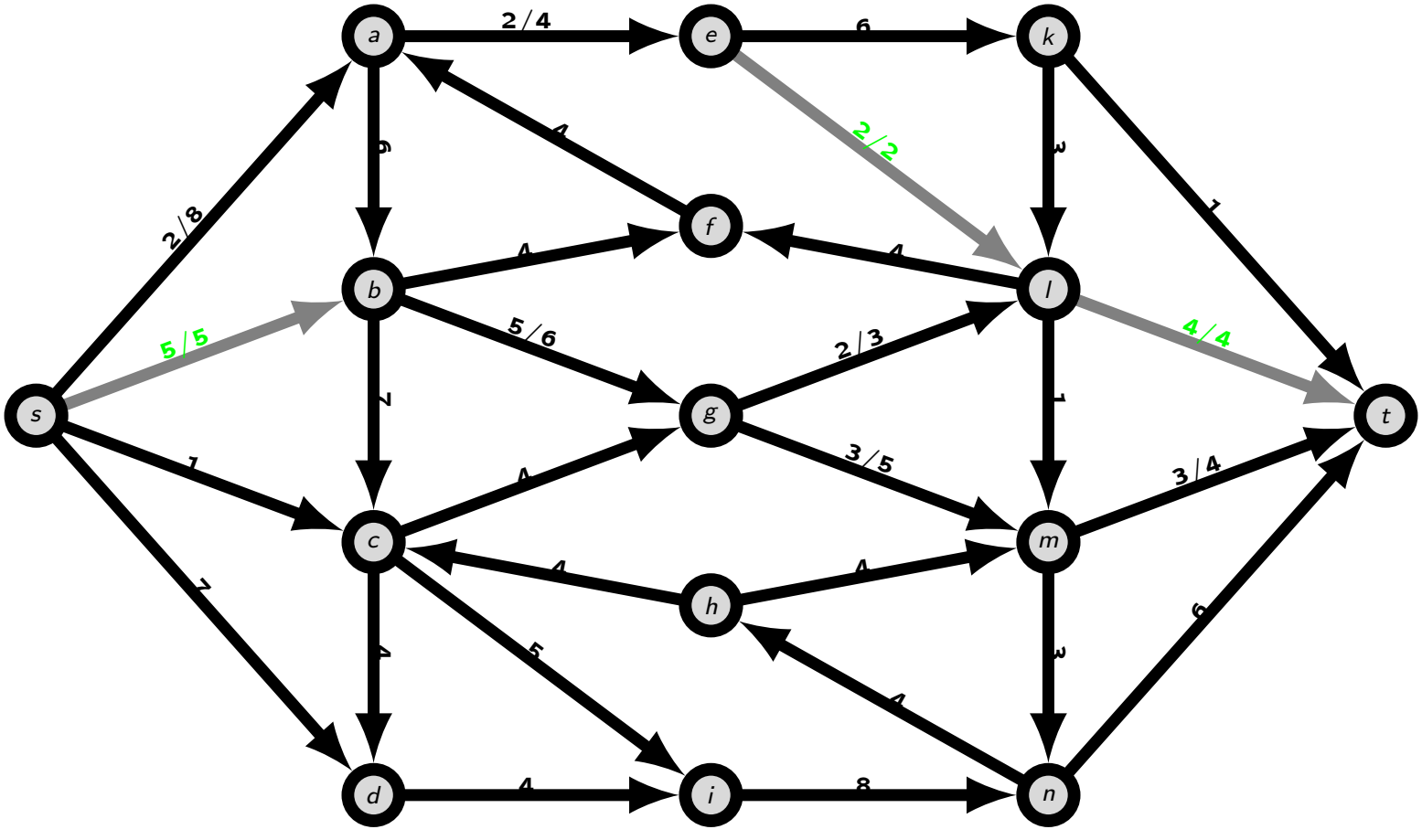
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

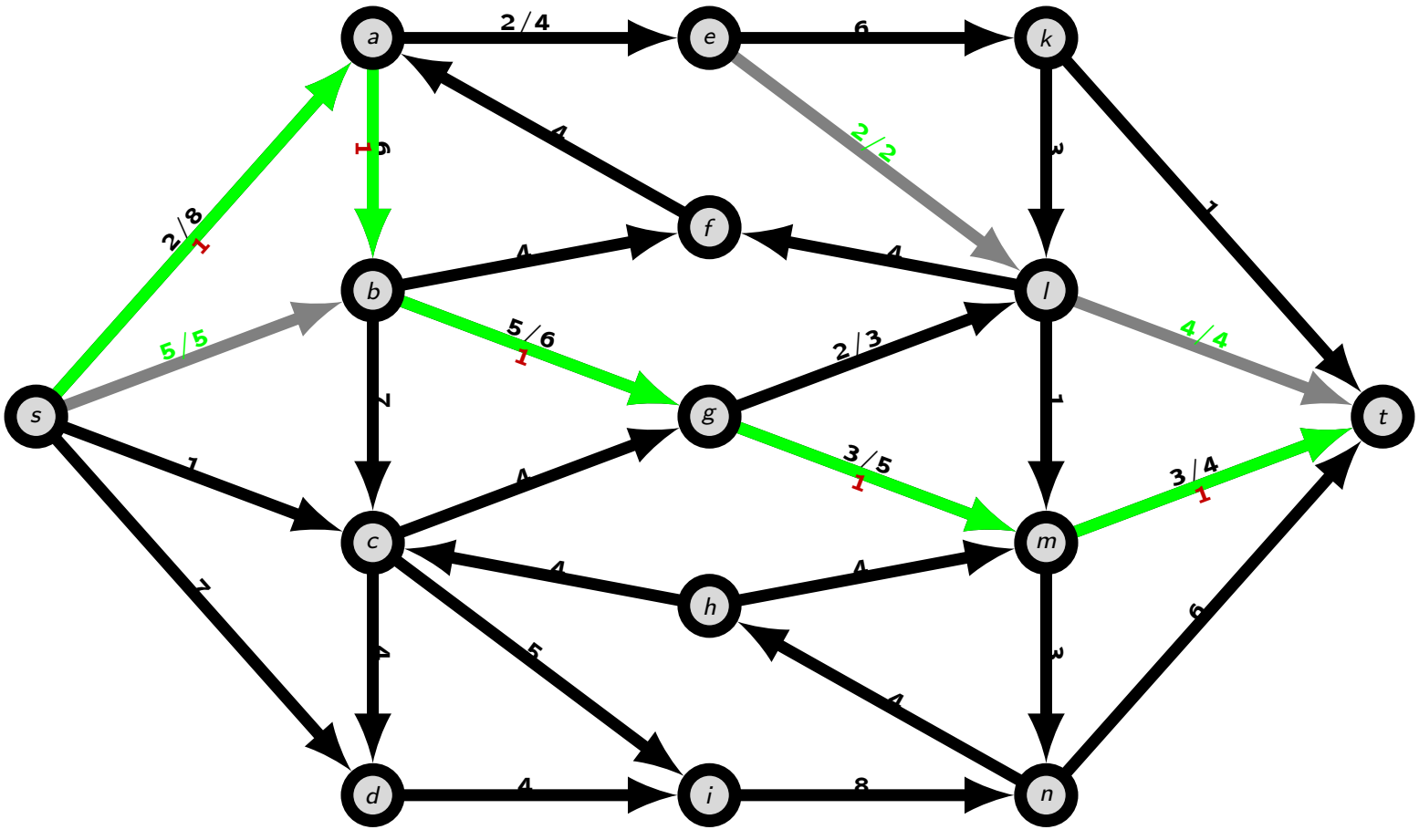
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

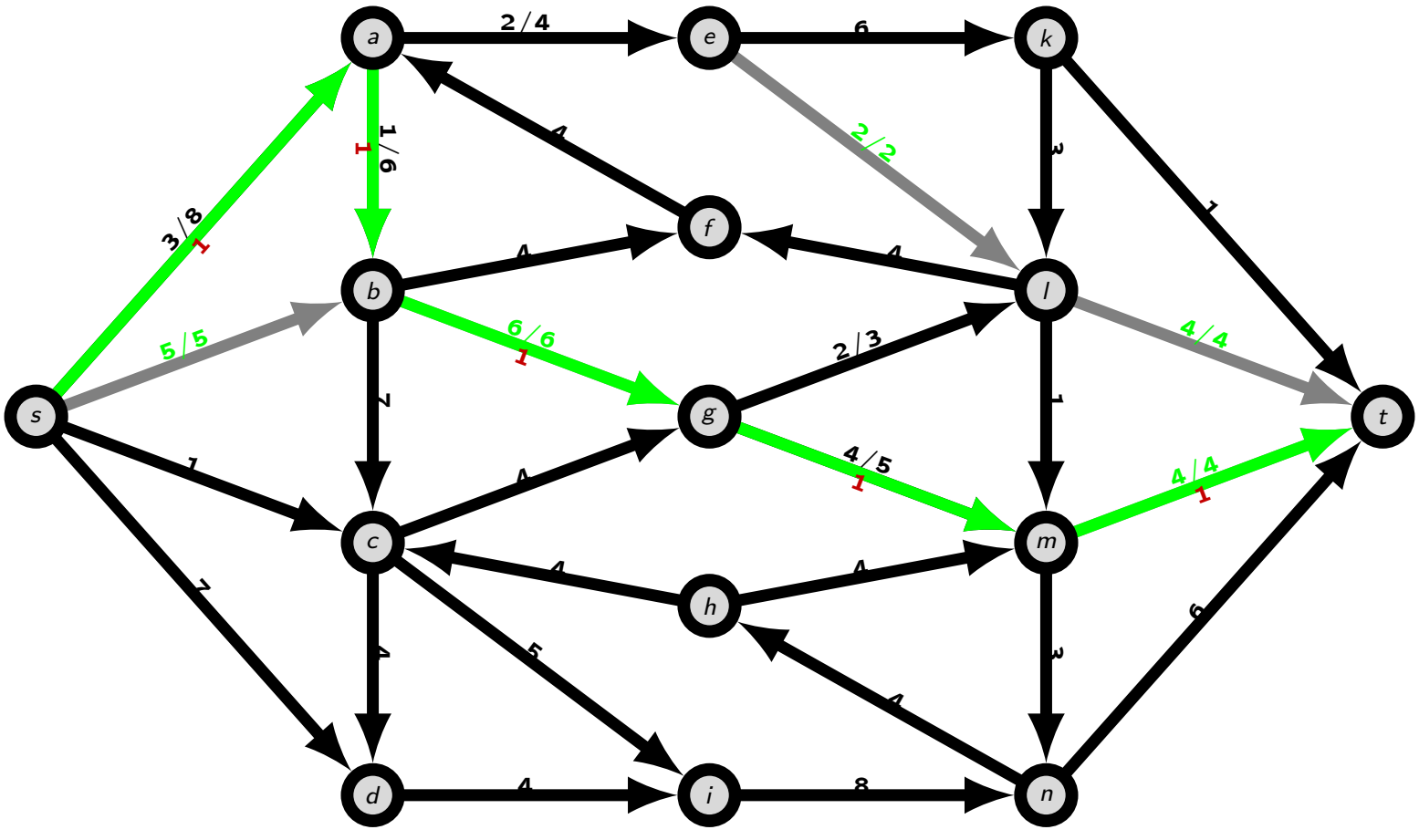
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

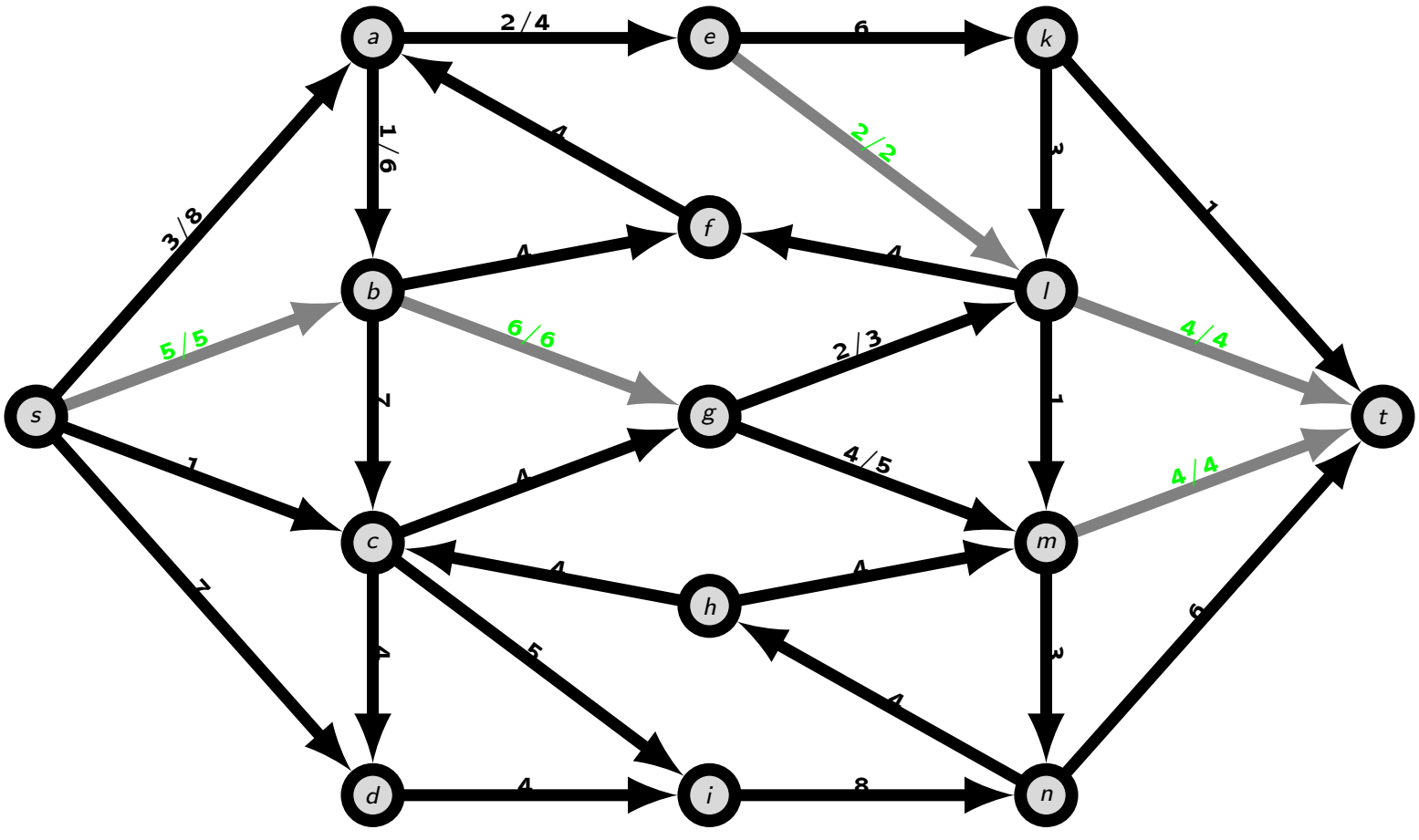
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

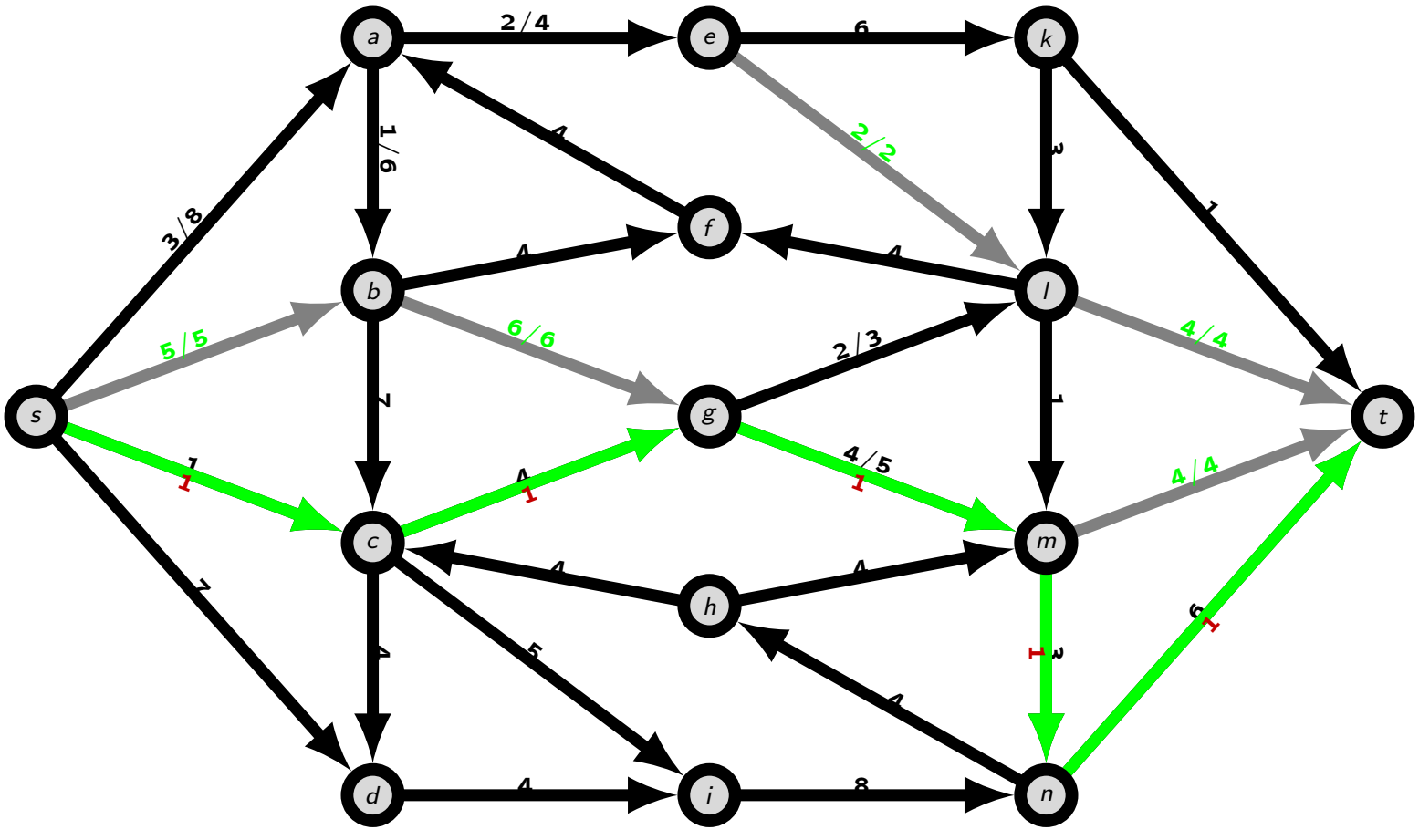
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

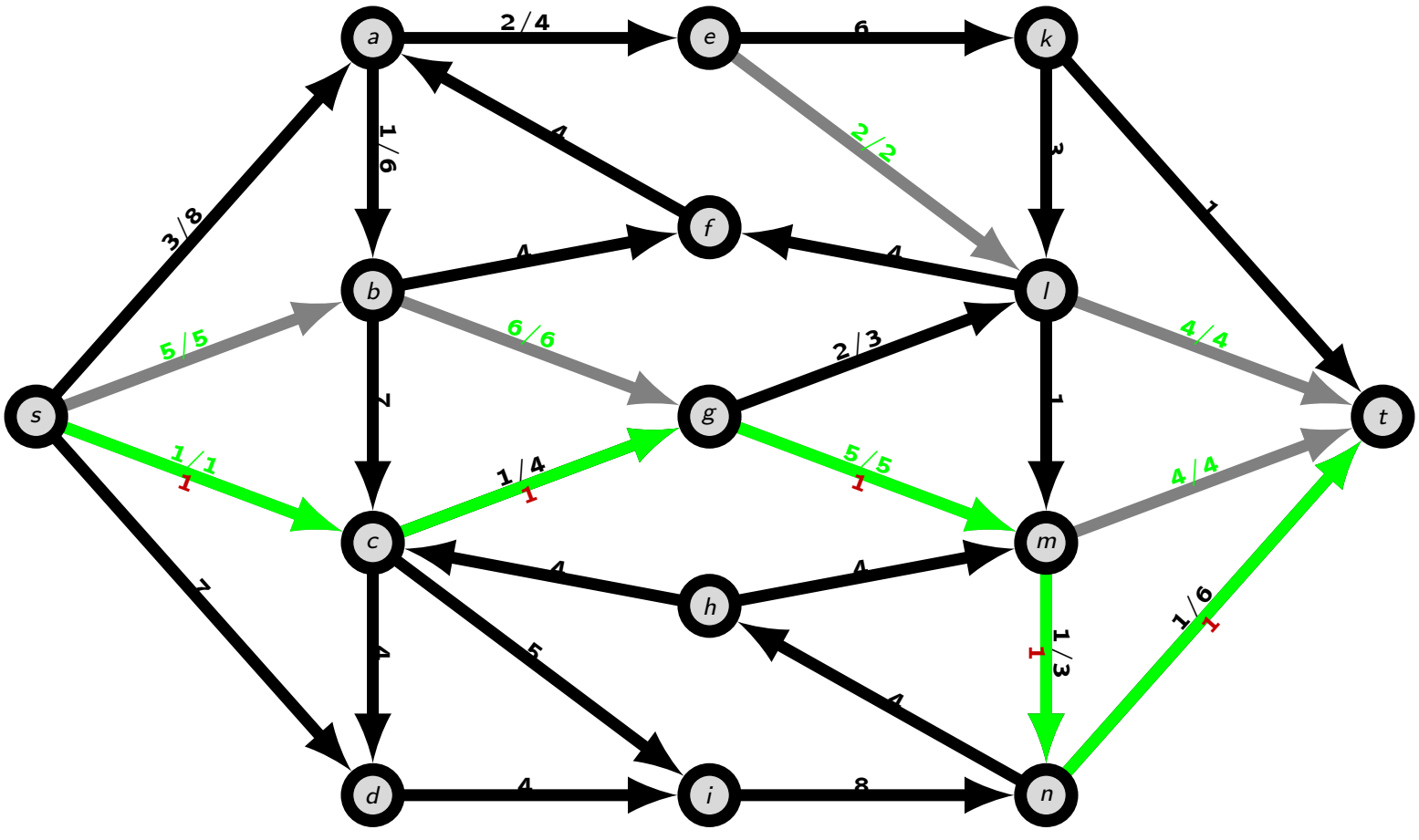
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

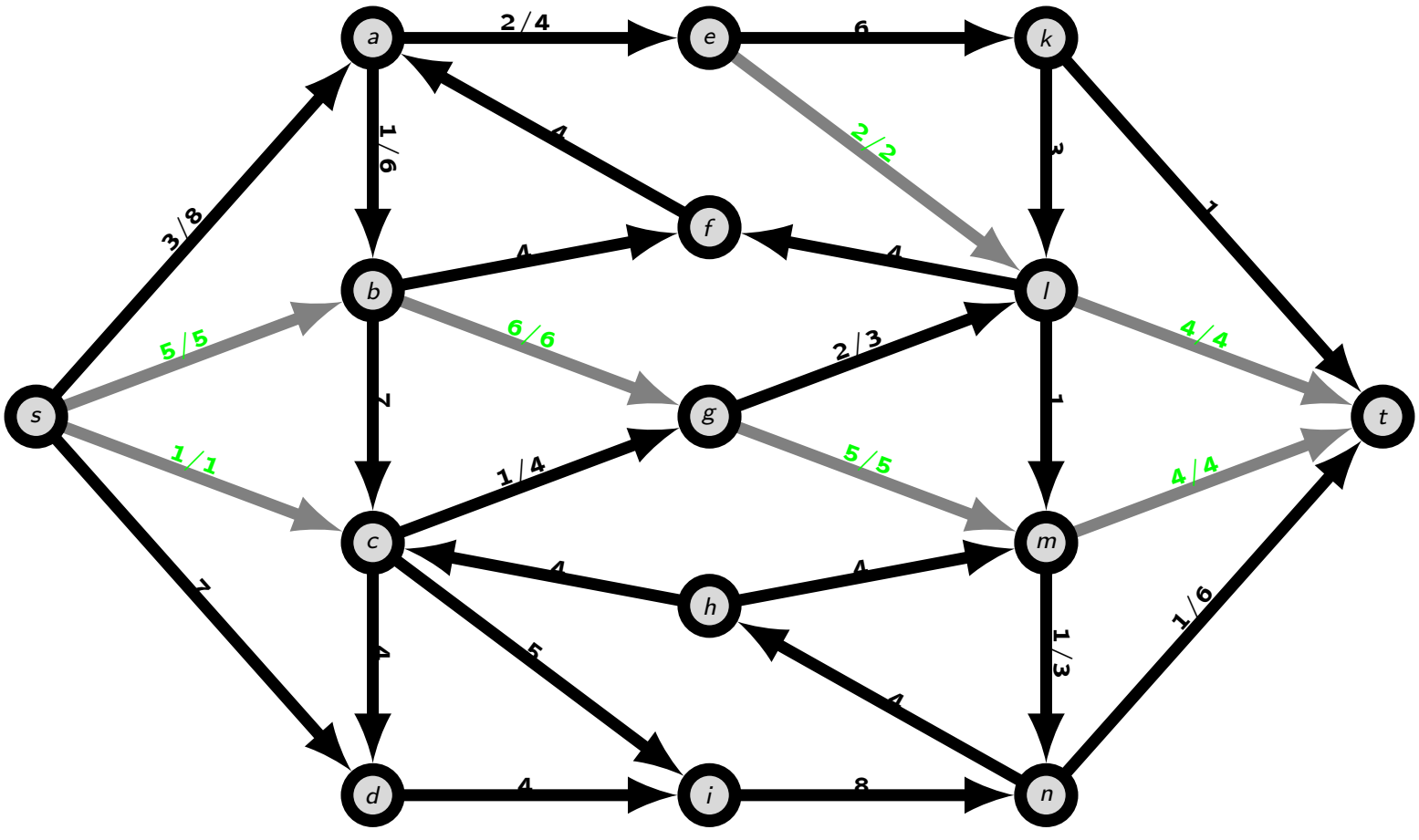
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

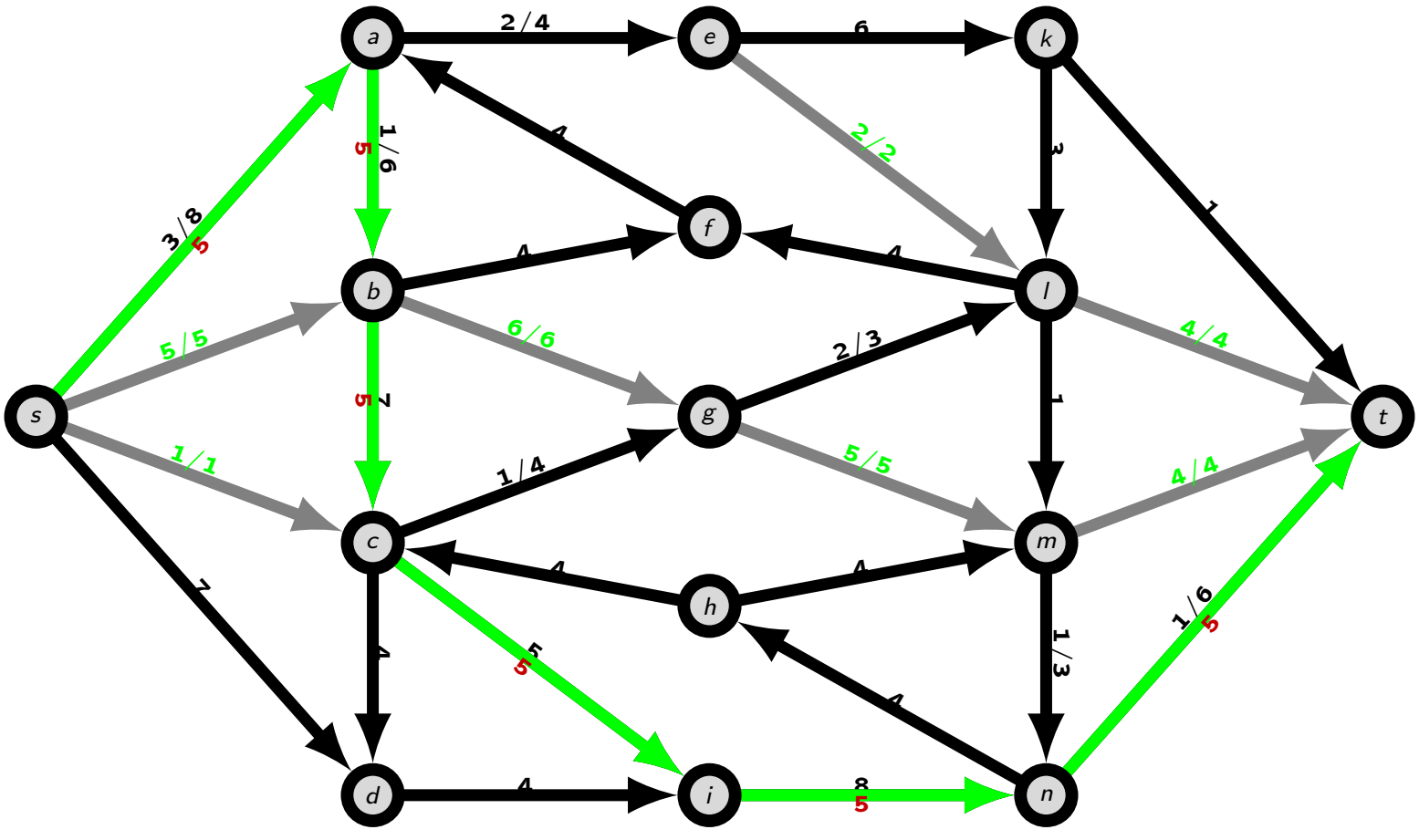
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

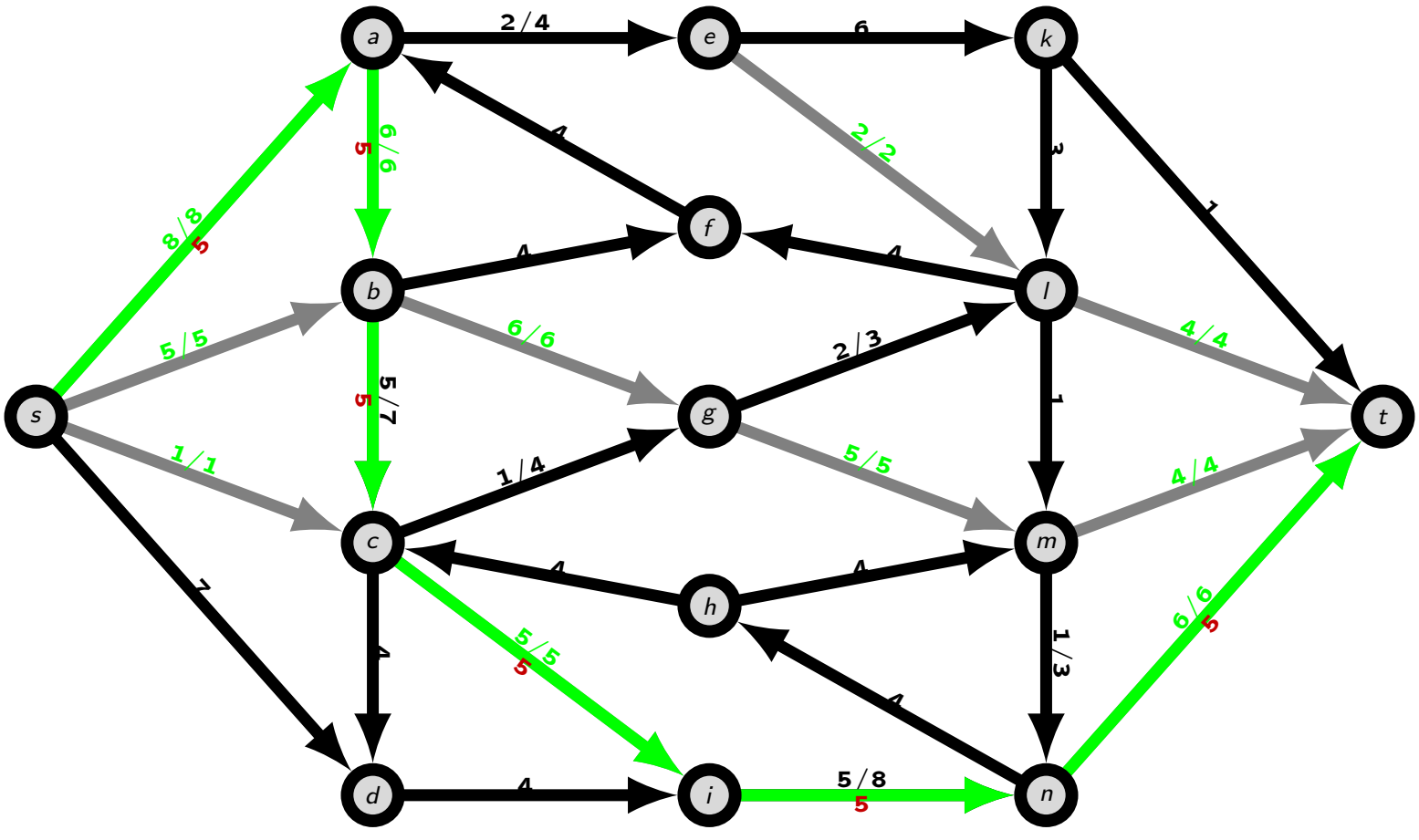
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

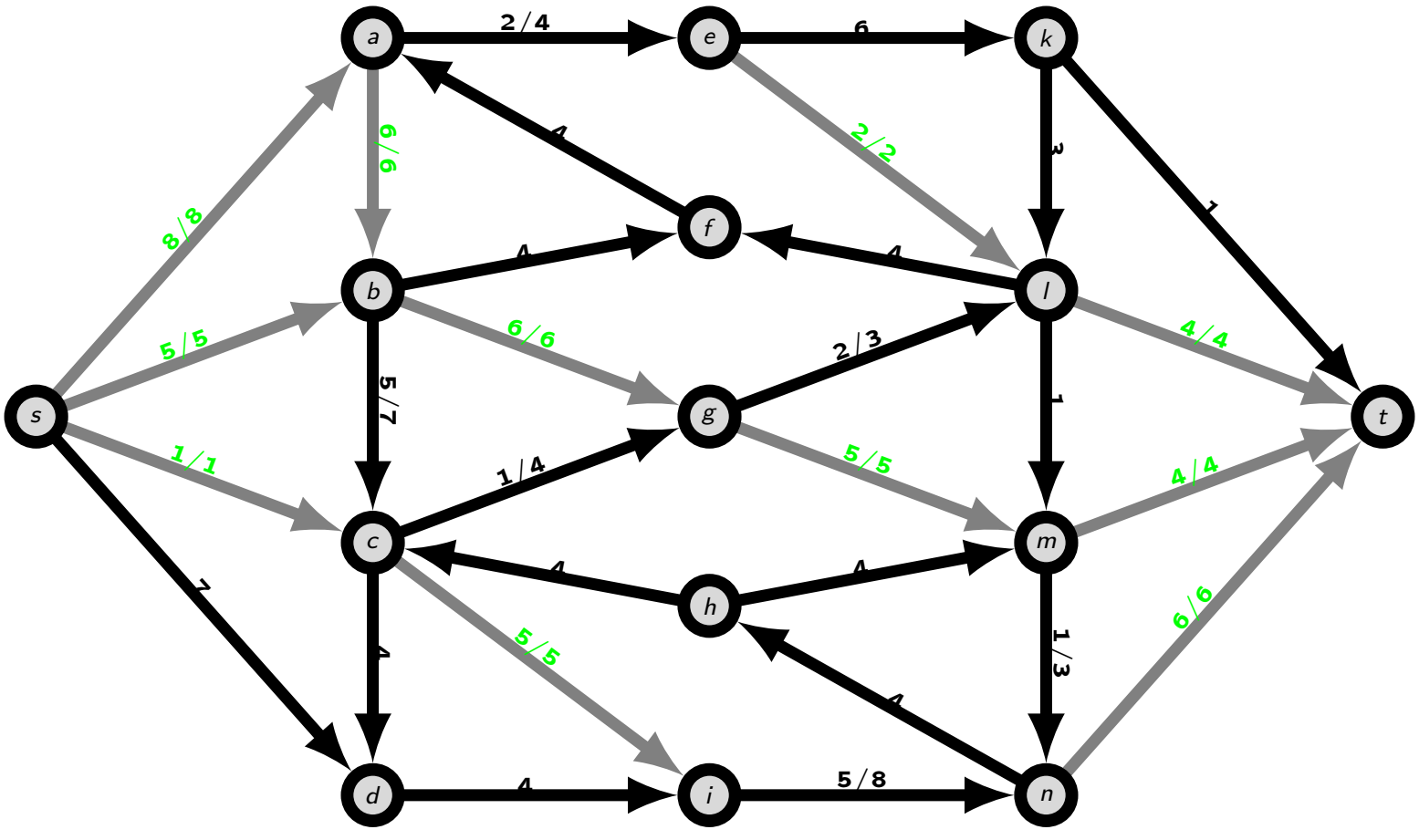
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

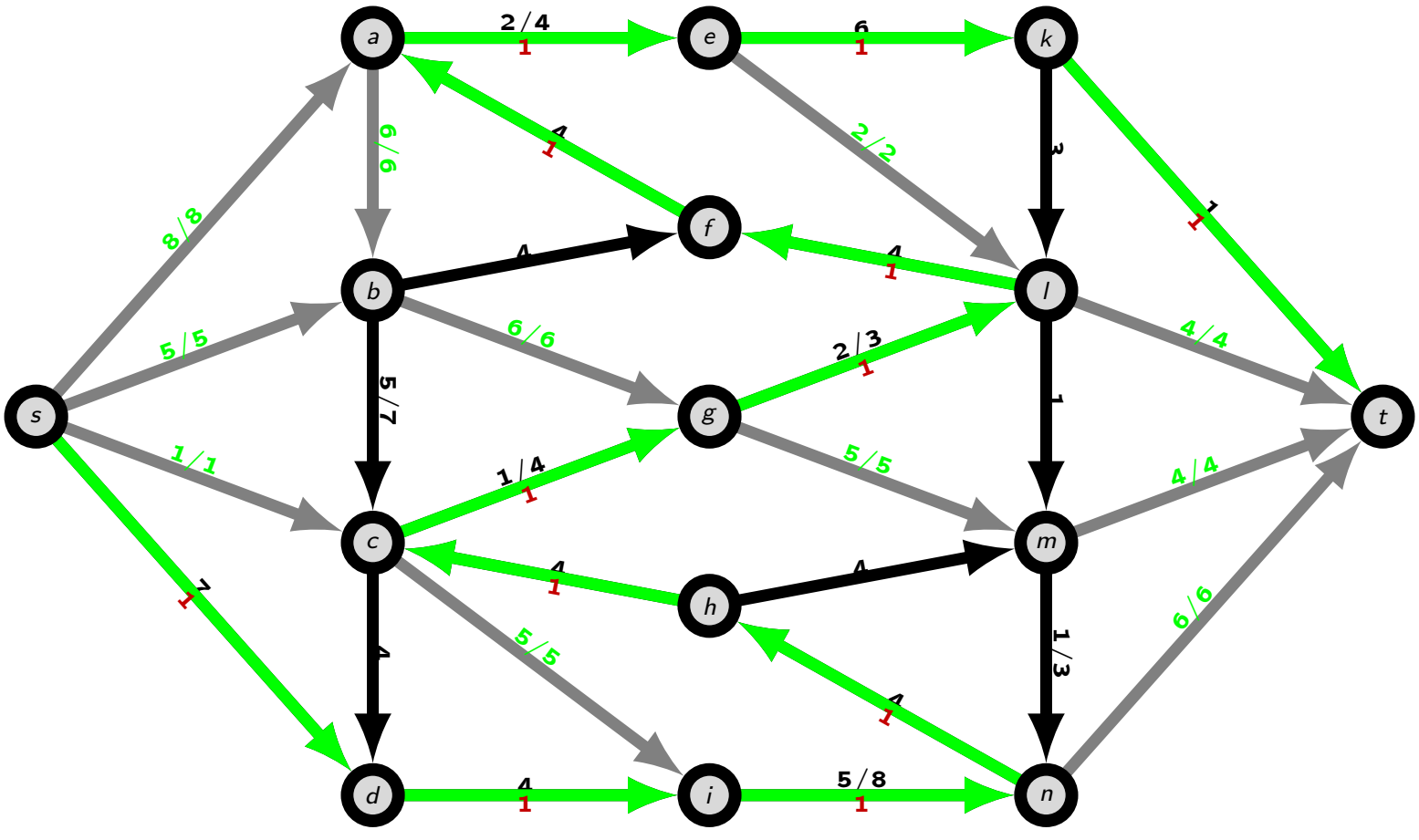
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

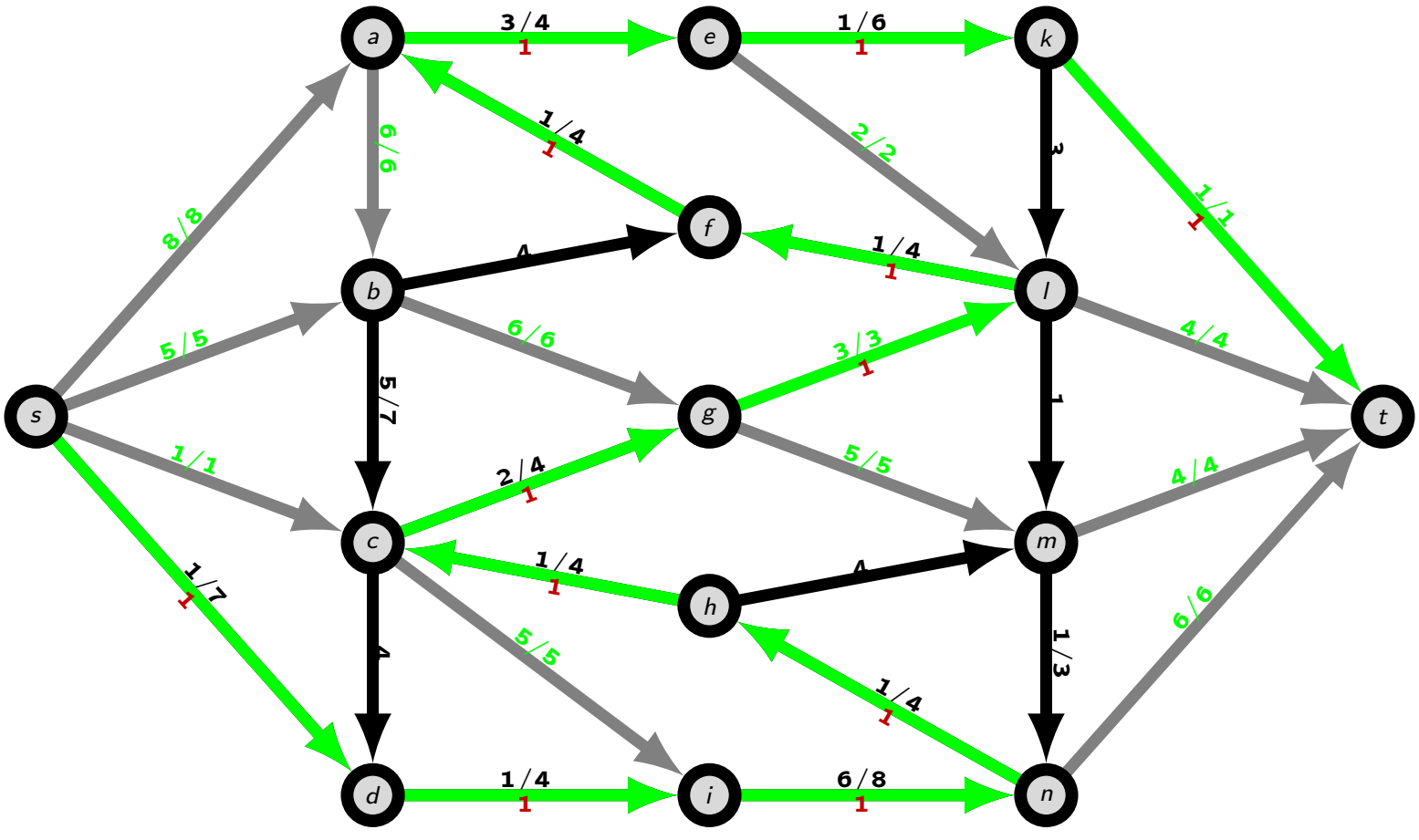
Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



schönes Beispiel

$n = |V|, m = |E|$

Idee: finde in jedem Schritt einen erweiternden Pfad von s nach t.



Situation

$n = |V|, m = |E|$

- Algorithmus findet in den Beispielen einen maximalen Fluss.

Situation

$n = |V|, m = |E|$

- Algorithmus findet in den Beispielen einen maximalen Fluss.
- Algorithmus "findet" eine Barriere, die die Optimalität anzeigt.

Situation

$n = |V|, m = |E|$

- Algorithmus findet in den Beispielen einen maximalen Fluss.
- Algorithmus "findet" eine Barriere, die die Optimalität anzeigt.
- Ziel: Nutze diese Barriere, um Optimalität zu zeigen.

Situation

$n = |V|, m = |E|$

- Algorithmus findet in den Beispielen einen maximalen Fluss.
- Algorithmus "findet" eine Barriere, die die Optimalität anzeigt.
- Ziel: Nutze diese Barriere, um Optimalität zu zeigen.
- Definiere diese Barriere (Cut oder Schnitt).

Situation

$n = |V|, m = |E|$

- Algorithmus findet in den Beispielen einen maximalen Fluss.
- Algorithmus "findet" eine Barriere, die die Optimalität anzeigt.
- Ziel: Nutze diese Barriere, um Optimalität zu zeigen.
- Definiere diese Barriere (Cut oder Schnitt).
- Und zeige dann: Größe des Schnittes und maximaler Fluss sind gleich.

Situation

$n = |V|, m = |E|$

- Algorithmus findet in den Beispielen einen maximalen Fluss.
- Algorithmus "findet" eine Barriere, die die Optimalität anzeigt.
- Ziel: Nutze diese Barriere, um Optimalität zu zeigen.
- Definiere diese Barriere (Cut oder Schnitt).
- Und zeige dann: Größe des Schnittes und maximaler Fluss sind gleich.

Minimaler Cut

$n = |V|, m = |E|$

Definition (Cut und Fluss über Cut)

Sei $G = (V, E, s, t, c)$ gegeben. Ein Cut (Schnitt) ist ein Paar (S, T) mit:

Minimaler Cut

$n = |V|, m = |E|$

Definition (Cut und Fluss über Cut)

Sei $G = (V, E, s, t, c)$ gegeben. Ein Cut (Schnitt) ist ein Paar (S, T) mit:

- $V = S \dot{\cup} T$ und $s \in S$ und $t \in T$.

Minimaler Cut

$n = |V|, m = |E|$

Definition (Cut und Fluss über Cut)

Sei $G = (V, E, s, t, c)$ gegeben. Ein Cut (Schnitt) ist ein Paar (S, T) mit:

- $V = S \dot{\cup} T$ und $s \in S$ und $t \in T$.
- Die Kapazität des Schnittes ist:

$$c(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} c(v, w).$$

Minimaler Cut

$n = |V|, m = |E|$

Definition (Cut und Fluss über Cut)

Sei $G = (V, E, s, t, c)$ gegeben. Ein Cut (Schnitt) ist ein Paar (S, T) mit:

- $V = S \dot{\cup} T$ und $s \in S$ und $t \in T$.
- Die Kapazität des Schnittes ist:

$$c(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} c(v, w).$$

- Für eine Flussfunktion f ist der Fluss über den Schnitt (S, T):

$$f(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} f(v, w) - \sum_{(w,v) \in E, v \in S, w \in T} f(w, v).$$

Minimaler Cut

$n = |V|, m = |E|$

Definition (Cut und Fluss über Cut)

Sei $G = (V, E, s, t, c)$ gegeben. Ein Cut (Schnitt) ist ein Paar (S, T) mit:

- $V = S \dot{\cup} T$ und $s \in S$ und $t \in T$.
- Die Kapazität des Schnittes ist:

$$c(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} c(v, w).$$

- Für eine Flussfunktion f ist der Fluss über den Schnitt (S, T):

$$f(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} f(v, w) - \sum_{(w,v) \in E, v \in S, w \in T} f(w, v).$$

- Zeige nun: Maximaler Fluss $\leqslant$ Minimaler Cut.

Minimaler Cut

$n = |V|, m = |E|$

Definition (Cut und Fluss über Cut)

Sei $G = (V, E, s, t, c)$ gegeben. Ein Cut (Schnitt) ist ein Paar (S, T) mit:

- $V = S \dot{\cup} T$ und $s \in S$ und $t \in T$.
- Die Kapazität des Schnittes ist:

$$c(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} c(v, w).$$

- Für eine Flussfunktion f ist der Fluss über den Schnitt (S, T):

$$f(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} f(v, w) - \sum_{(w,v) \in E, v \in S, w \in T} f(w, v).$$

- Zeige nun: Maximaler Fluss $\leqslant$ Minimaler Cut.
- Zeige weiter: Maximaler Fluss $\geqslant$ Minimaler Cut.

Minimaler Cut

$n = |V|, m = |E|$

Definition (Cut und Fluss über Cut)

Sei $G = (V, E, s, t, c)$ gegeben. Ein Cut (Schnitt) ist ein Paar (S, T) mit:

- $V = S \dot{\cup} T$ und $s \in S$ und $t \in T$.
- Die Kapazität des Schnittes ist:

$$c(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} c(v, w).$$

- Für eine Flussfunktion f ist der Fluss über den Schnitt (S, T):

$$f(S, T) = \sum_{(v,w) \in E, v \in S, w \in T} f(v, w) - \sum_{(w,v) \in E, v \in S, w \in T} f(w, v).$$

- Zeige nun: Maximaler Fluss $\leqslant$ Minimaler Cut.
- Zeige weiter: Maximaler Fluss $\geqslant$ Minimaler Cut.

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

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

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

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

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) \quad = \quad & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) \quad = \quad & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) \quad &= \quad \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= \quad w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Fluss und Schnitt

Lemma (Max-Flow $\leqslant$ Min-Cut)

Für jeden Fluss f und jeden Schnitt (S, T) gilt: $w(f) = f(S, T) \leqslant c(S, T)$.

Beweis: Zeige: $w(f) = f(S, T)$:

- $f(S, T) = \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v)$
- $\sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) = 0$
- $\sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) = 0$
- Zusammengefasst ergibt sich:

$$\begin{aligned} f(S, T) = & \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ & + \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) \\ & + \sum_{(w,v)\in E, v\in S, w\in S} f(w, v) - \sum_{(v,w)\in E, v\in S, w\in S} f(v, w) \end{aligned}$$

- Weiter ergibt sich:

$$\begin{aligned} f(S, T) &= \sum_{(v,w)\in E, v\in S, w\in V} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in V} f(w, v) \\ &= w(f) \text{ wegen Flusserhaltung} \end{aligned}$$

Situation

$n = |V|, m = |E|$

- Es gilt: $w(f) = f(S,T) \leqslant c(S,T)$

$$\sum_{(v,w)\in E, v\in S, w\in T} f(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w,v) \leqslant \sum_{(v,w)\in E, v\in S, w\in T} c(v,w)$$

Situation

$n = |V|, m = |E|$

- Es gilt: $w(f) = f(S,T) \leqslant c(S,T)$

$$\sum_{(v,w)\in E, v\in S, w\in T} f(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w,v) \leqslant \sum_{(v,w)\in E, v\in S, w\in T} c(v,w)$$

- Wir hoffen auf Gleichheit, d.h. hoffen auf:

$$\sum_{(w,v)\in E, v\in S, w\in T} f(w,v) = 0$$

Situation

$n = |V|, m = |E|$

- Es gilt: $w(f) = f(S,T) \leqslant c(S,T)$

$$\sum_{(v,w)\in E, v\in S, w\in T} f(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w,v) \leqslant \sum_{(v,w)\in E, v\in S, w\in T} c(v,w)$$

- Wir hoffen auf Gleichheit, d.h. hoffen auf:

$$\sum_{(w,v)\in E, v\in S, w\in T} f(w,v) = 0$$

- D.h. die "Rückwärtskanten" sollen keinen Fluss führen.

Situation

$n = |V|, m = |E|$

- Es gilt: $w(f) = f(S,T) \leqslant c(S,T)$

$$\sum_{(v,w)\in E, v\in S, w\in T} f(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w,v) \leqslant \sum_{(v,w)\in E, v\in S, w\in T} c(v,w)$$

- Wir hoffen auf Gleichheit, d.h. hoffen auf:

$$\sum_{(w,v)\in E, v\in S, w\in T} f(w,v) = 0$$

- D.h. die "Rückwärtskanten" sollen keinen Fluss führen.
- War das in den bisherigen Beispielen schon mal der Fall?

Situation

$n = |V|, m = |E|$

- Es gilt: $w(f) = f(S,T) \leqslant c(S,T)$

$$\sum_{(v,w)\in E, v\in S, w\in T} f(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w,v) \leqslant \sum_{(v,w)\in E, v\in S, w\in T} c(v,w)$$

- Wir hoffen auf Gleichheit, d.h. hoffen auf:

$$\sum_{(w,v)\in E, v\in S, w\in T} f(w,v) = 0$$

- D.h. die "Rückwärtskanten" sollen keinen Fluss führen.
- War das in den bisherigen Beispielen schon mal der Fall?
- Betrachte dazu das folgende Beispiel.

Situation

$n = |V|, m = |E|$

- Es gilt: $w(f) = f(S,T) \leqslant c(S,T)$

$$\sum_{(v,w)\in E, v\in S, w\in T} f(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w,v) \leqslant \sum_{(v,w)\in E, v\in S, w\in T} c(v,w)$$

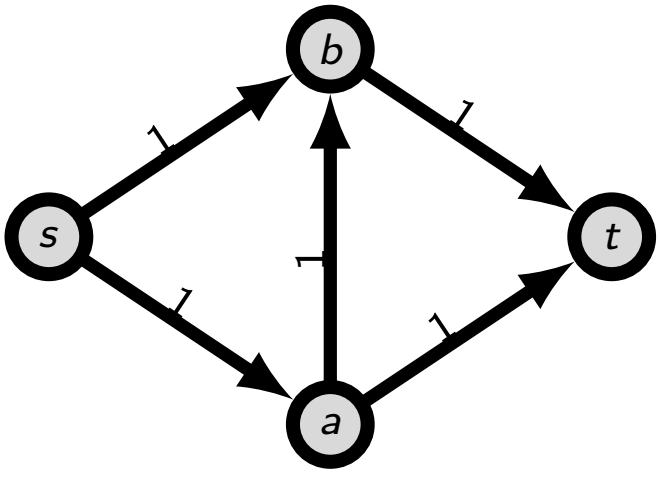
- Wir hoffen auf Gleichheit, d.h. hoffen auf:

$$\sum_{(w,v)\in E, v\in S, w\in T} f(w,v) = 0$$

- D.h. die "Rückwärtskanten" sollen keinen Fluss führen.
- War das in den bisherigen Beispielen schon mal der Fall?
- Betrachte dazu das folgende Beispiel.

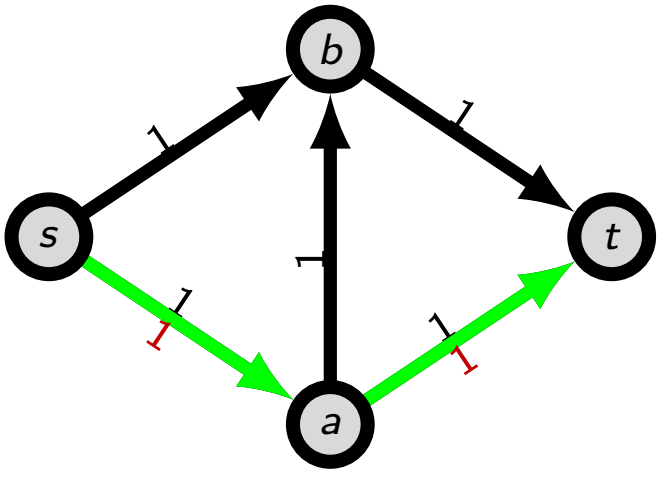
Beispiele

$n = |V|, m = |E|$



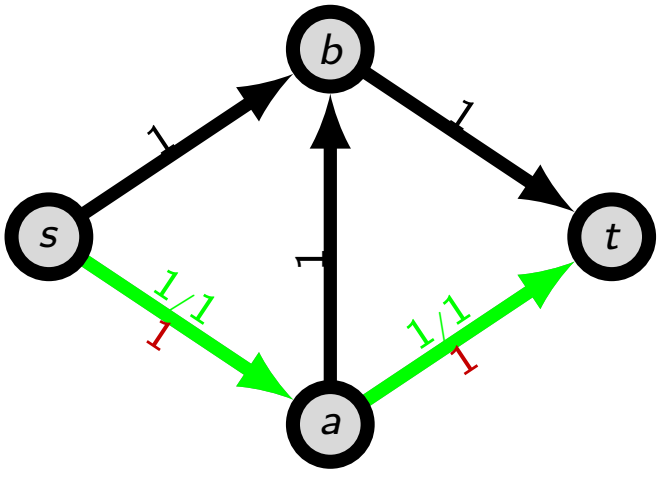
Beispiele

$n = |V|, m = |E|$



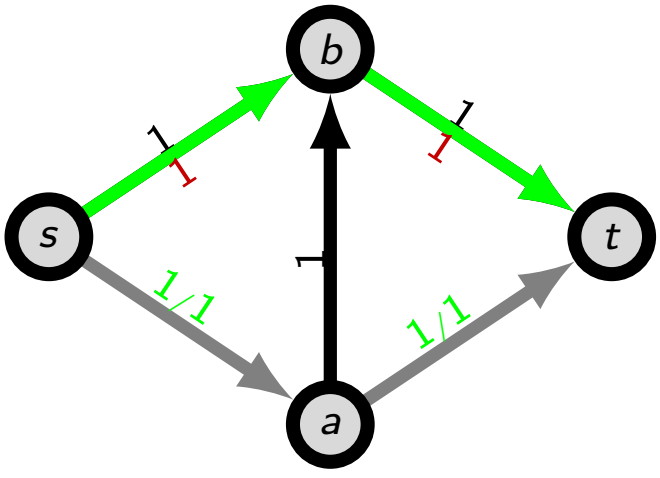
Beispiele

$n = |V|, m = |E|$



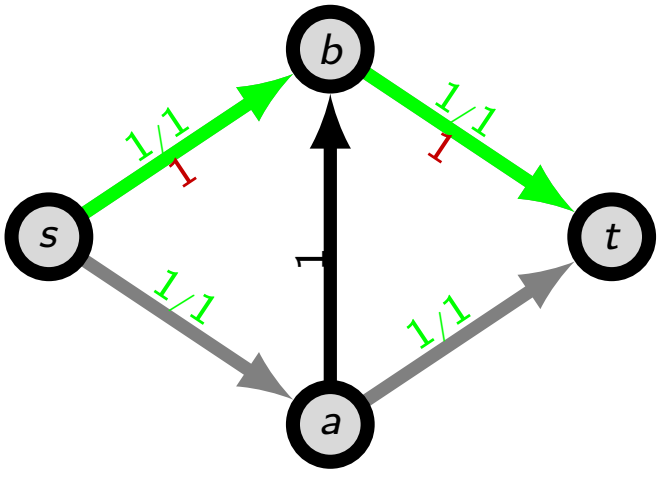
Beispiele

$n = |V|, m = |E|$



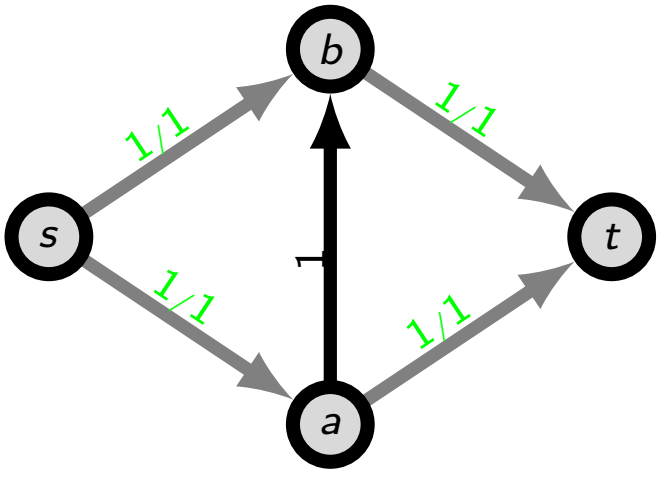
Beispiele

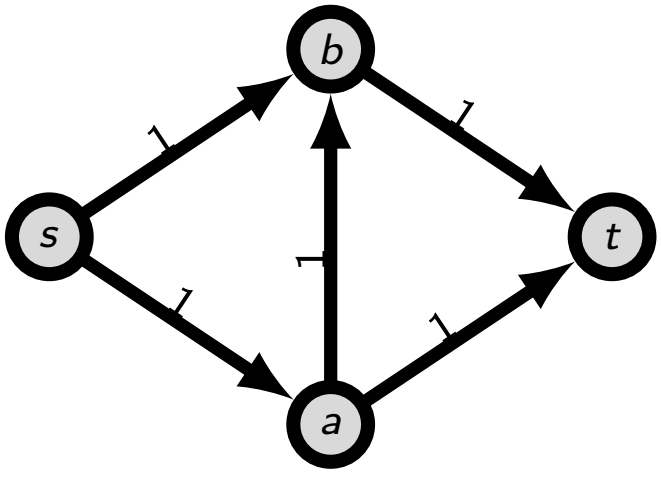
$n = |V|, m = |E|$



Beispiele

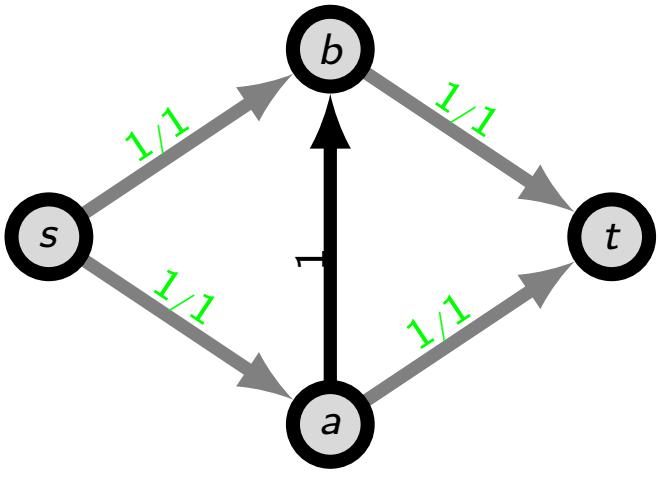
$n = |V|, m = |E|$

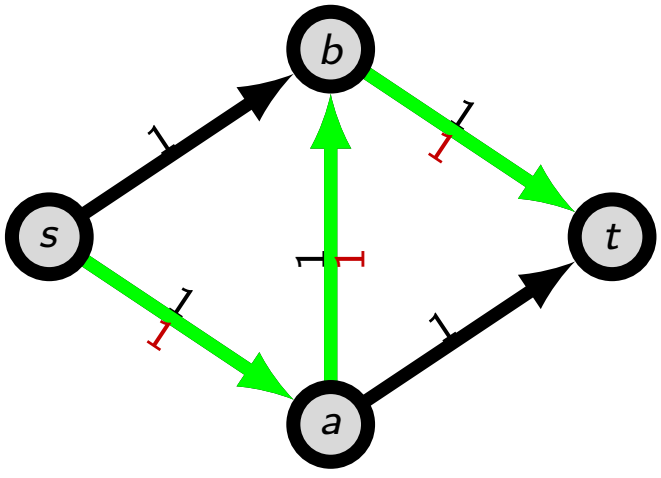




Beispiele

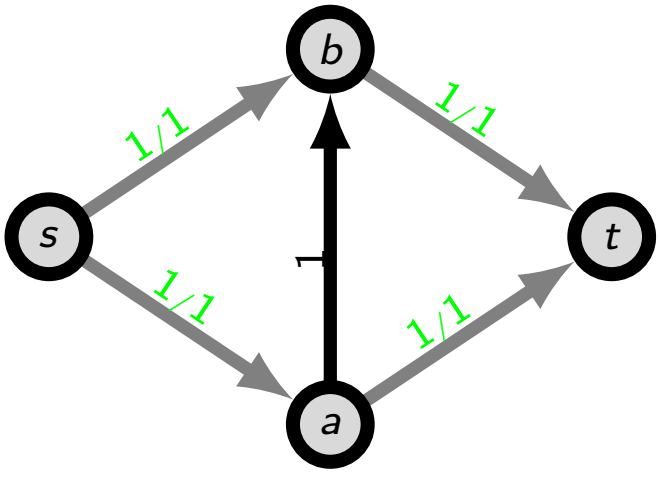
$n = |V|, m = |E|$

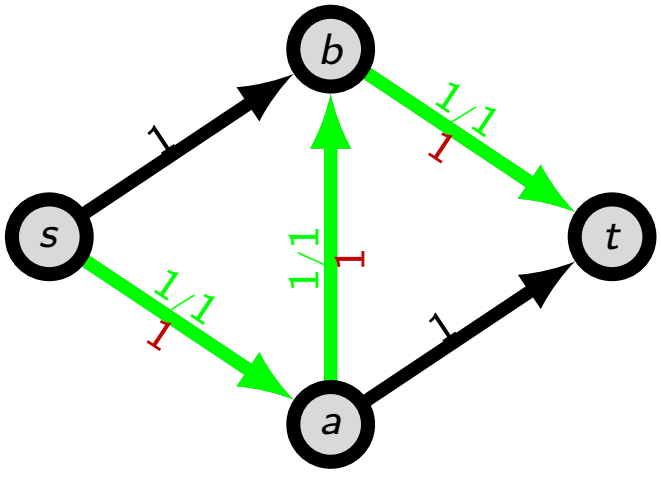




Beispiele

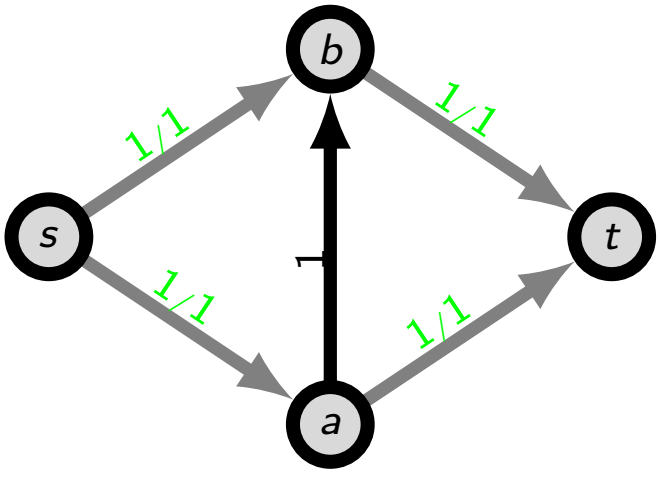
$n = |V|, m = |E|$

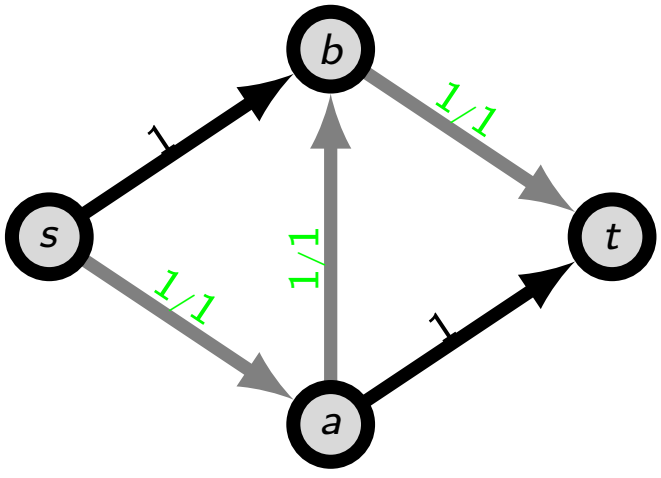




Beispiele

$n = |V|, m = |E|$



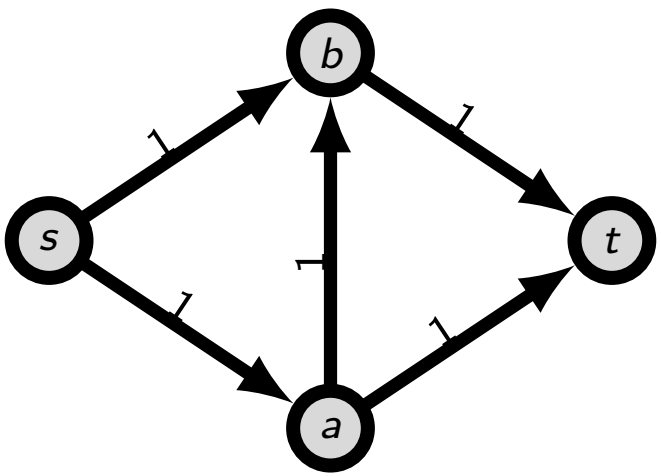


Da ist nicht immer ein Cut zu sehen!

Neue Situation

$n = |V|, m = |E|$

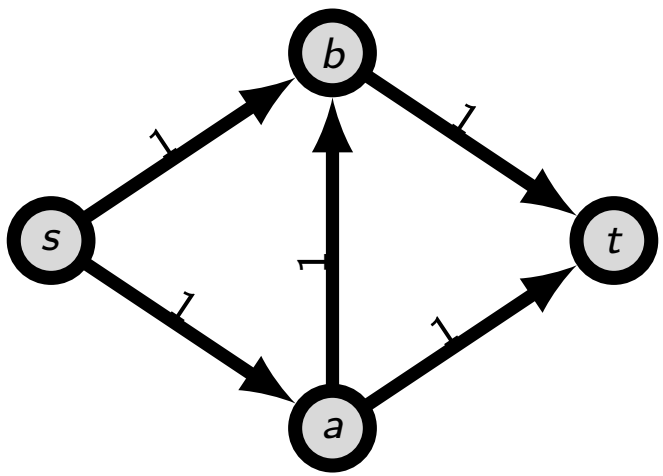
- Beim bisherigen Verfahren können Rückwärtskanten mit einem nichtleeren Fluss auftreten.



Neue Situation

$n = |V|, m = |E|$

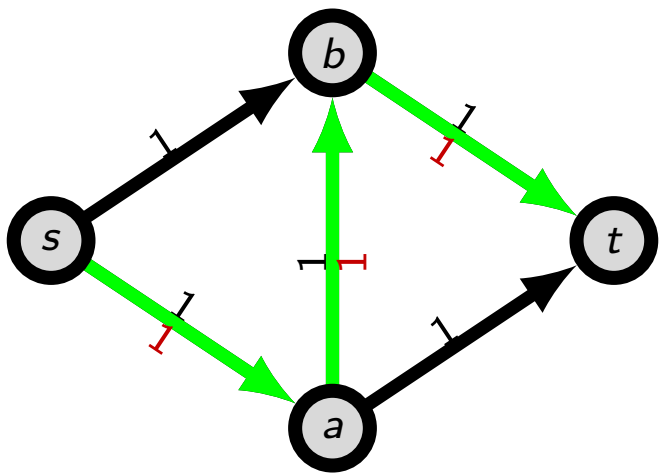
- Beim bisherigen Verfahren können Rückwärtskanten mit einem nichtleeren Fluss auftreten.
- Der bisherige Algorithmus liefert nicht immer den maximalen Fluss.



Neue Situation

$n = |V|, m = |E|$

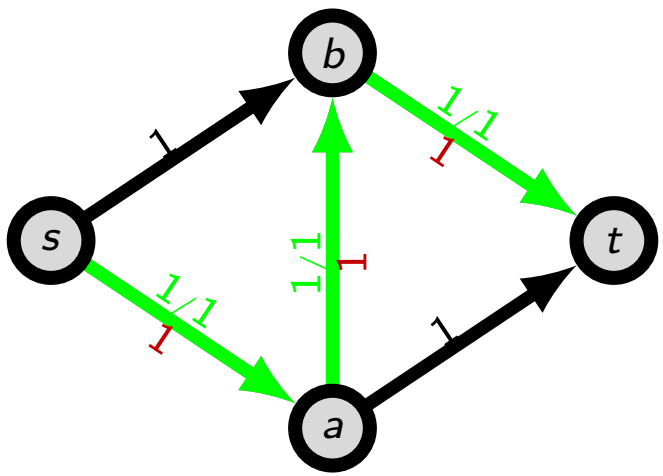
- Beim bisherigen Verfahren können Rückwärtskanten mit einem nichtleeren Fluss auftreten.
- Der bisherige Algorithmus liefert nicht immer den maximalen Fluss.
- Fehlentscheidungen müssen behoben werden.



Neue Situation

$n = |V|, m = |E|$

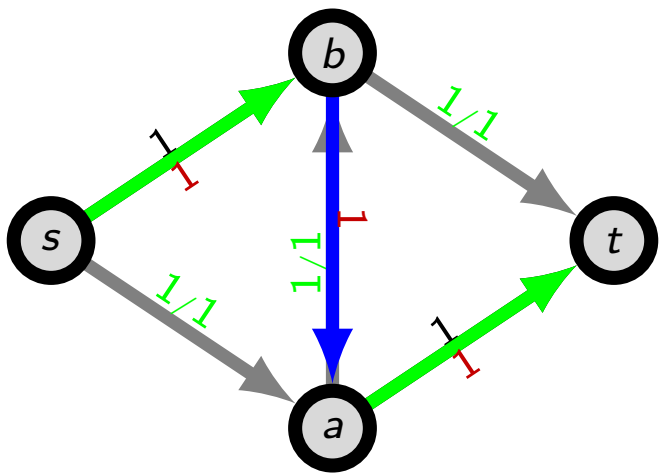
- Beim bisherigen Verfahren können Rückwärtskanten mit einem nichtleeren Fluss auftreten.
- Der bisherige Algorithmus liefert nicht immer den maximalen Fluss.
- Fehlentscheidungen müssen behoben werden.
- Der Fluss über die Rückwärtskante muss zurückgenommen werden!



Neue Situation

$n = |V|, m = |E|$

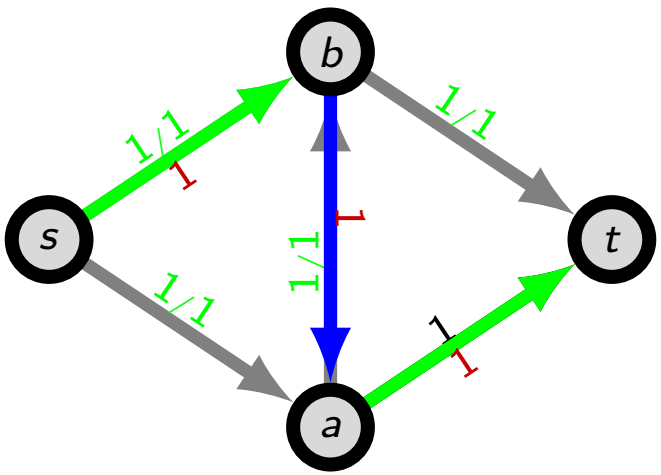
- Beim bisherigen Verfahren können Rückwärtskanten mit einem nichtleeren Fluss auftreten.
- Der bisherige Algorithmus liefert nicht immer den maximalen Fluss.
- Fehlentscheidungen müssen behoben werden.
- Der Fluss über die Rückwärtskante muss zurückgenommen werden!



Neue Situation

$n = |V|, m = |E|$

- Beim bisherigen Verfahren können Rückwärtskanten mit einem nichtleeren Fluss auftreten.
- Der bisherige Algorithmus liefert nicht immer den maximalen Fluss.
- Fehlentscheidungen müssen behoben werden.
- Der Fluss über die Rückwärtskante muss zurückgenommen werden!



Restnetzwerk

Wir nehmen o.B.d.A bei einem Netzwerk $G = (V, E, s, t, c)$ an:

- $\forall e = (v, w) \in E : f(v, w) \cdot f(w, v) = 0$.

Restnetzwerk

Wir nehmen o.B.d.A bei einem Netzwerk $G = (V, E, s, t, c)$ an:

- $\forall e = (v, w) \in E : f(v, w) \cdot f(w, v) = 0$.
- Alternativ o.E.d.A: $(v, w) \in E \Longrightarrow (w, v) \notin E$.

Restnetzwerk

$n = |V|, m = |E|$

Wir nehmen o.B.d.A bei einem Netzwerk $G = (V, E, s, t, c)$ an:

- $\forall e = (v, w) \in E : f(v, w) \cdot f(w, v) = 0$.
- Alternativ o.E.d.A: $(v, w) \in E \Longrightarrow (w, v) \notin E$.

Restnetzwerk

$n = |V|, m = |E|$

Wir nehmen o.B.d.A bei einem Netzwerk $G = (V, E, s, t, c)$ an:

- $\forall e = (v, w) \in E : f(v, w) \cdot f(w, v) = 0$.
- Alternativ o.E.d.A: $(v, w) \in E \Longrightarrow (w, v) \notin E$.

Definition (Restnetzwerk)

Das Restnetzwerk G_f zu einem Netzwerk $G = (V, E, s, t, c)$ ist:

Restnetzwerk

$n = |V|, m = |E|$

Wir nehmen o.B.d.A bei einem Netzwerk $G = (V, E, s, t, c)$ an:

- $\forall e = (v, w) \in E : f(v, w) \cdot f(w, v) = 0$.
- Alternativ o.E.d.A: $(v, w) \in E \Longrightarrow (w, v) \notin E$.

Definition (Restnetzwerk)

Das Restnetzwerk G_f zu einem Netzwerk $G = (V, E, s, t, c)$ ist:

- Für alle Paare $(v, w) \in V^2$:

$$\text{rest}_f(v, w) = \begin{cases} c(v, w) - f(v, w) & (v, w) \in E \\ f(w, v) & (w, v) \in E \\ 0 & \text{sonst} \end{cases}$$

Restnetzwerk

$n = |V|, m = |E|$

Wir nehmen o.B.d.A bei einem Netzwerk $G = (V, E, s, t, c)$ an:

- $\forall e = (v, w) \in E : f(v, w) \cdot f(w, v) = 0$.
- Alternativ o.E.d.A: $(v, w) \in E \Longrightarrow (w, v) \notin E$.

Definition (Restnetzwerk)

Das Restnetzwerk G_f zu einem Netzwerk $G = (V, E, s, t, c)$ ist:

- Für alle Paare $(v, w) \in V^2$:

$$\text{rest}_f(v, w) = \begin{cases} c(v, w) - f(v, w) & (v, w) \in E \\ f(w, v) & (w, v) \in E \\ 0 & \text{sonst} \end{cases}$$

- $G_f = (V, E_f)$ mit $E_f = \{(v, w) \in V^2 \mid \text{rest}_f(v, w) > 0\}$

Restnetzwerk

$n = |V|, m = |E|$

Wir nehmen o.B.d.A bei einem Netzwerk $G = (V, E, s, t, c)$ an:

- $\forall e = (v, w) \in E : f(v, w) \cdot f(w, v) = 0$.
- Alternativ o.E.d.A: $(v, w) \in E \Longrightarrow (w, v) \notin E$.

Definition (Restnetzwerk)

Das Restnetzwerk G_f zu einem Netzwerk $G = (V, E, s, t, c)$ ist:

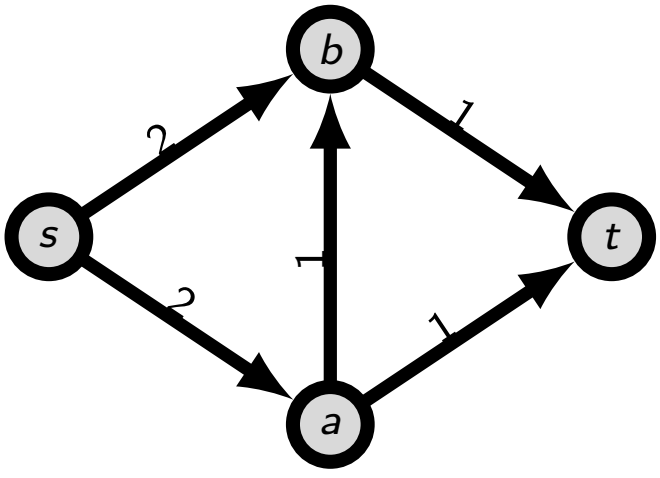
- Für alle Paare $(v, w) \in V^2$:

$$\text{rest}_f(v, w) = \begin{cases} c(v, w) - f(v, w) & (v, w) \in E \\ f(w, v) & (w, v) \in E \\ 0 & \text{sonst} \end{cases}$$

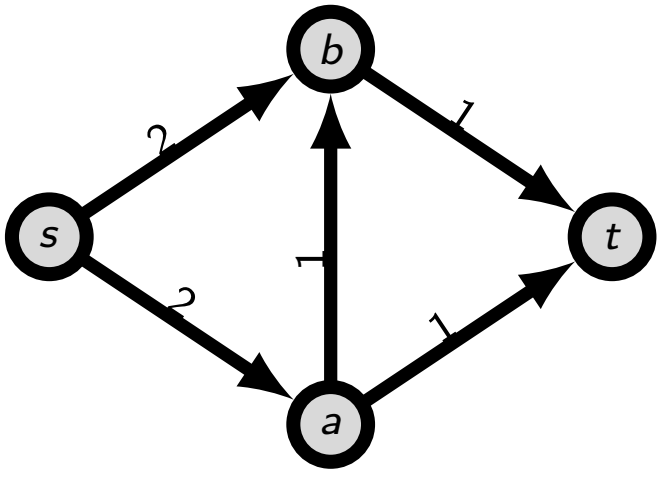
- $G_f = (V, E_f)$ mit $E_f = \{(v, w) \in V^2 \mid \text{rest}_f(v, w) > 0\}$

Einfaches Beispiel für Restnetzwerk

$n = |V|, m = |E|$



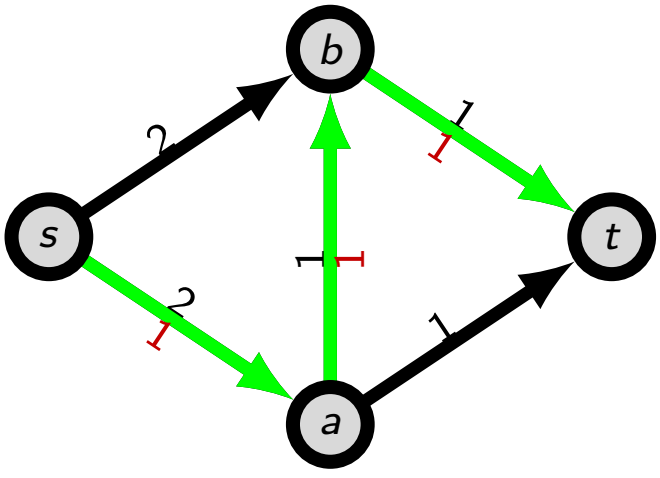
Das Restnetzwerk



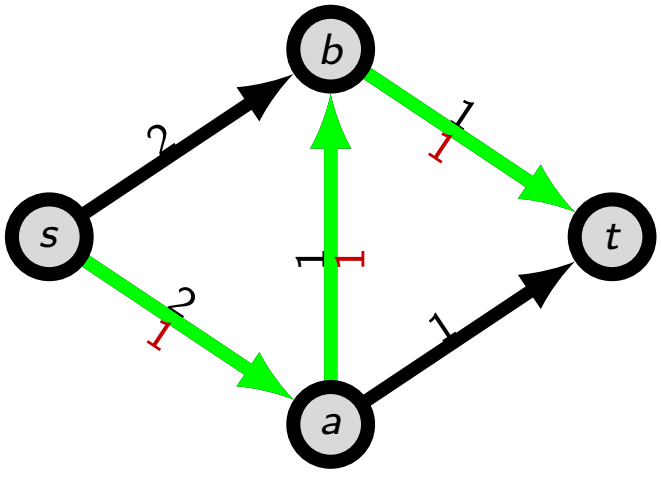
Das Netzwerk

Einfaches Beispiel für Restnetzwerk

$n = |V|, m = |E|$



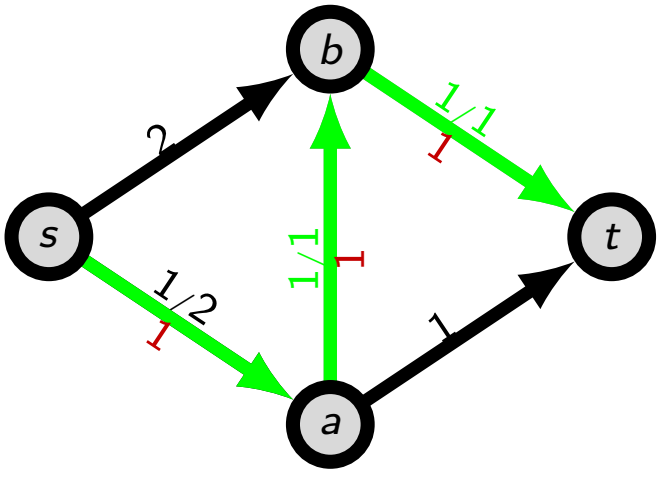
Das Restnetzwerk



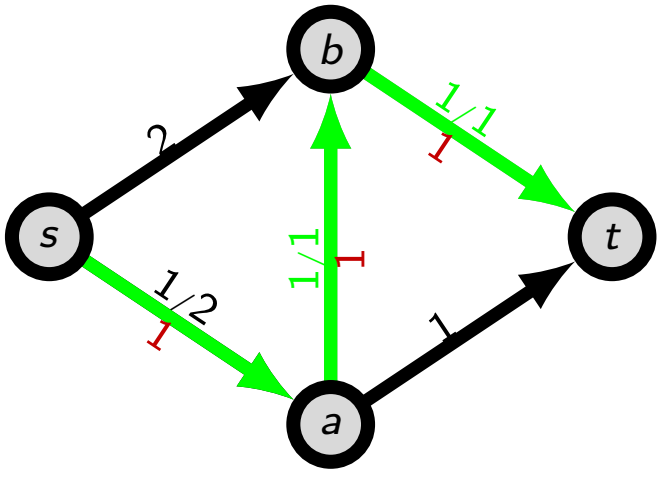
Das Netzwerk

Einfaches Beispiel für Restnetzwerk

$n = |V|, m = |E|$



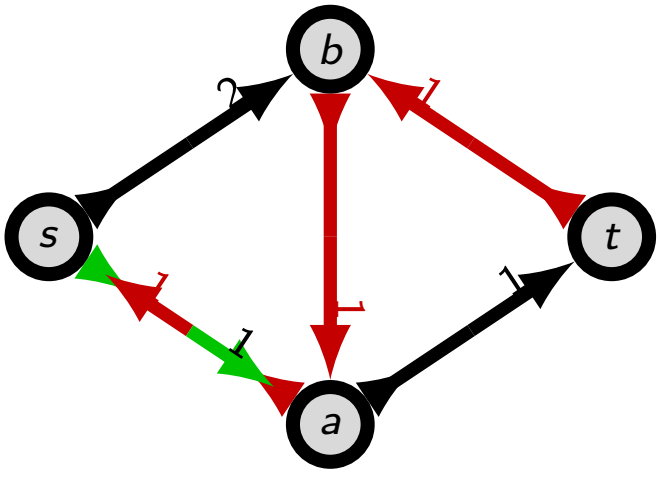
Das Restnetzwerk



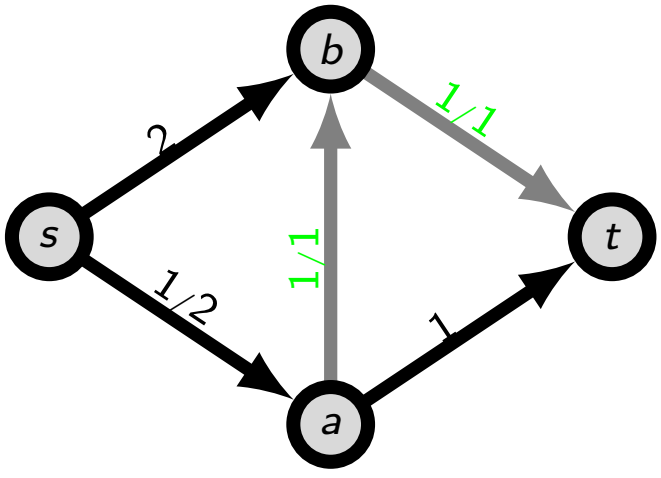
Das Netzwerk

Einfaches Beispiel für Restnetzwerk

$n = |V|, m = |E|$



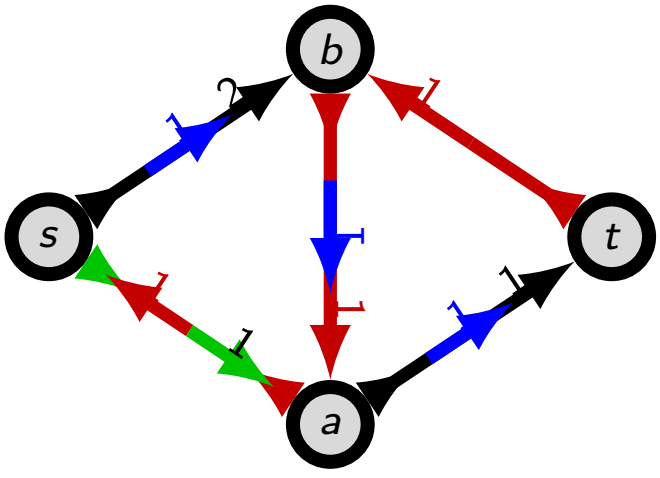
Das Restnetzwerk



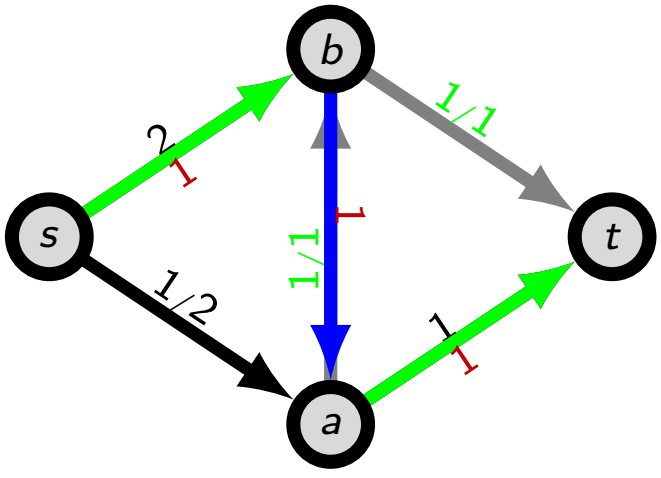
Das Netzwerk

Einfaches Beispiel für Restnetzwerk

$n = |V|, m = |E|$



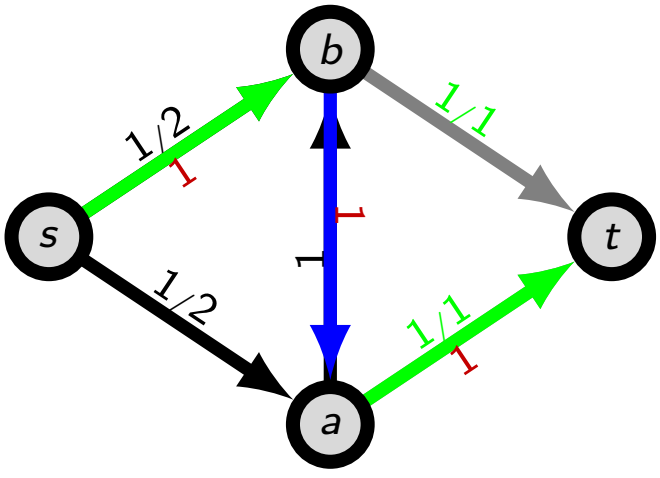
Das Restnetzwerk



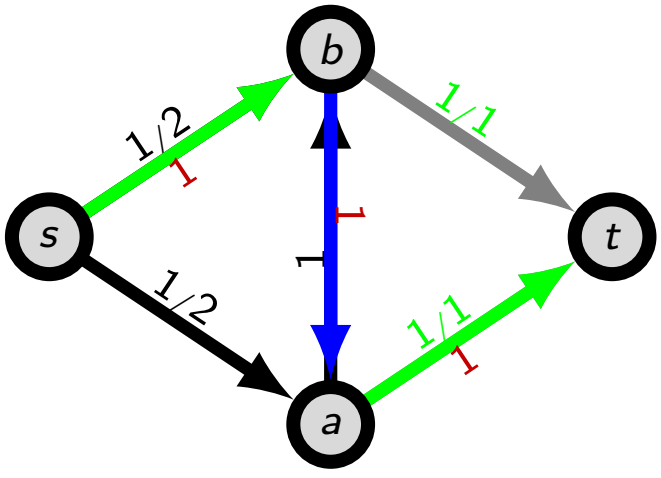
Das Netzwerk

Einfaches Beispiel für Restnetzwerk

$n = |V|, m = |E|$



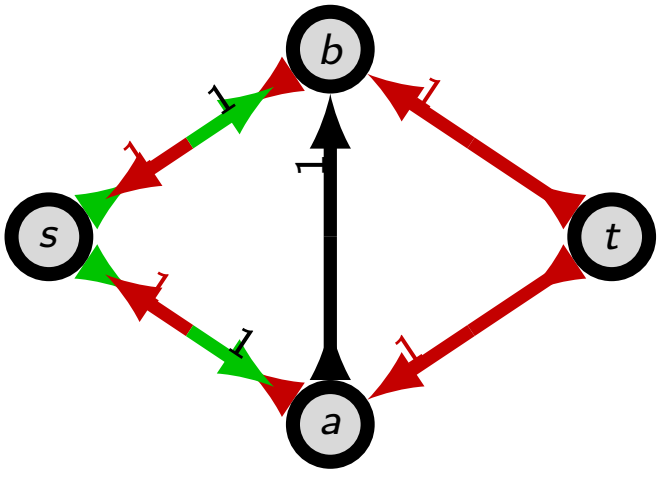
Das Restnetzwerk



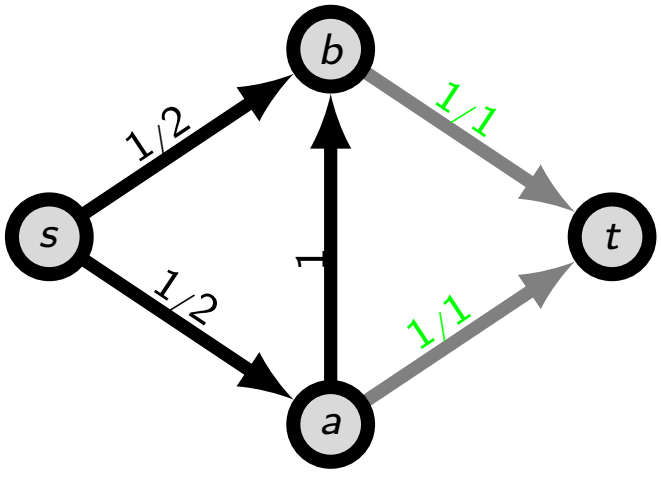
Das Netzwerk

Einfaches Beispiel für Restnetzwerk

$n = |V|, m = |E|$



Das Restnetzwerk



Das Netzwerk

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:
   1. Bestimme Restnetzwerk G_f

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:
   1. Bestimme Restnetzwerk G_f
   2. Bestimme flussvergrößernden Weg P in G_f.

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:
   1. Bestimme Restnetzwerk G_f
   2. Bestimme flussvergrößernden Weg P in G_f.
   3. Falls es P nicht gibt, gebe f aus und breche ab.

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:
   1. Bestimme Restnetzwerk G_f
   2. Bestimme flussvergrößernden Weg P in G_f.
   3. Falls es P nicht gibt, gebe f aus und breche ab.
   4. Bestimme maximale Flussvergrößerung p auf P.

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:
   1. Bestimme Restnetzwerk G_f
   2. Bestimme flussvergrößernden Weg P in G_f.
   3. Falls es P nicht gibt, gebe f aus und breche ab.
   4. Bestimme maximale Flussvergrößerung p auf P.
   5. Vergrößere f auf dem Pfad P um p.

Ford-Fulkerson-Methode (1957)

$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:
   1. Bestimme Restnetzwerk G_f
   2. Bestimme flussvergrößernden Weg P in G_f.
   3. Falls es P nicht gibt, gebe f aus und breche ab.
   4. Bestimme maximale Flussvergrößerung p auf P.
   5. Vergrößere f auf dem Pfad P um p.

Ford-Fulkerson-Methode (1957)

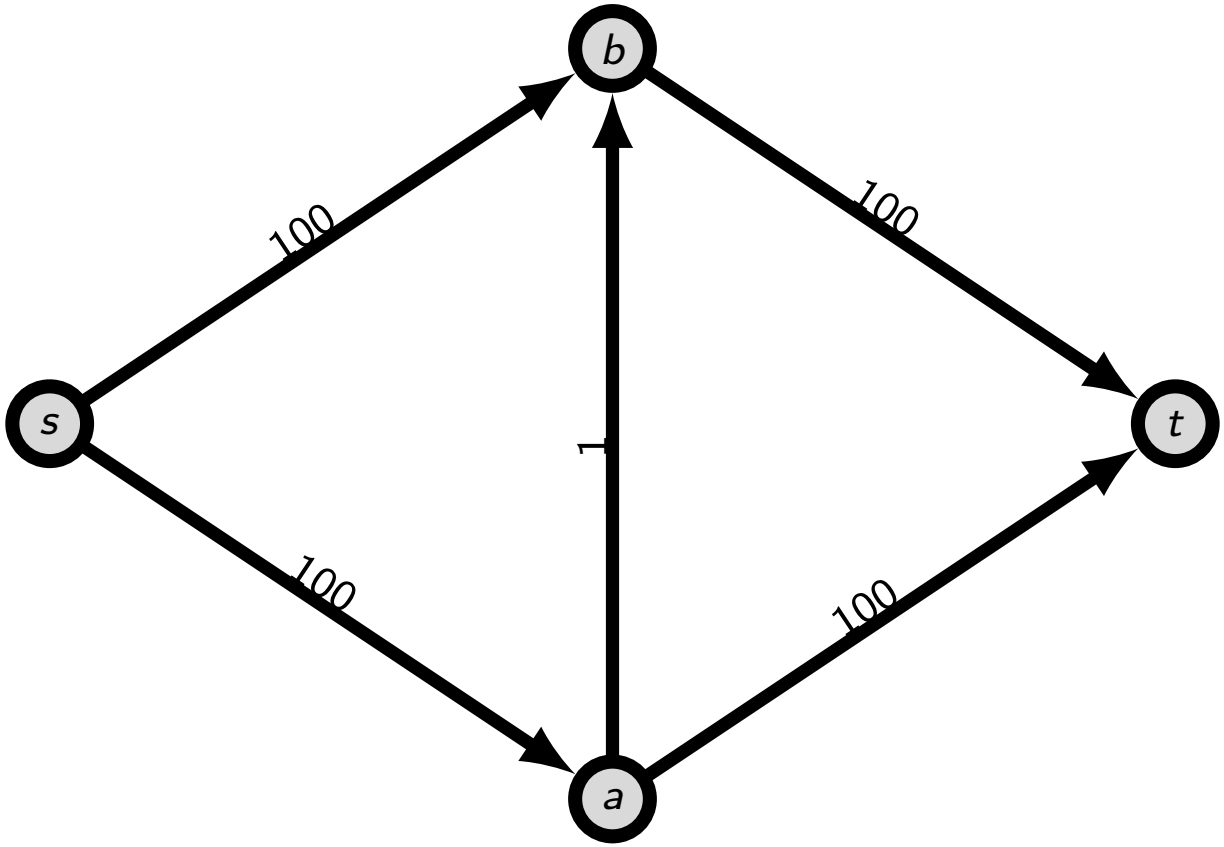
$n = |V|, m = |E|$

1. Gegeben ist Netzwerk: $G = (V, E, s, t, c)$
2. $\forall e \in E : f(e) = 0$.
3. Wiederhole:
   1. Bestimme Restnetzwerk G_f
   2. Bestimme flussvergrößernden Weg P in G_f.
   3. Falls es P nicht gibt, gebe f aus und breche ab.
   4. Bestimme maximale Flussvergrößerung p auf P.
   5. Vergrößere f auf dem Pfad P um p.

Laufzeit: $O(C \cdot m)$ mit $C = \sum_{e \in E} c(e)$.

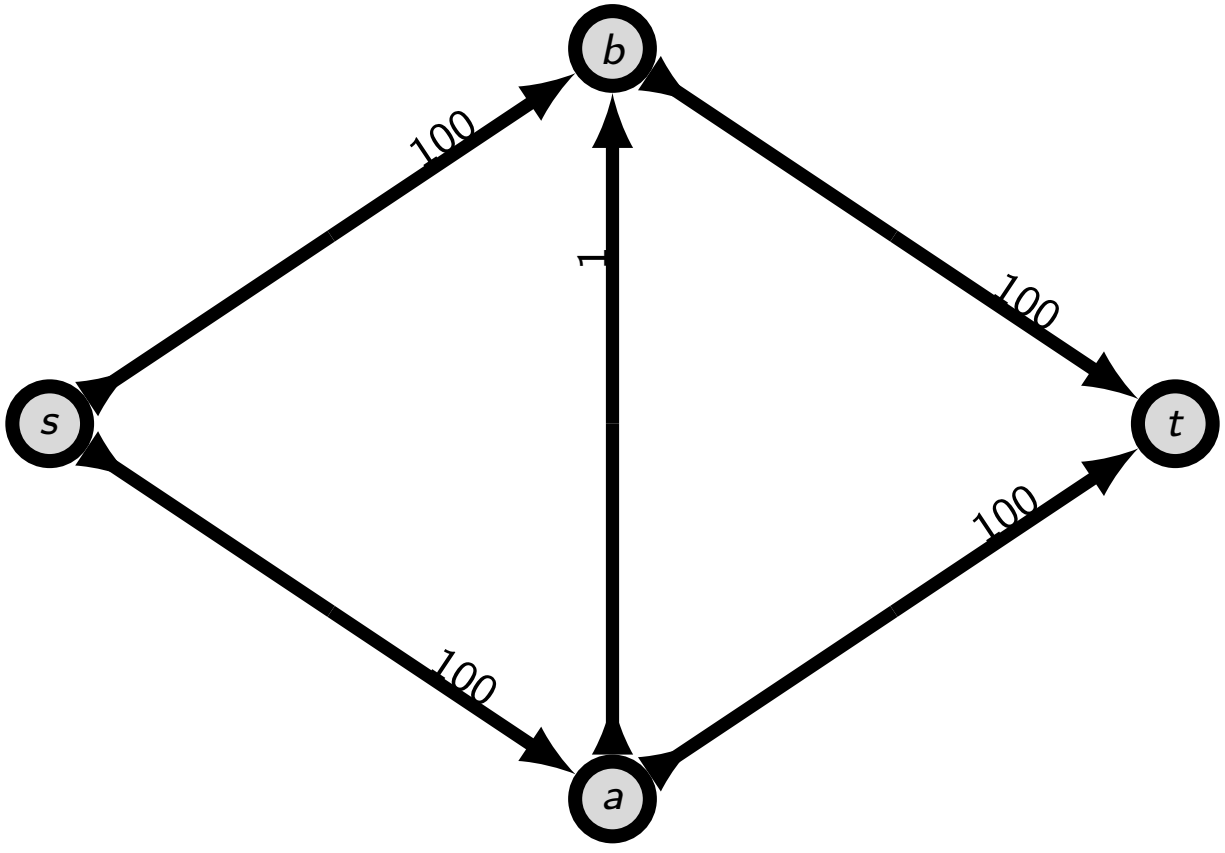
Beispiel

$n = |V|, m = |E|$



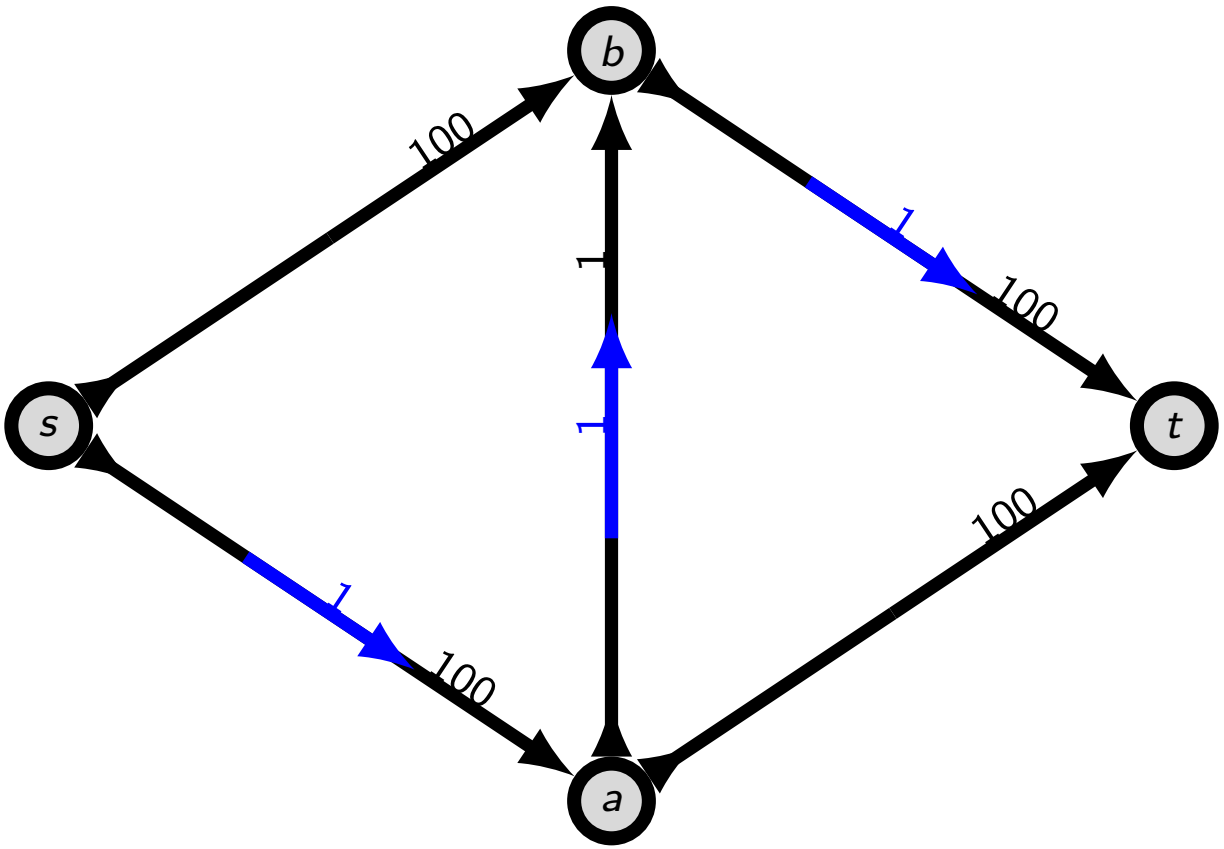
Beispiel

$n = |V|, m = |E|$



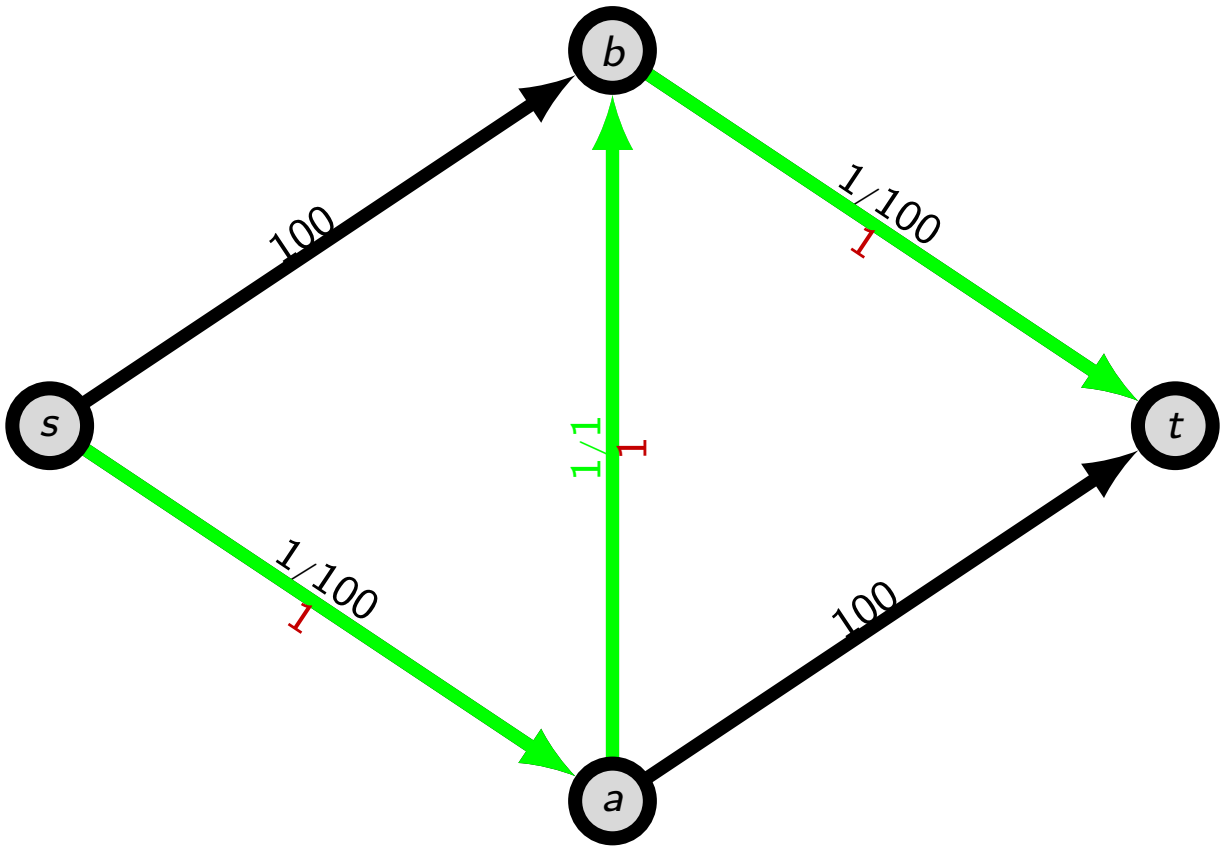
Beispiel

$n = |V|, m = |E|$



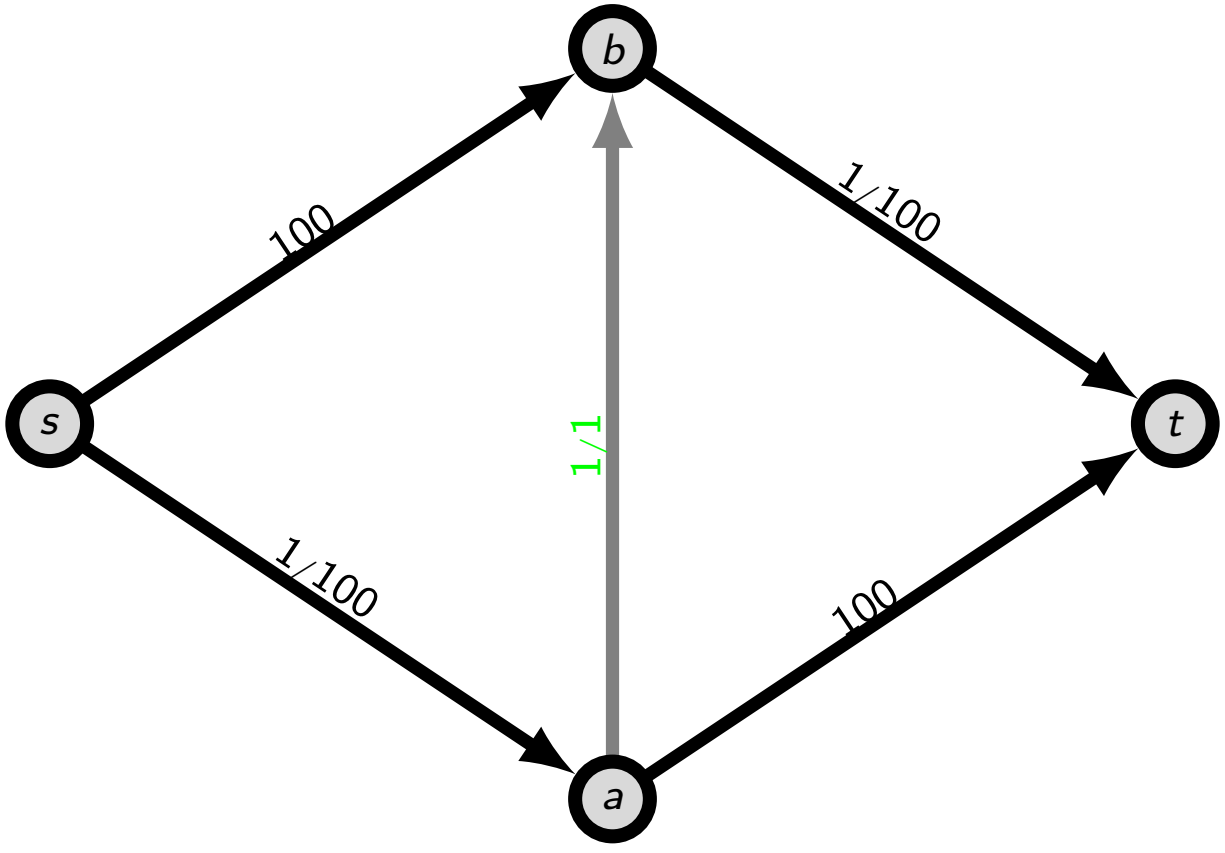
Beispiel

$n = |V|, m = |E|$



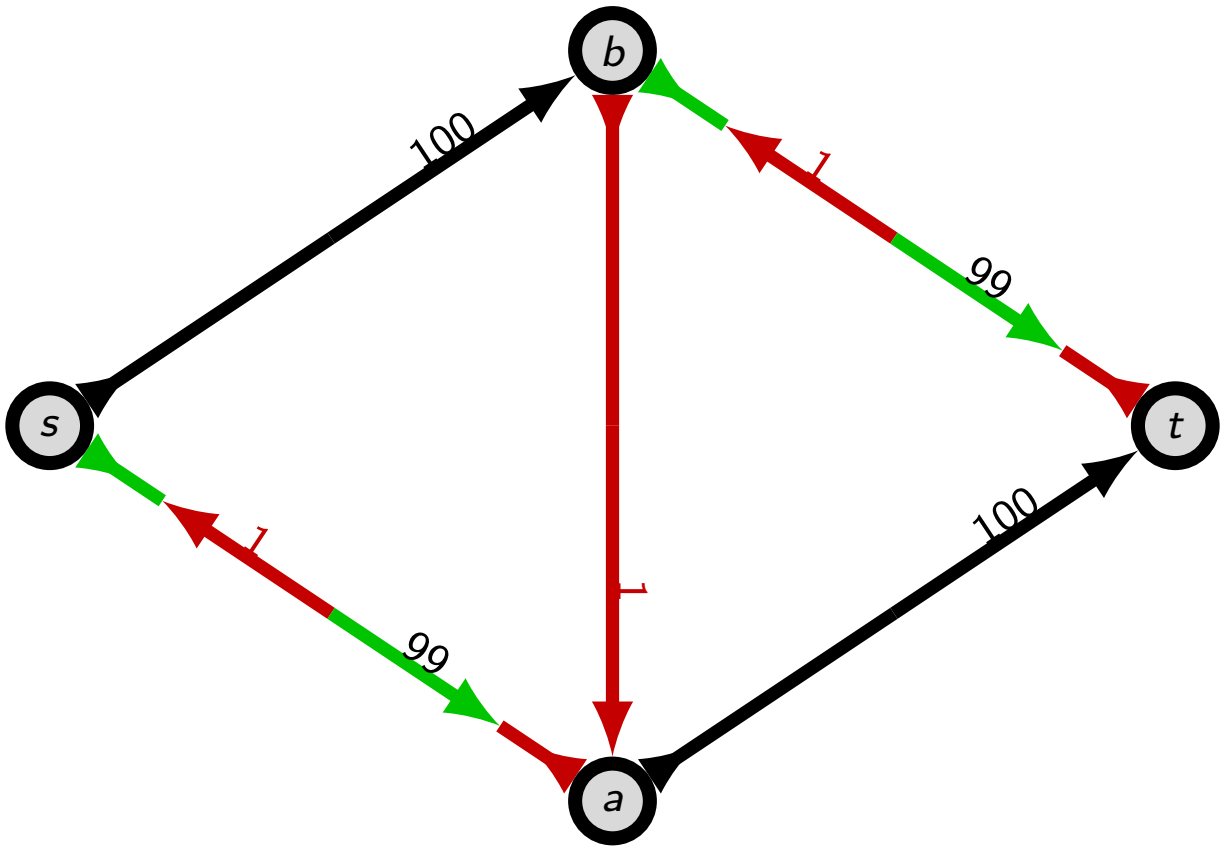
Beispiel

$n = |V|, m = |E|$



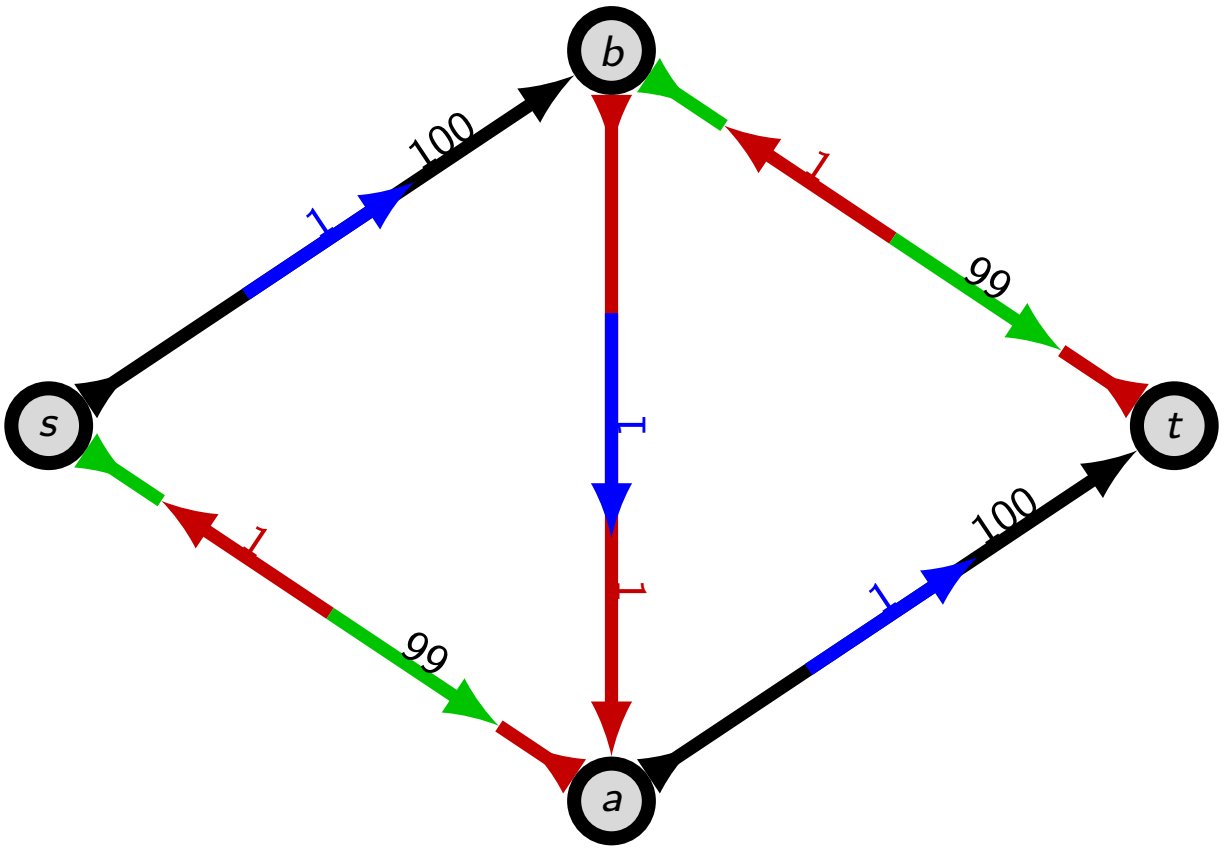
Beispiel

$n = |V|, m = |E|$



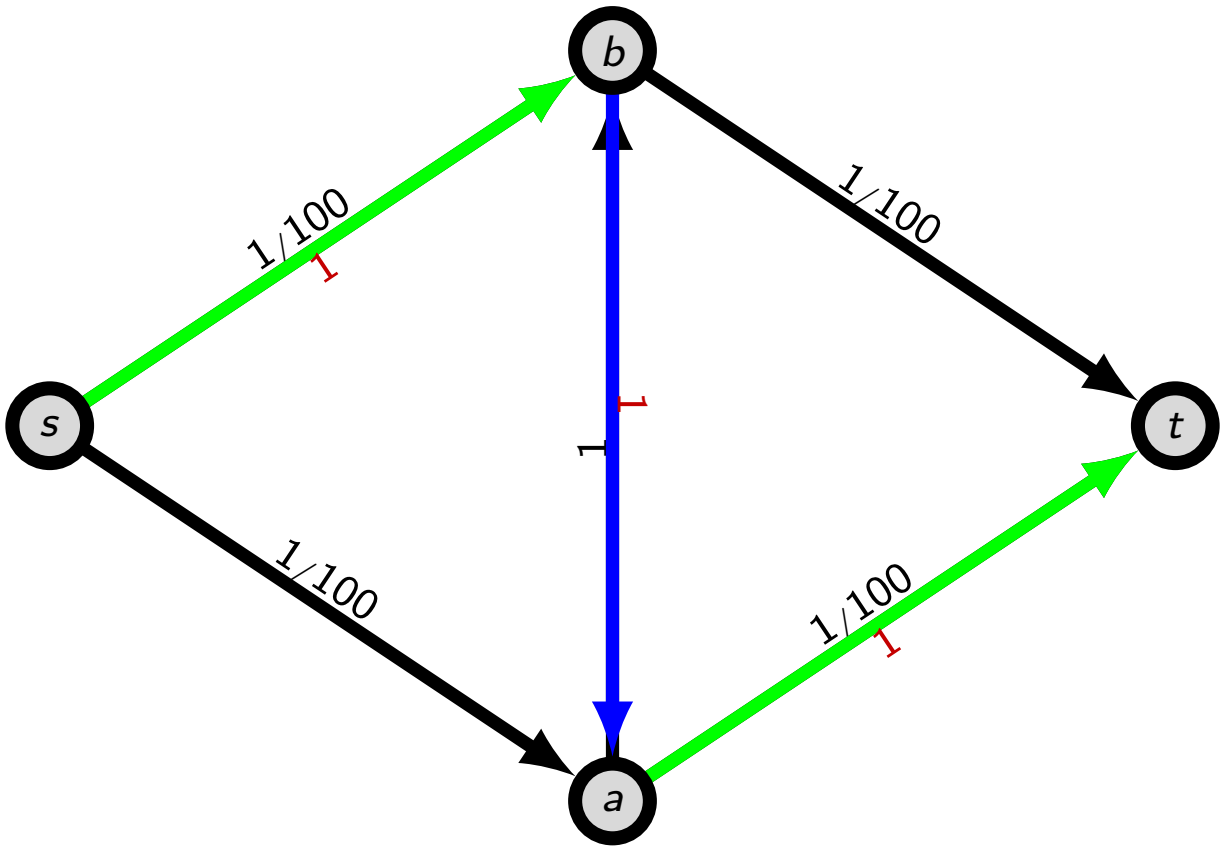
Beispiel

$n = |V|, m = |E|$



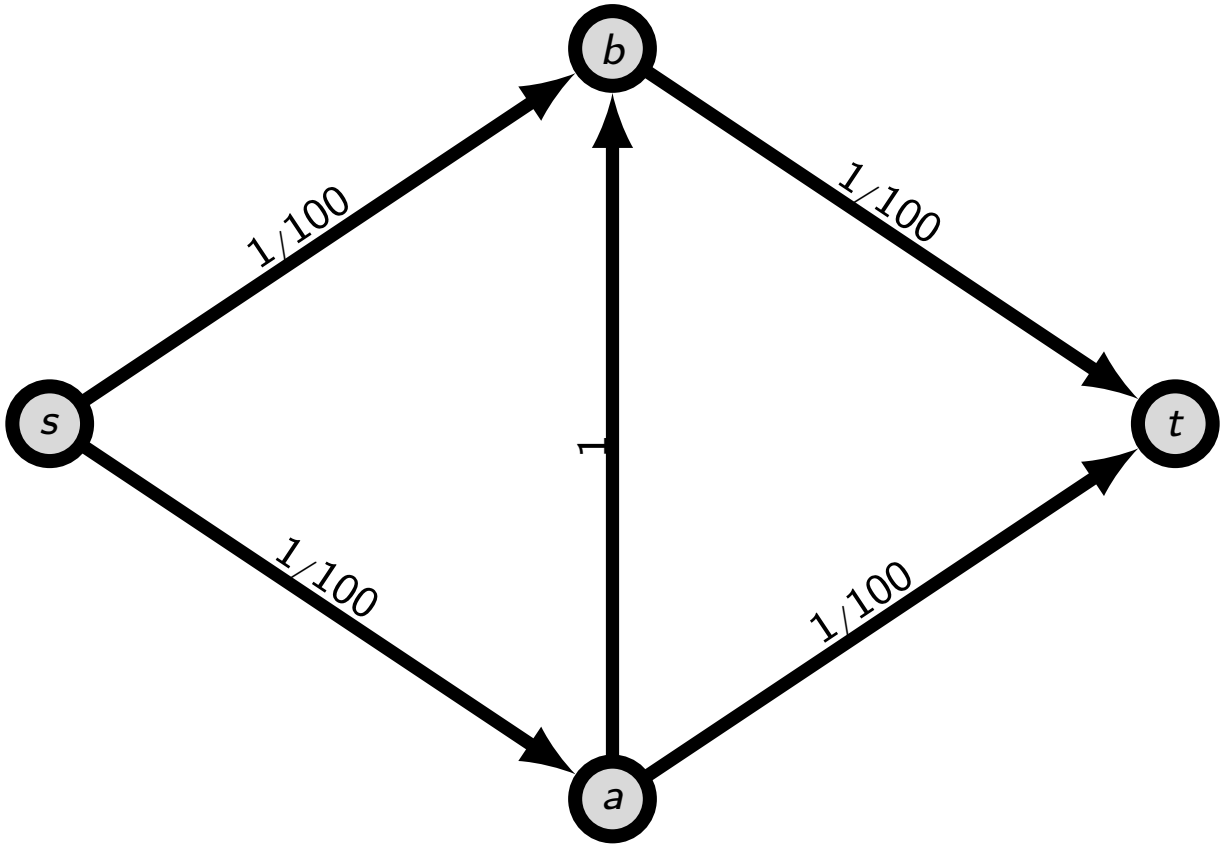
Beispiel

$n = |V|, m = |E|$



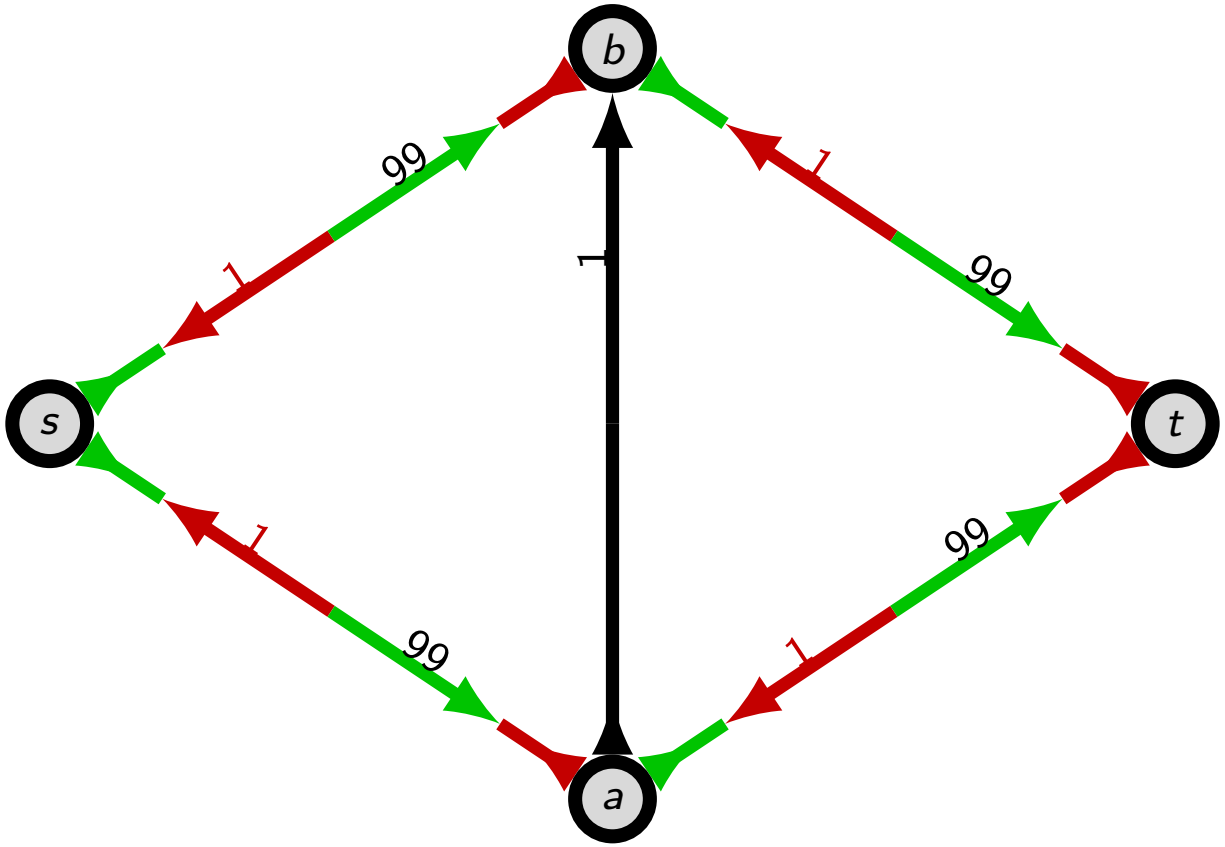
Beispiel

$n = |V|, m = |E|$



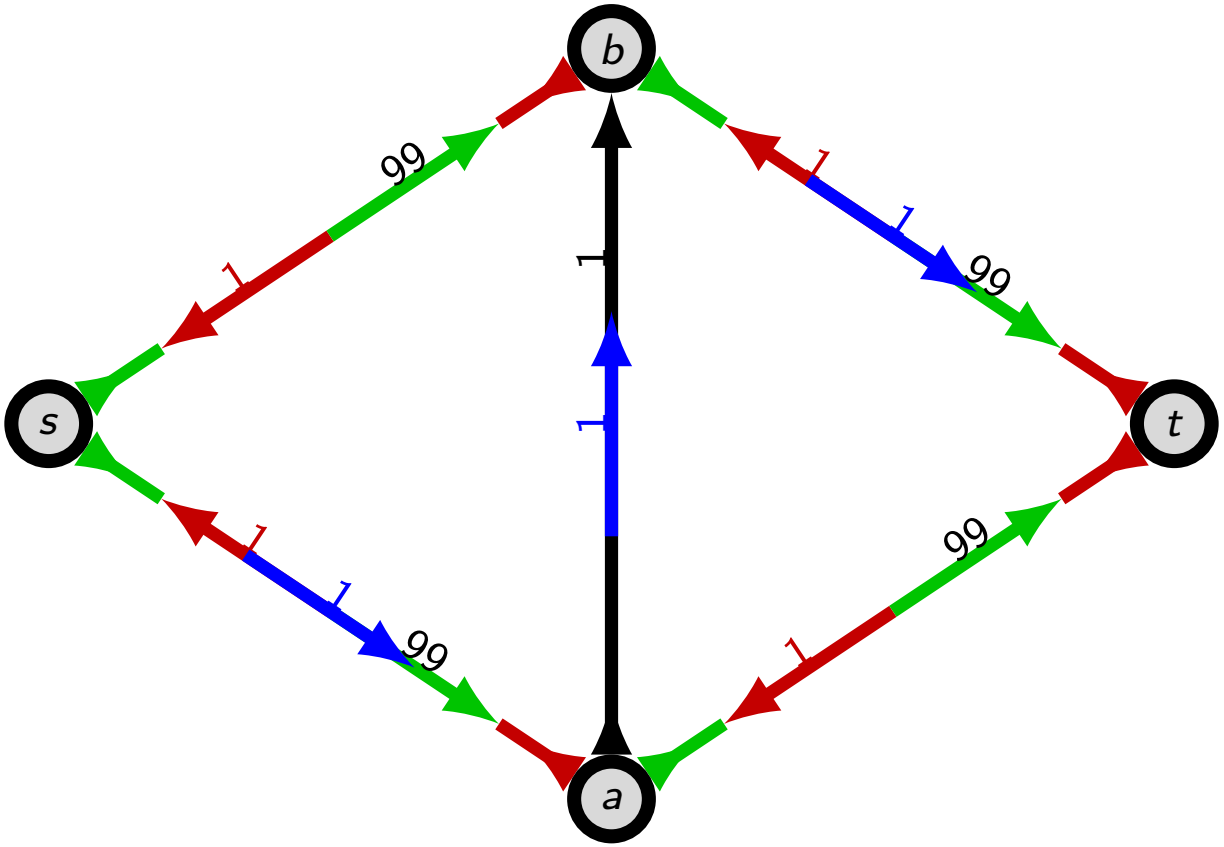
Beispiel

$n = |V|, m = |E|$



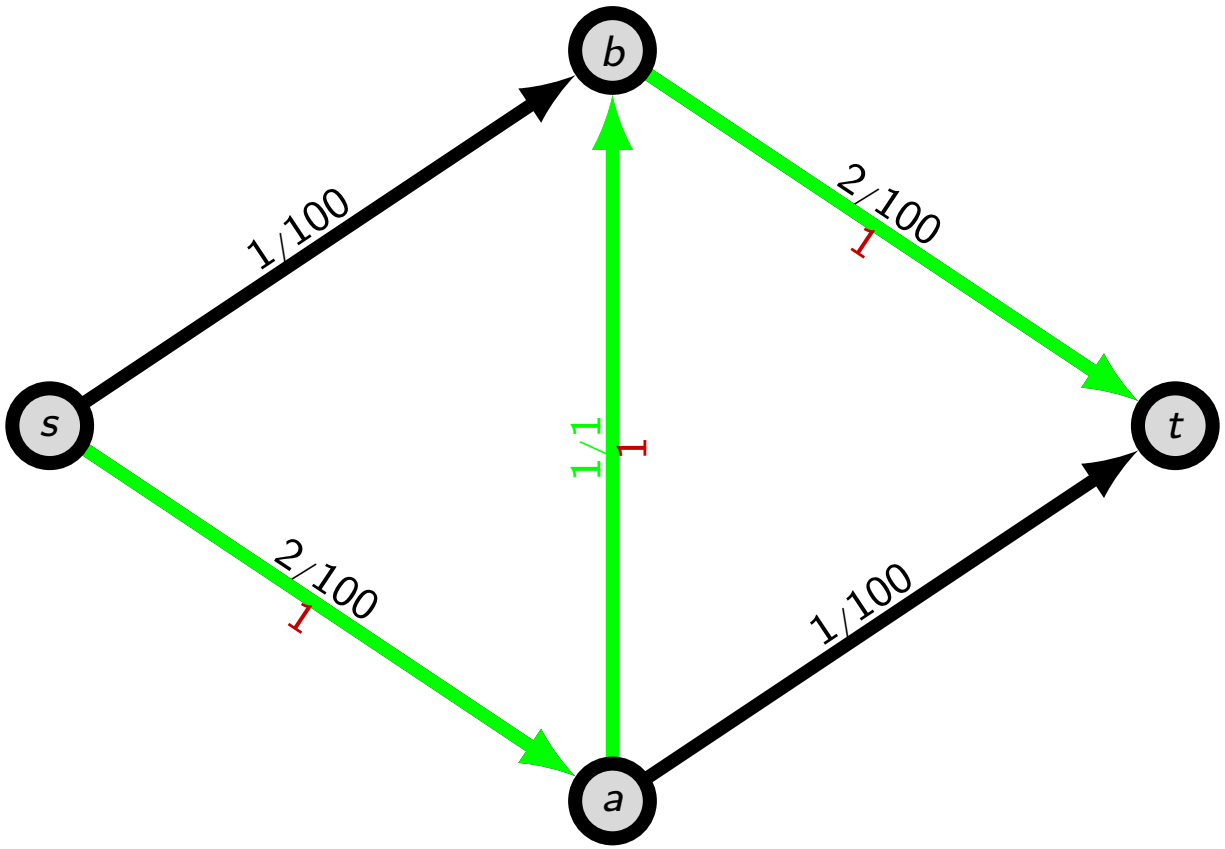
Beispiel

$n = |V|, m = |E|$



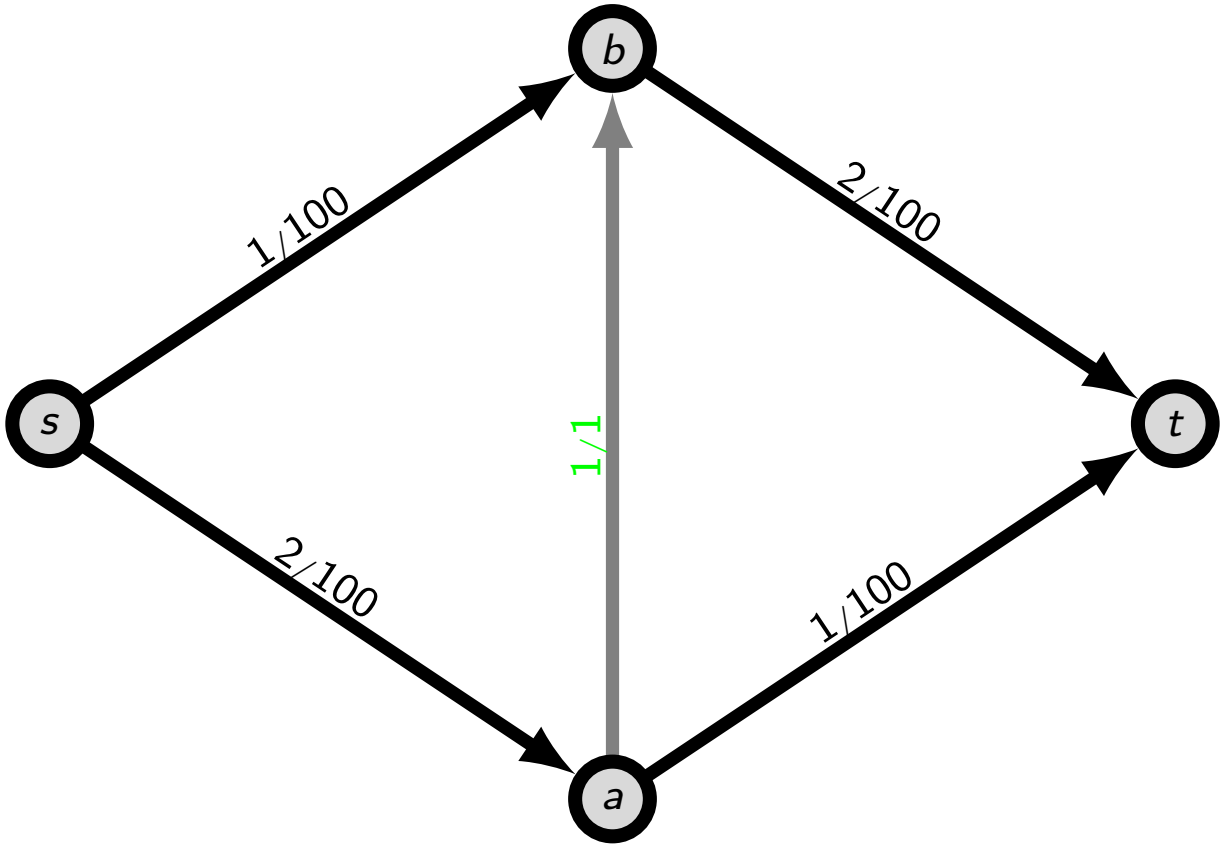
Beispiel

$n = |V|, m = |E|$



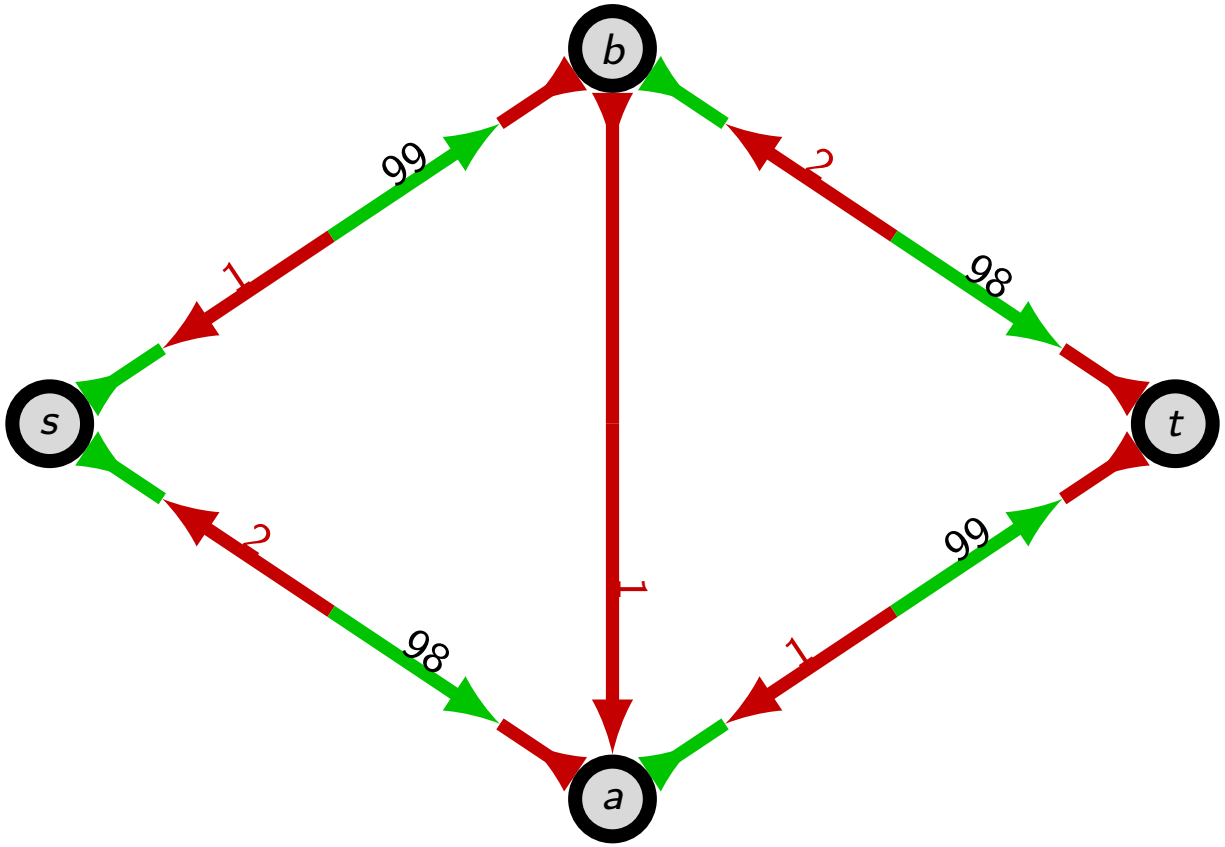
Beispiel

$n = |V|, m = |E|$



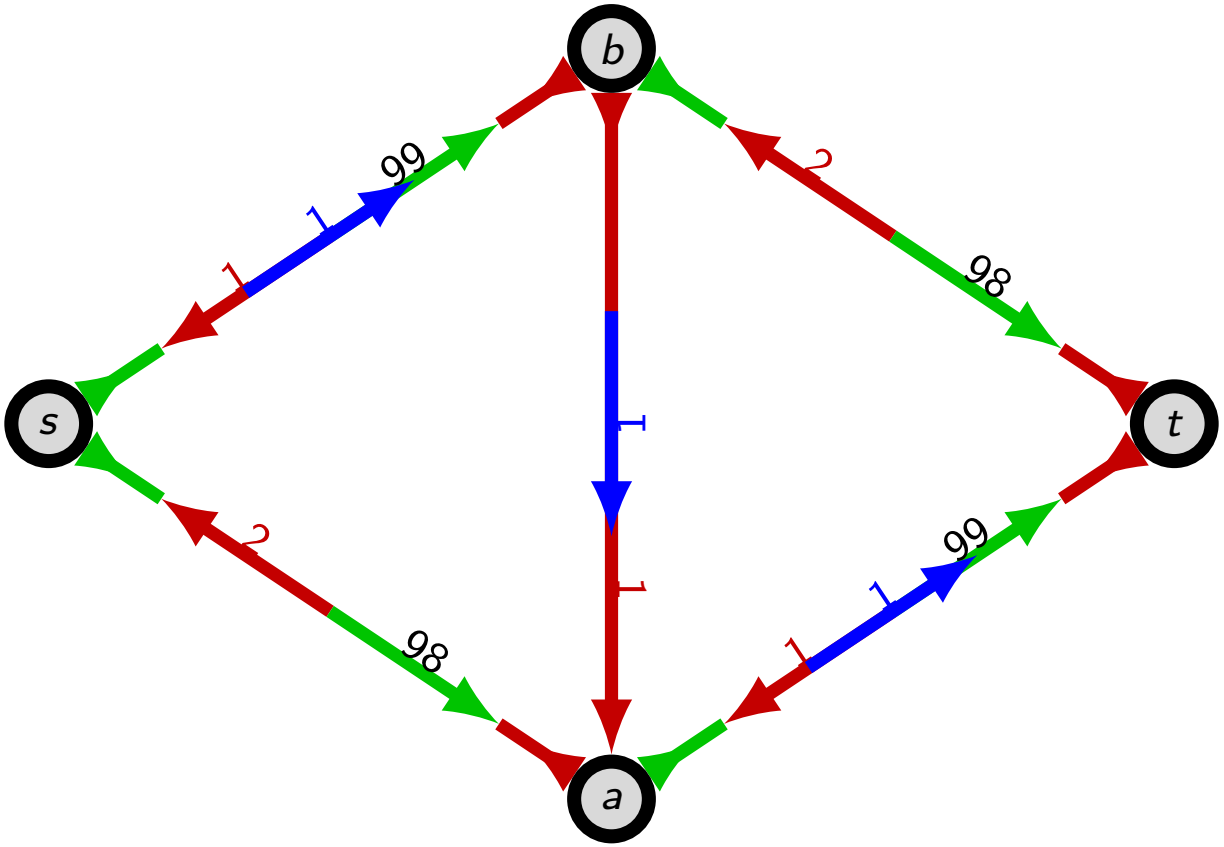
Beispiel

$n = |V|, m = |E|$



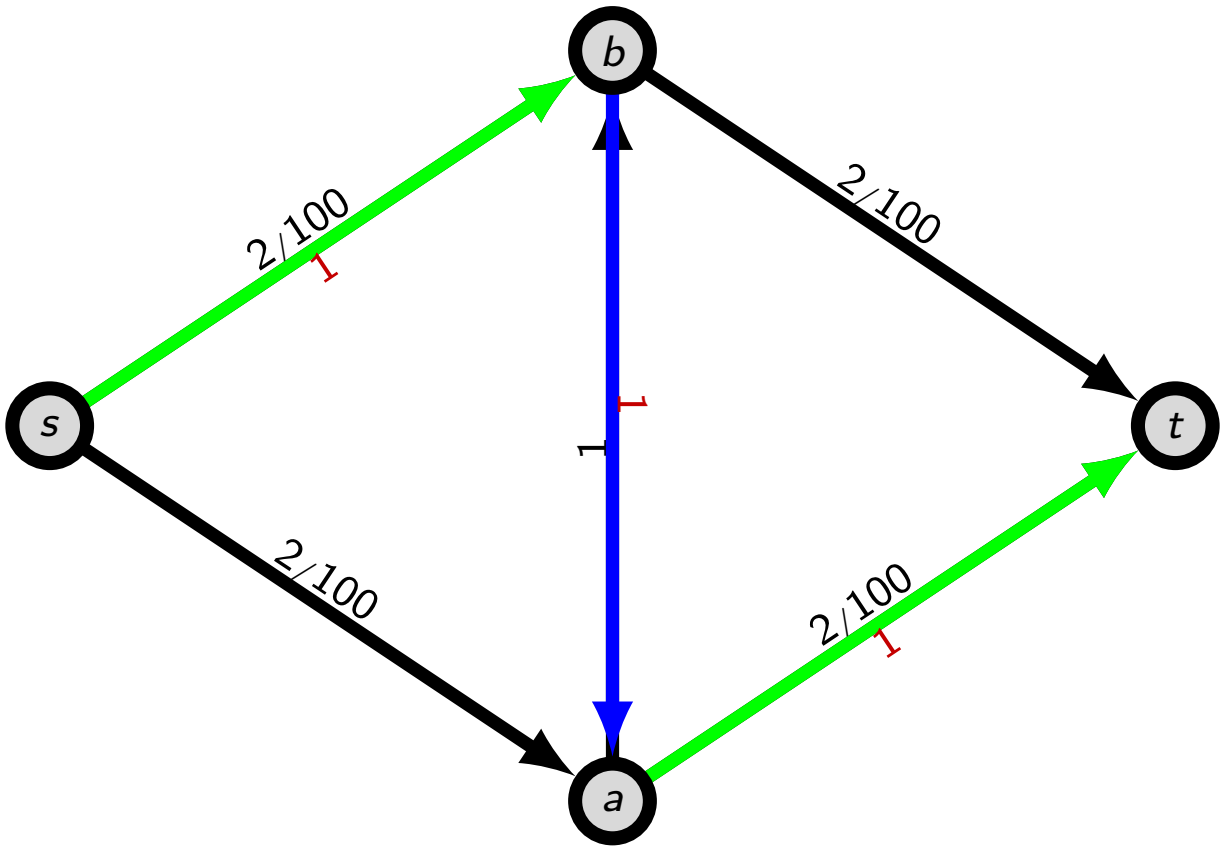
Beispiel

$n = |V|, m = |E|$



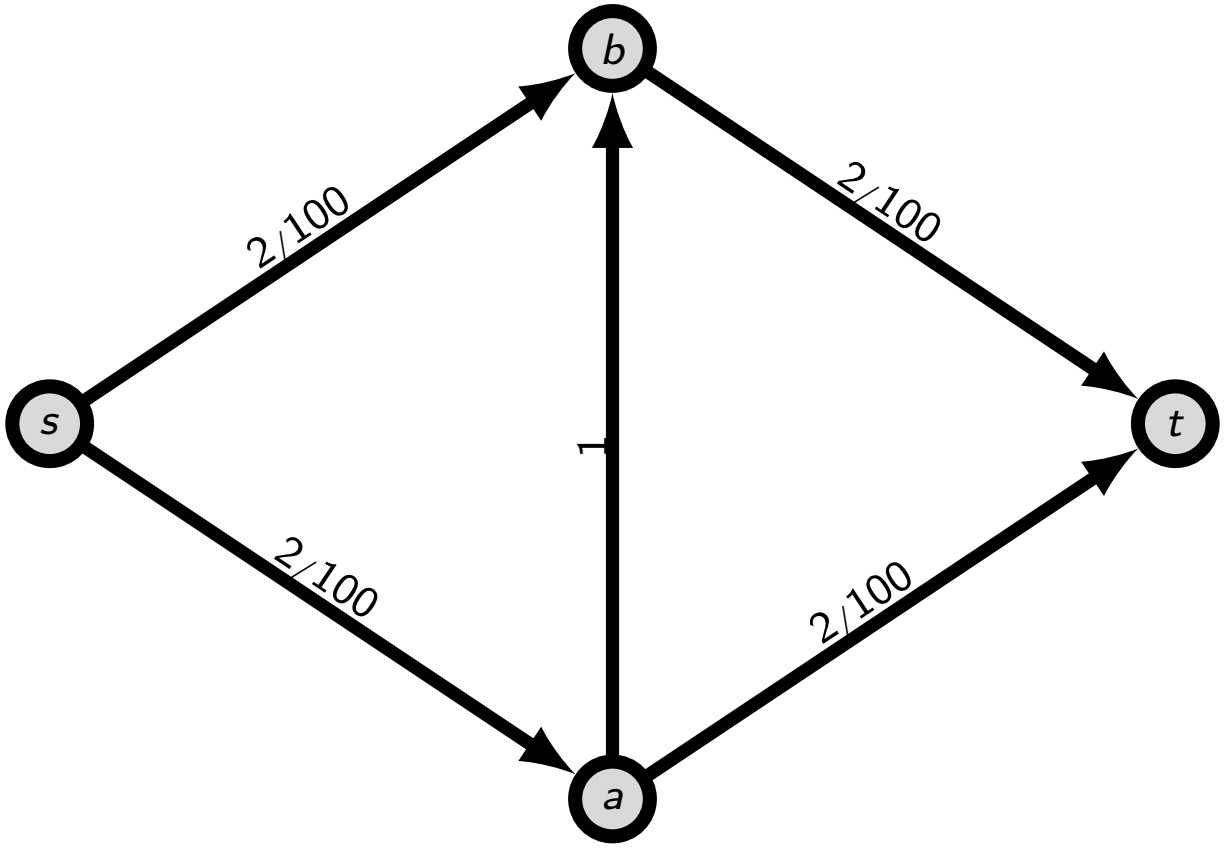
Beispiel

$n = |V|, m = |E|$



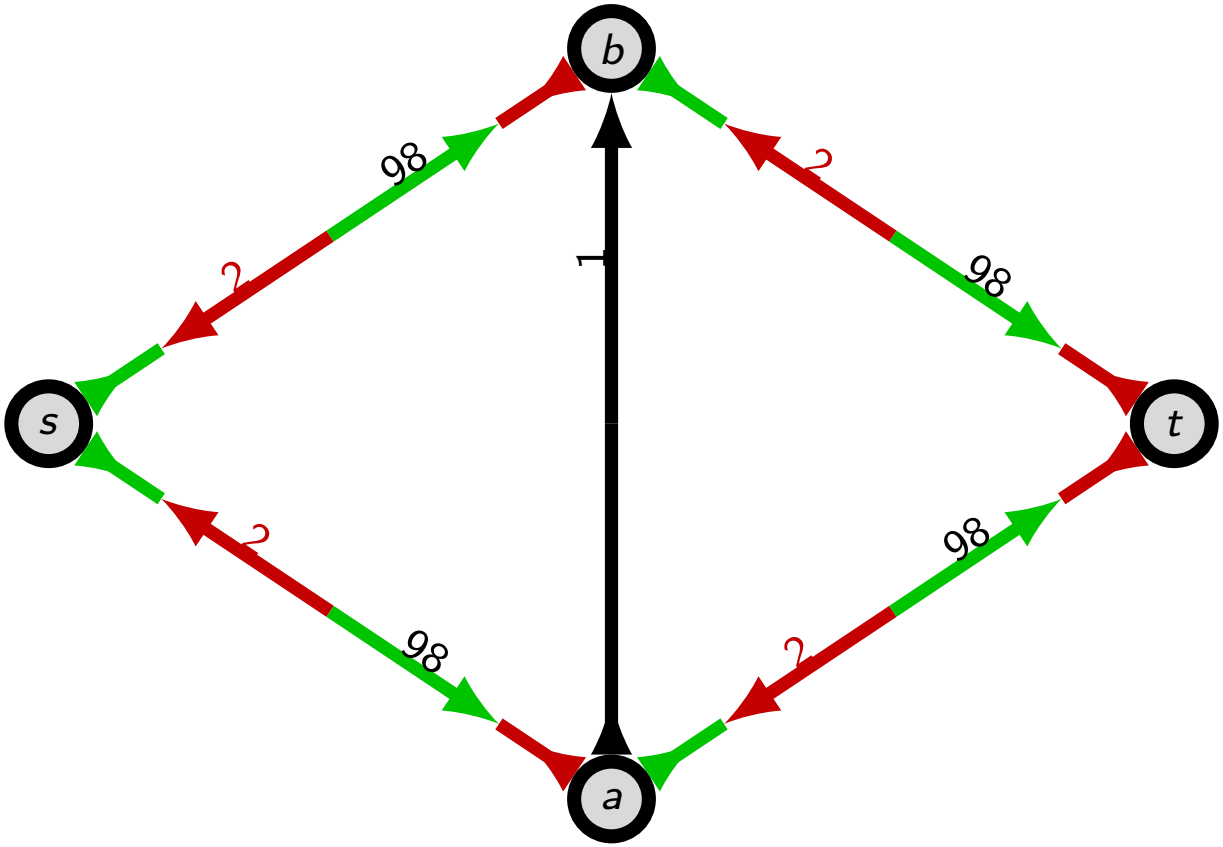
Beispiel

$n = |V|, m = |E|$



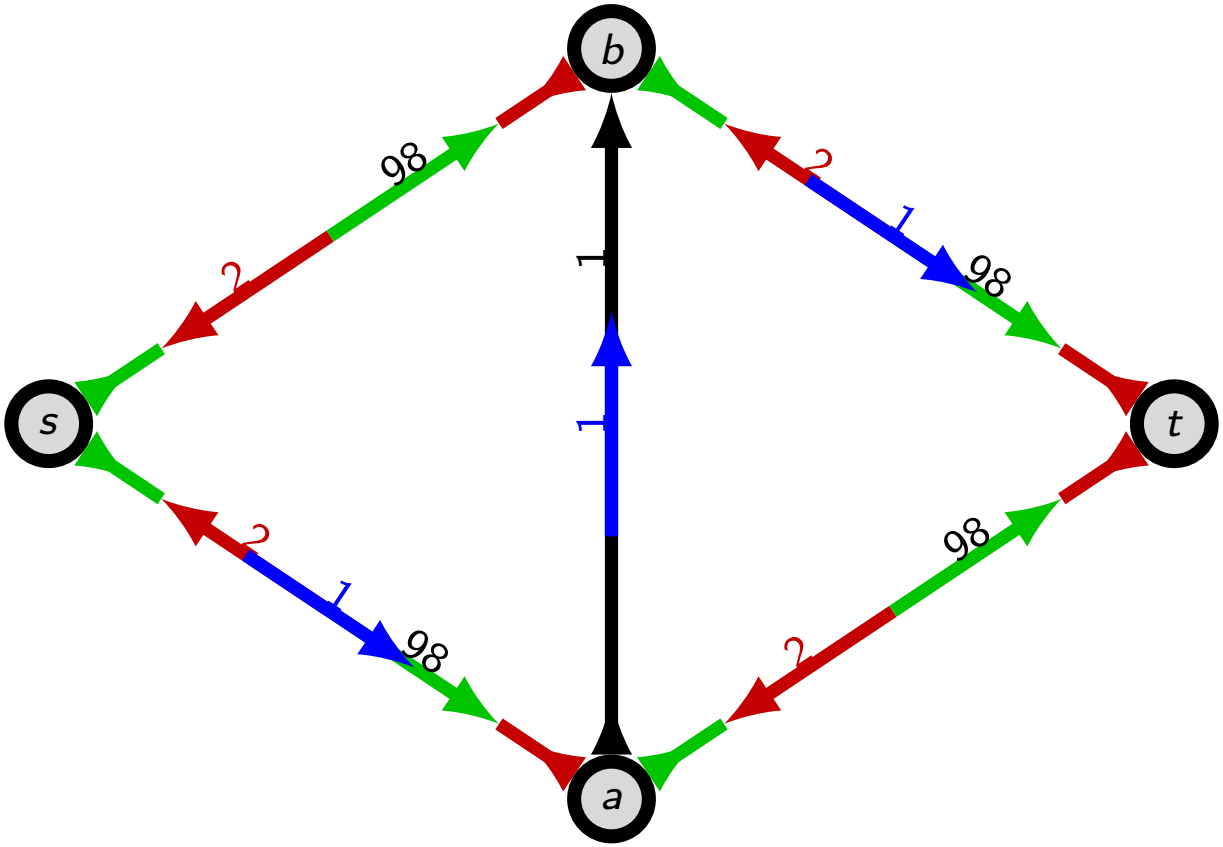
Beispiel

$n = |V|, m = |E|$



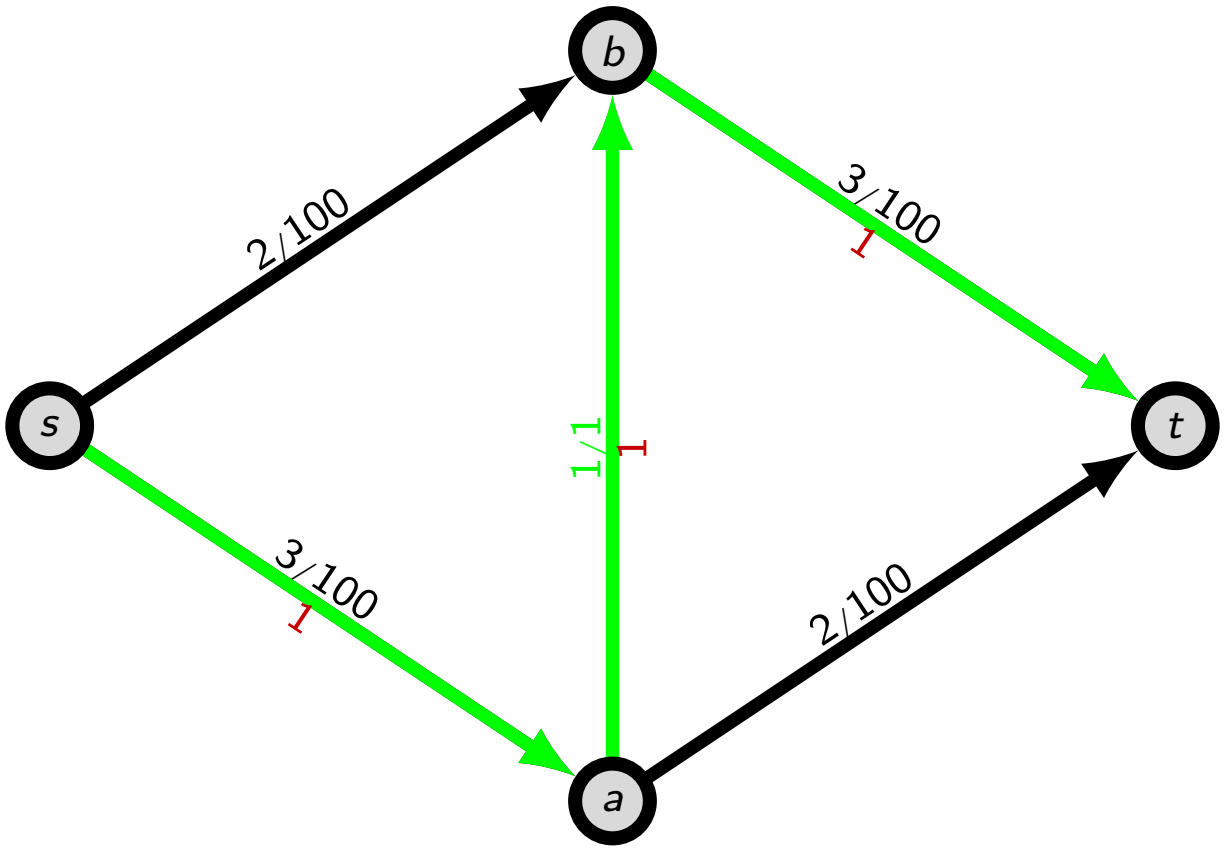
Beispiel

$n = |V|, m = |E|$



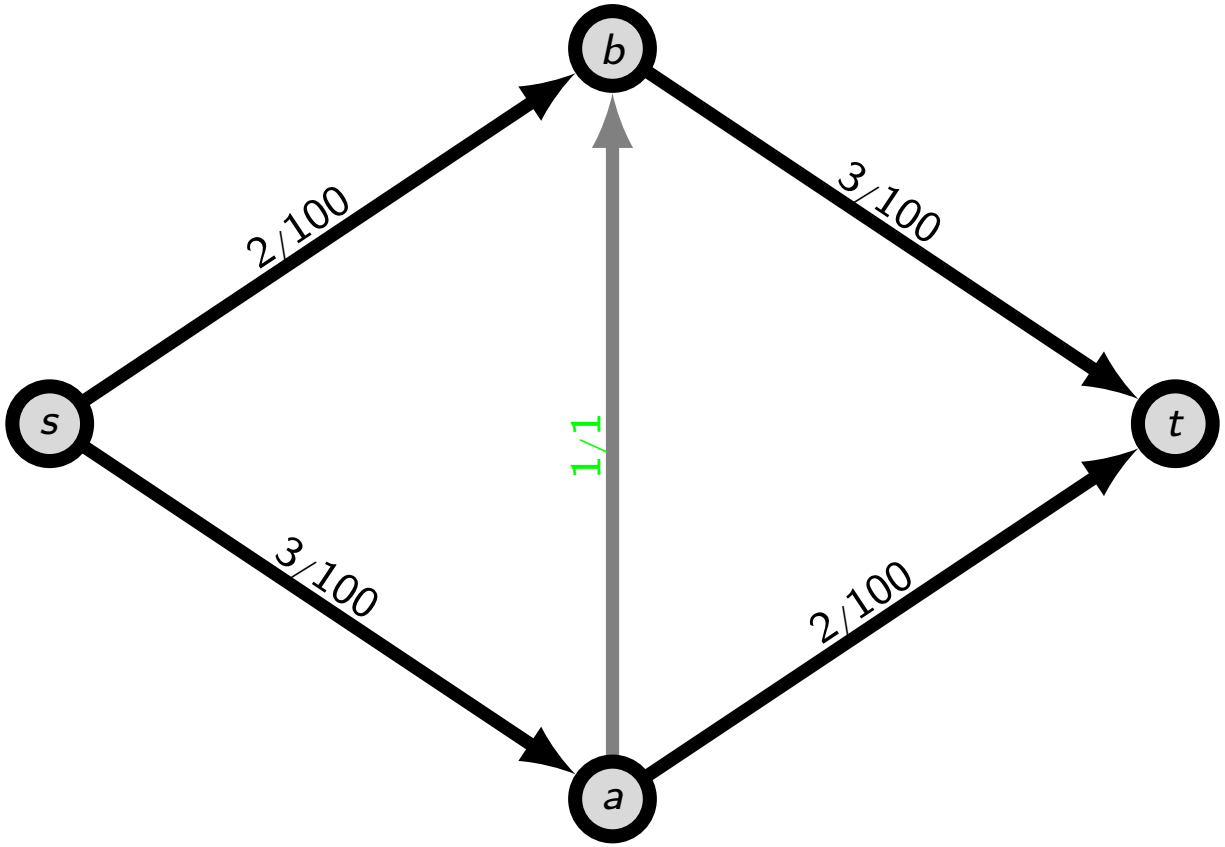
Beispiel

$n = |V|, m = |E|$



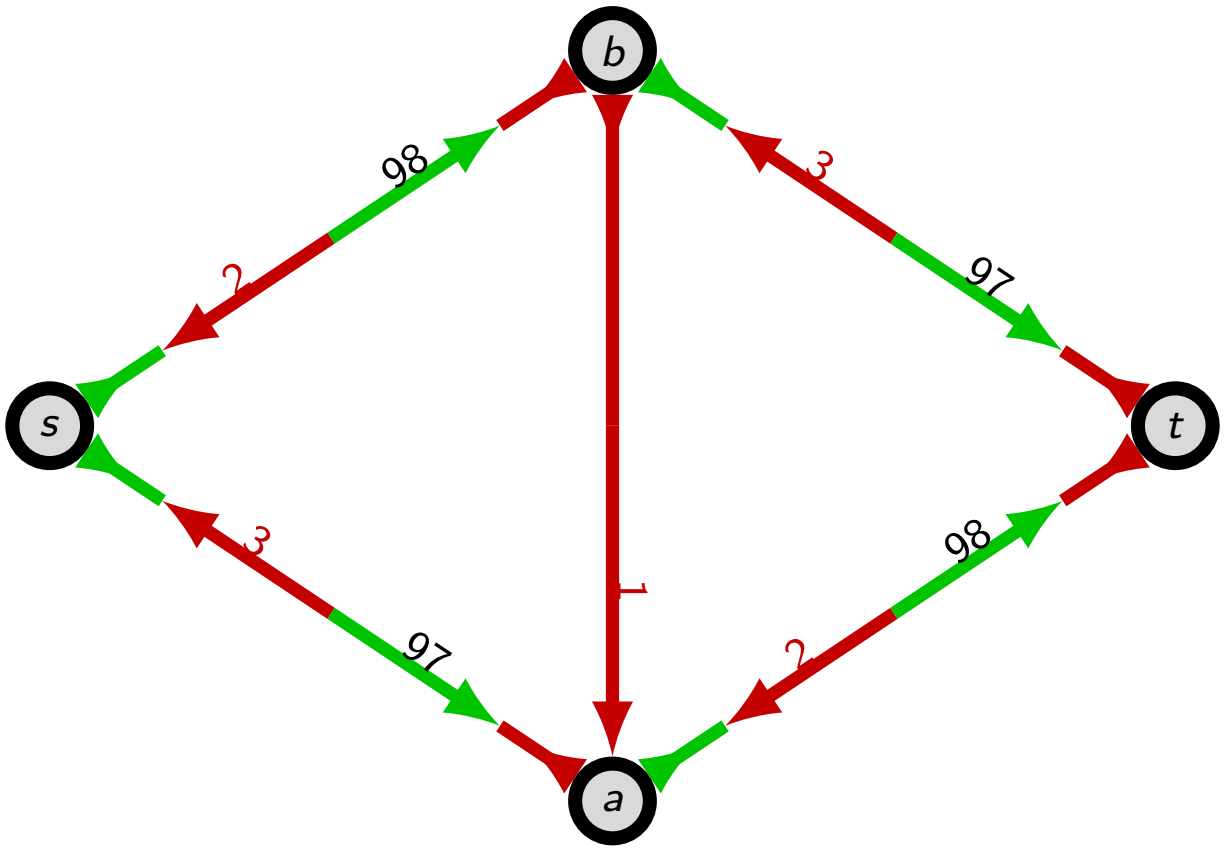
Beispiel

$n = |V|, m = |E|$



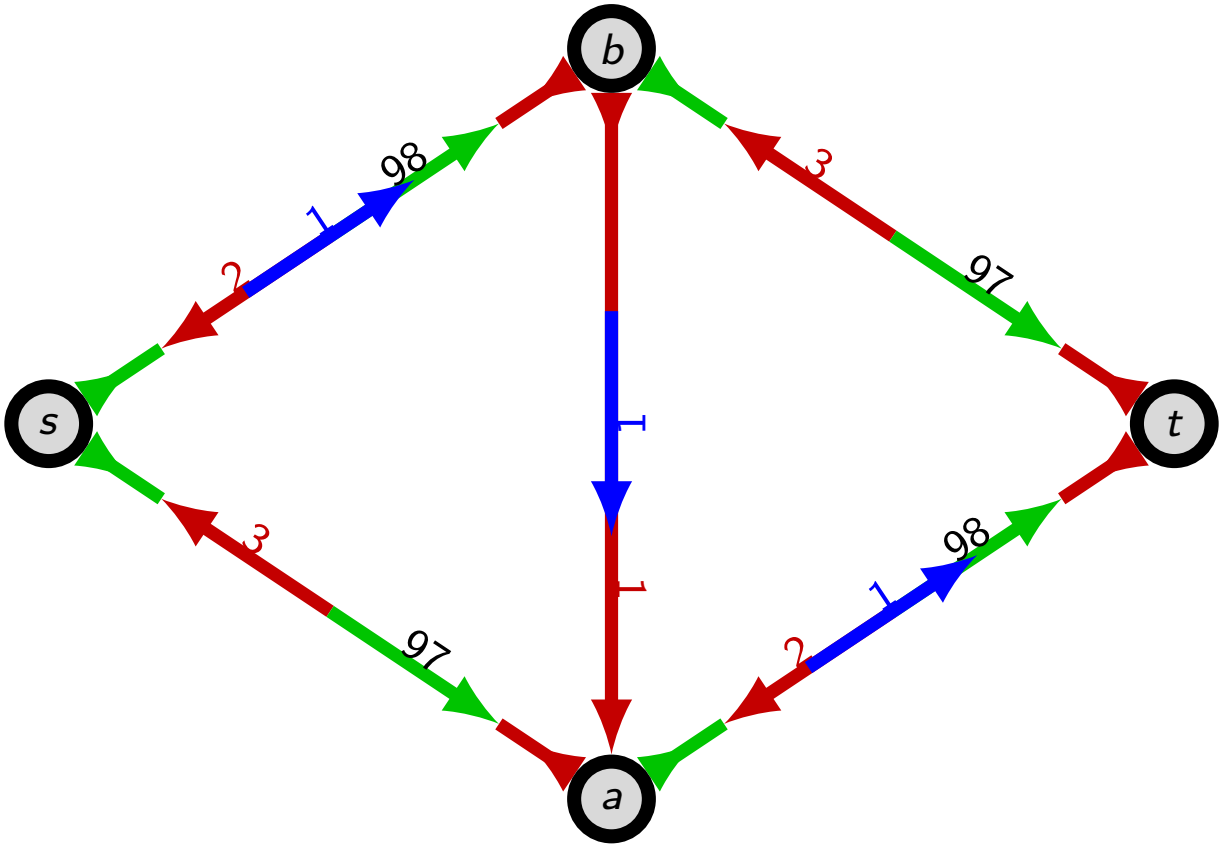
Beispiel

$n = |V|, m = |E|$



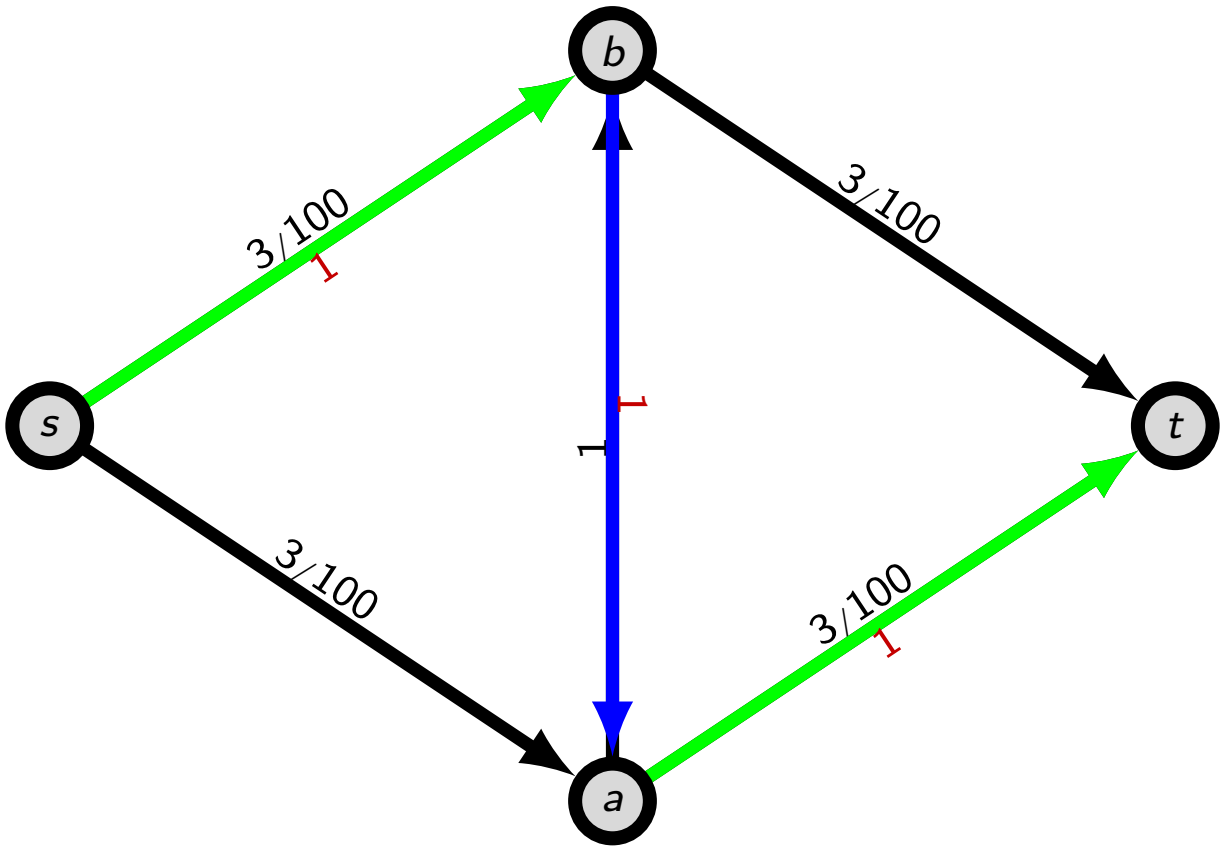
Beispiel

$n = |V|, m = |E|$



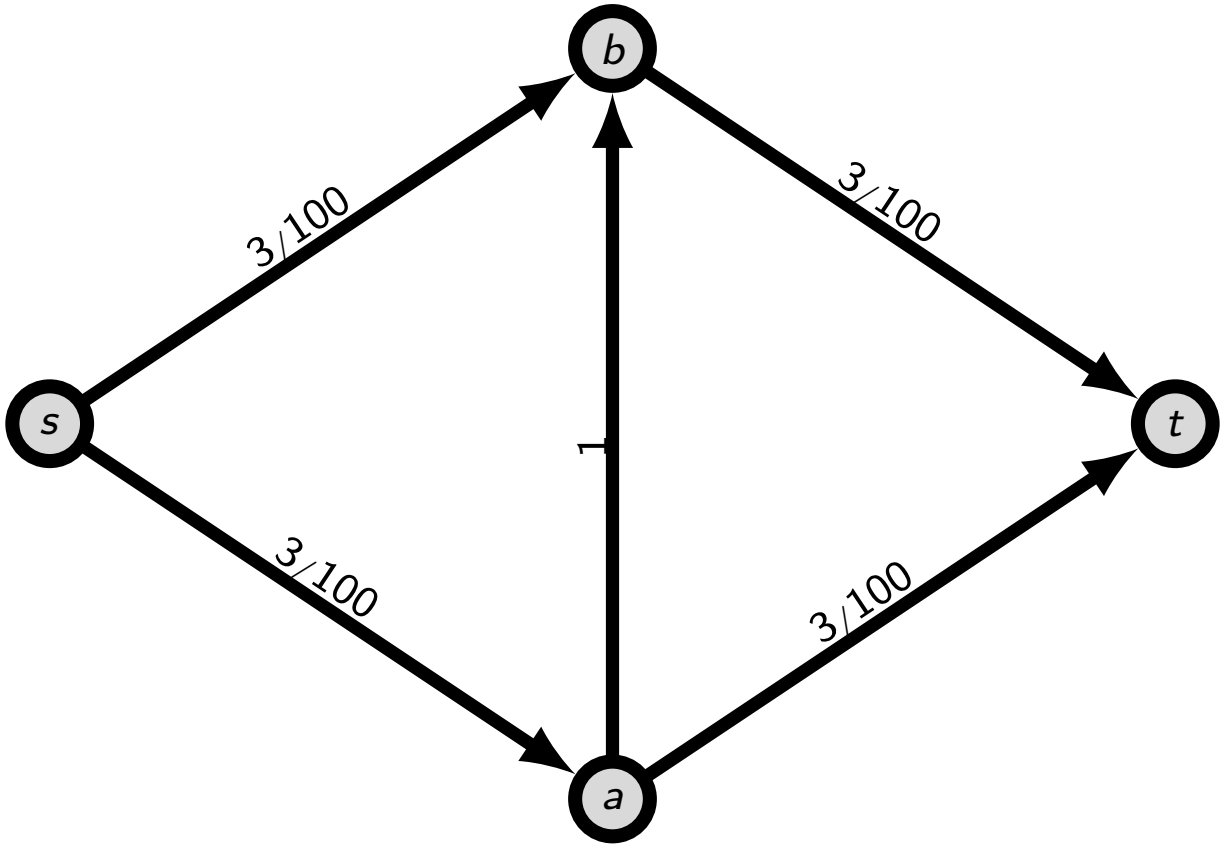
Beispiel

$n = |V|, m = |E|$



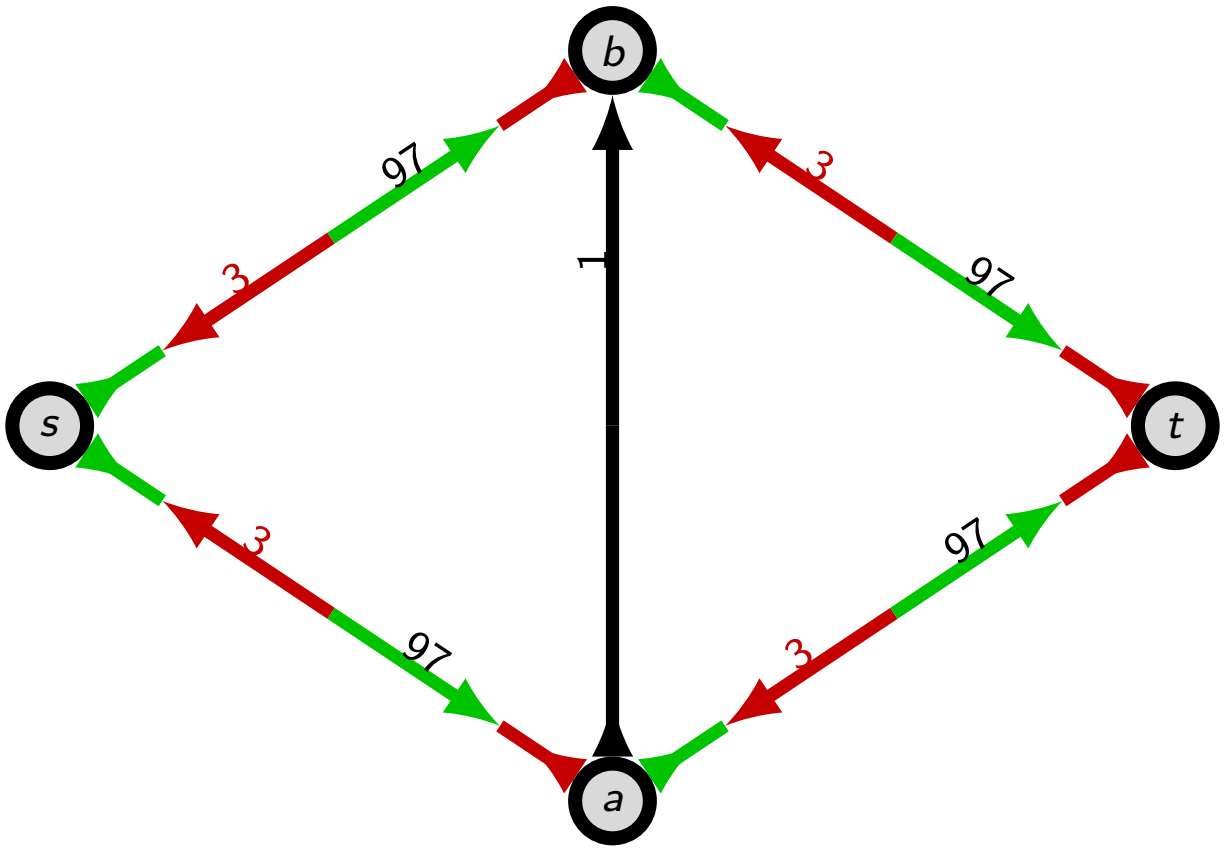
Beispiel

$n = |V|, m = |E|$



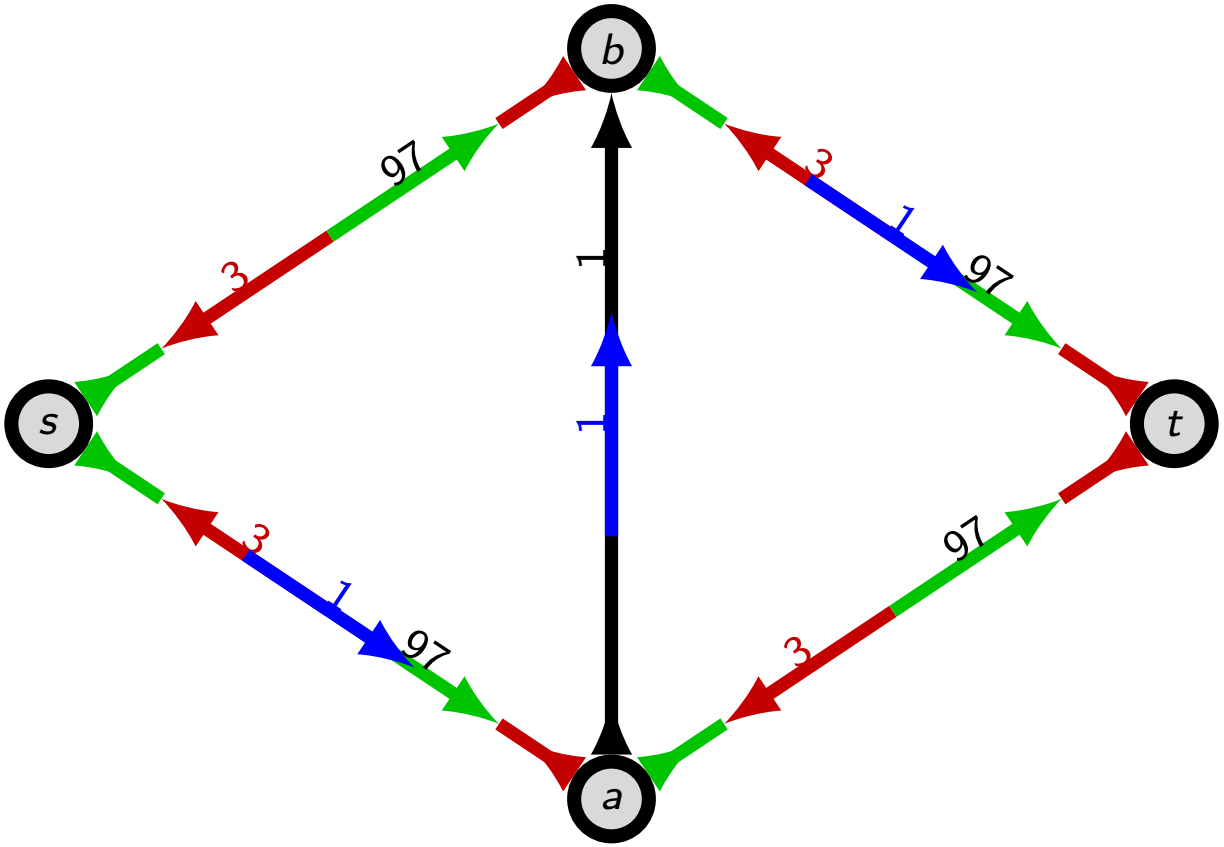
Beispiel

$n = |V|, m = |E|$



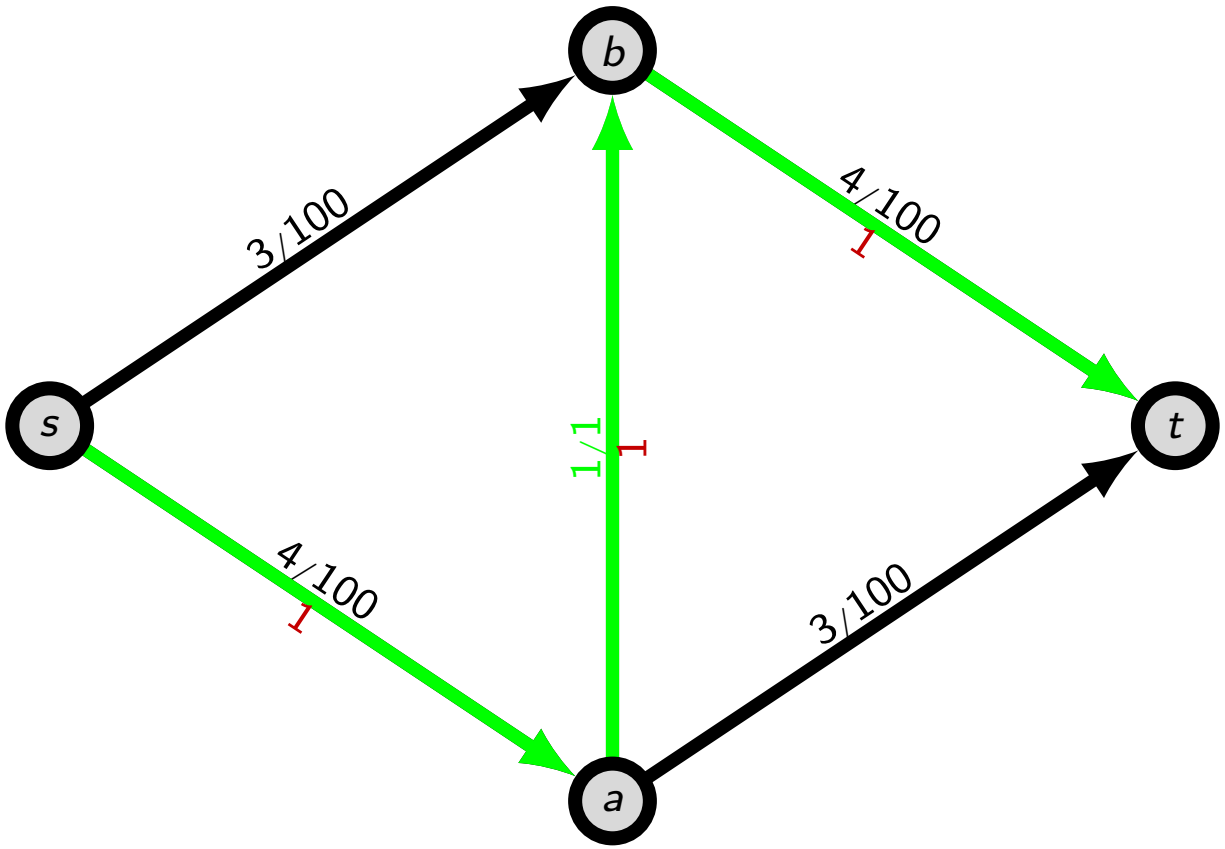
Beispiel

$n = |V|, m = |E|$



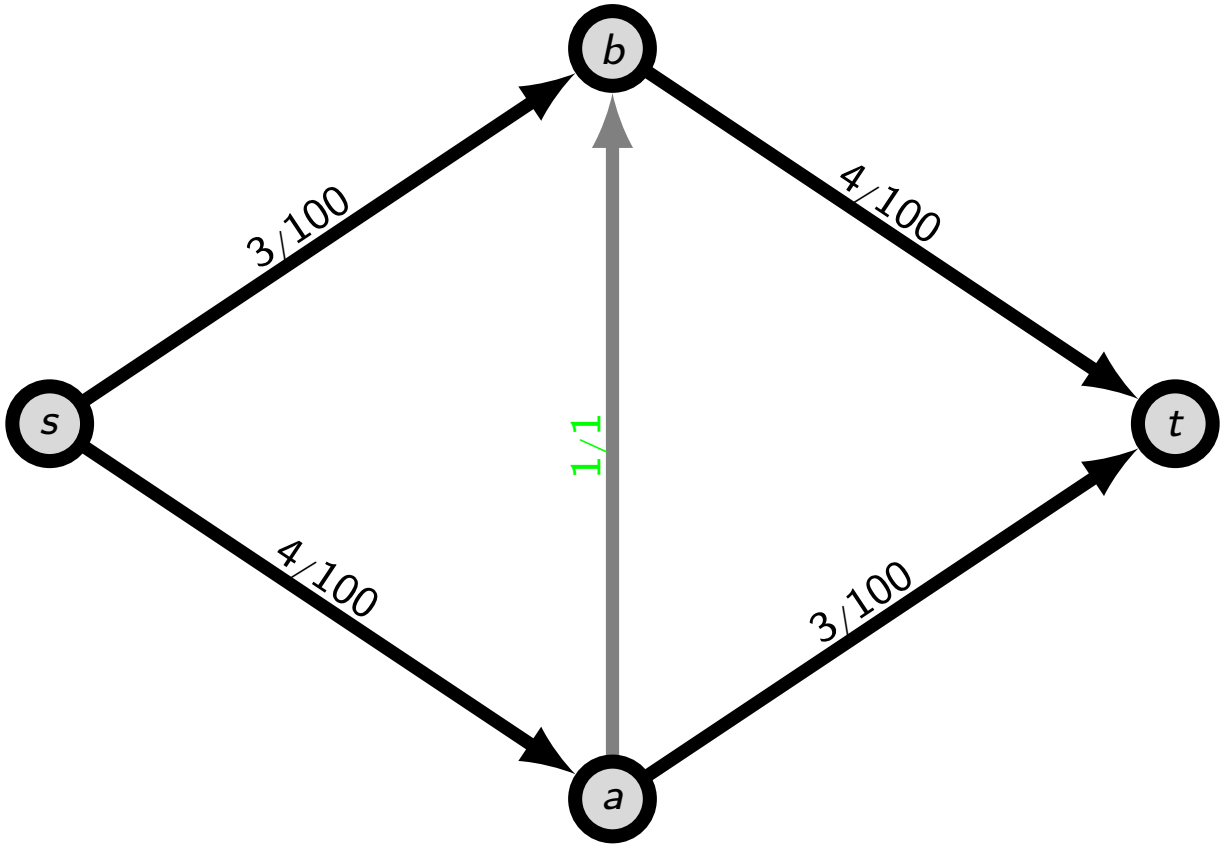
Beispiel

$n = |V|, m = |E|$



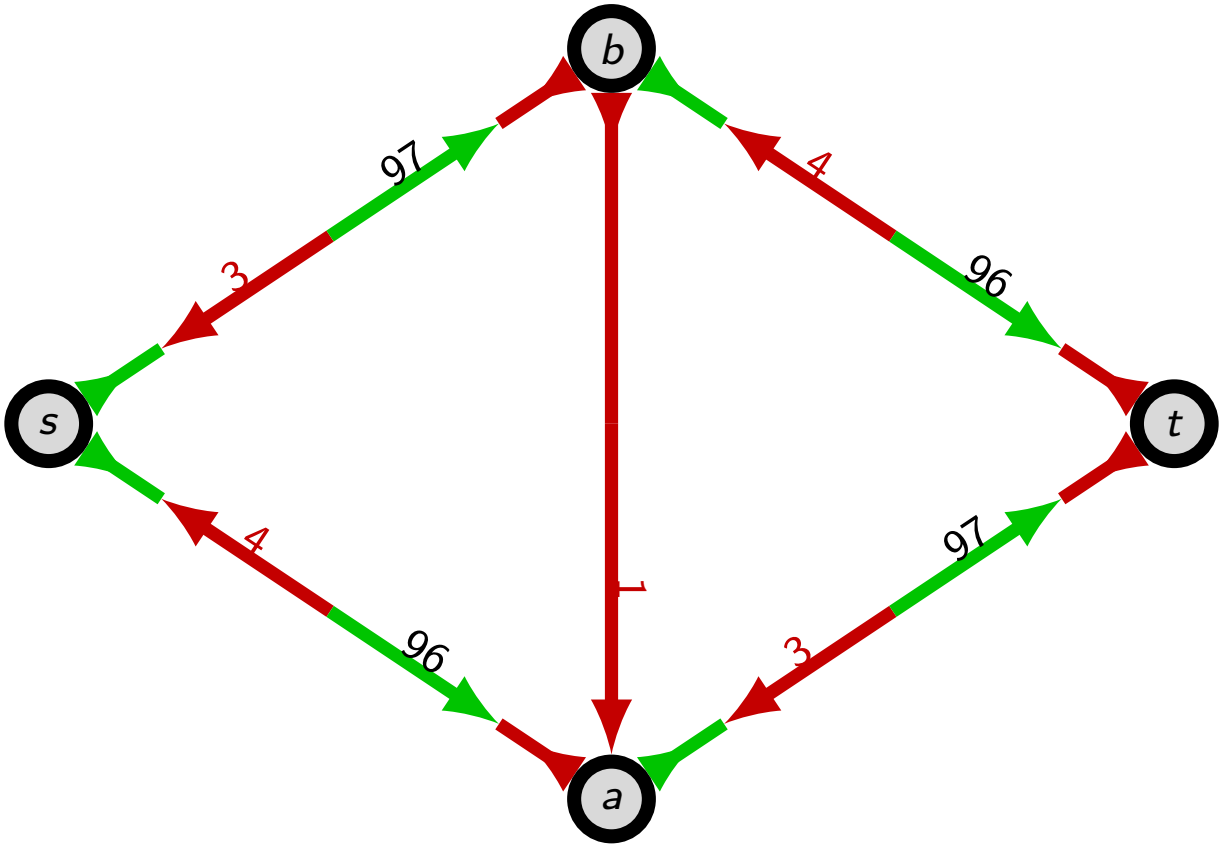
Beispiel

$n = |V|, m = |E|$



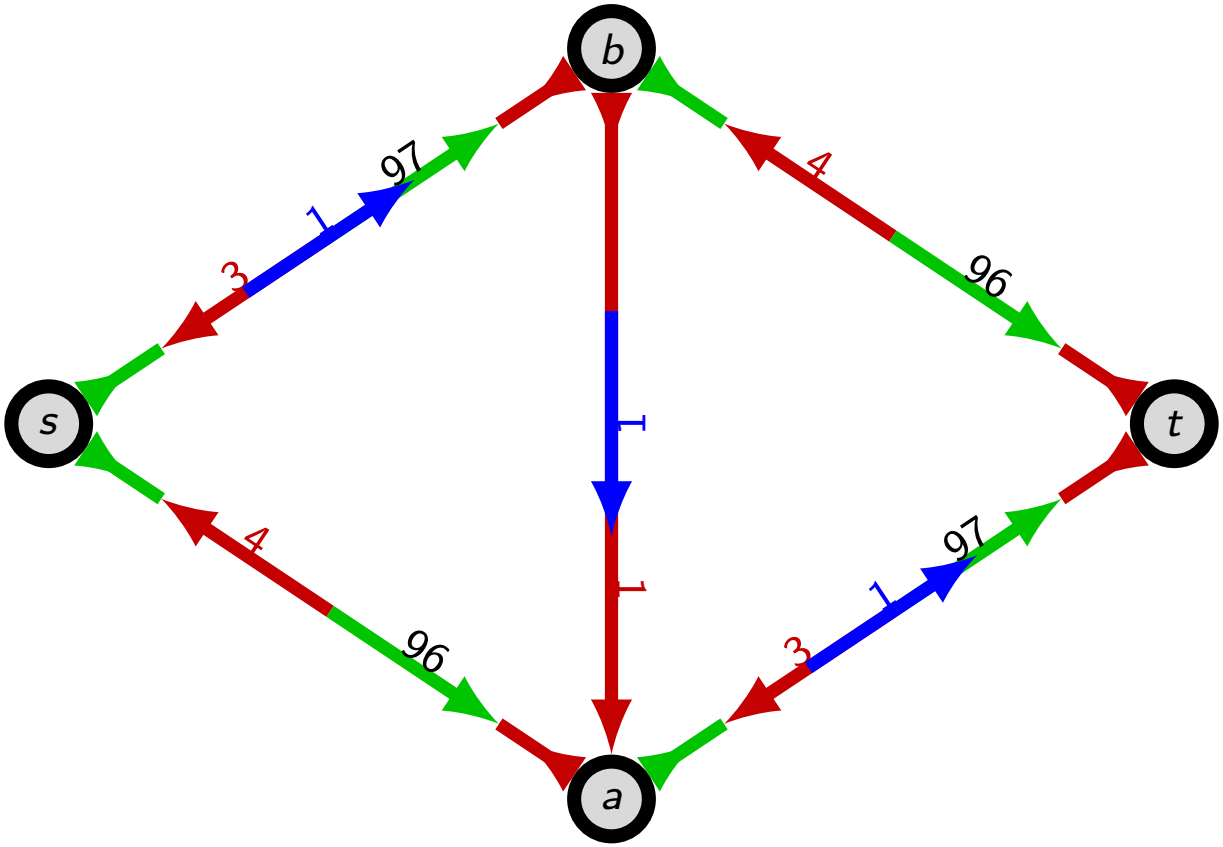
Beispiel

$n = |V|, m = |E|$



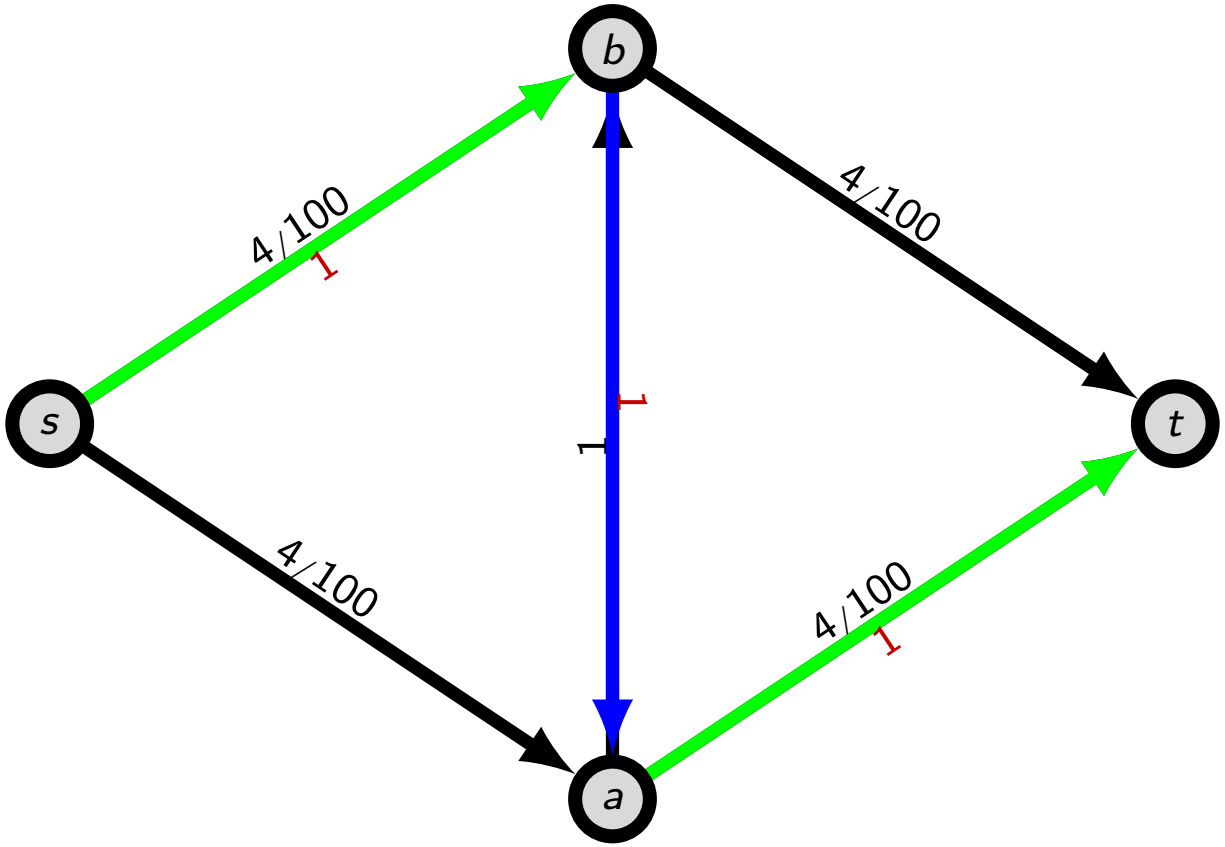
Beispiel

$n = |V|, m = |E|$



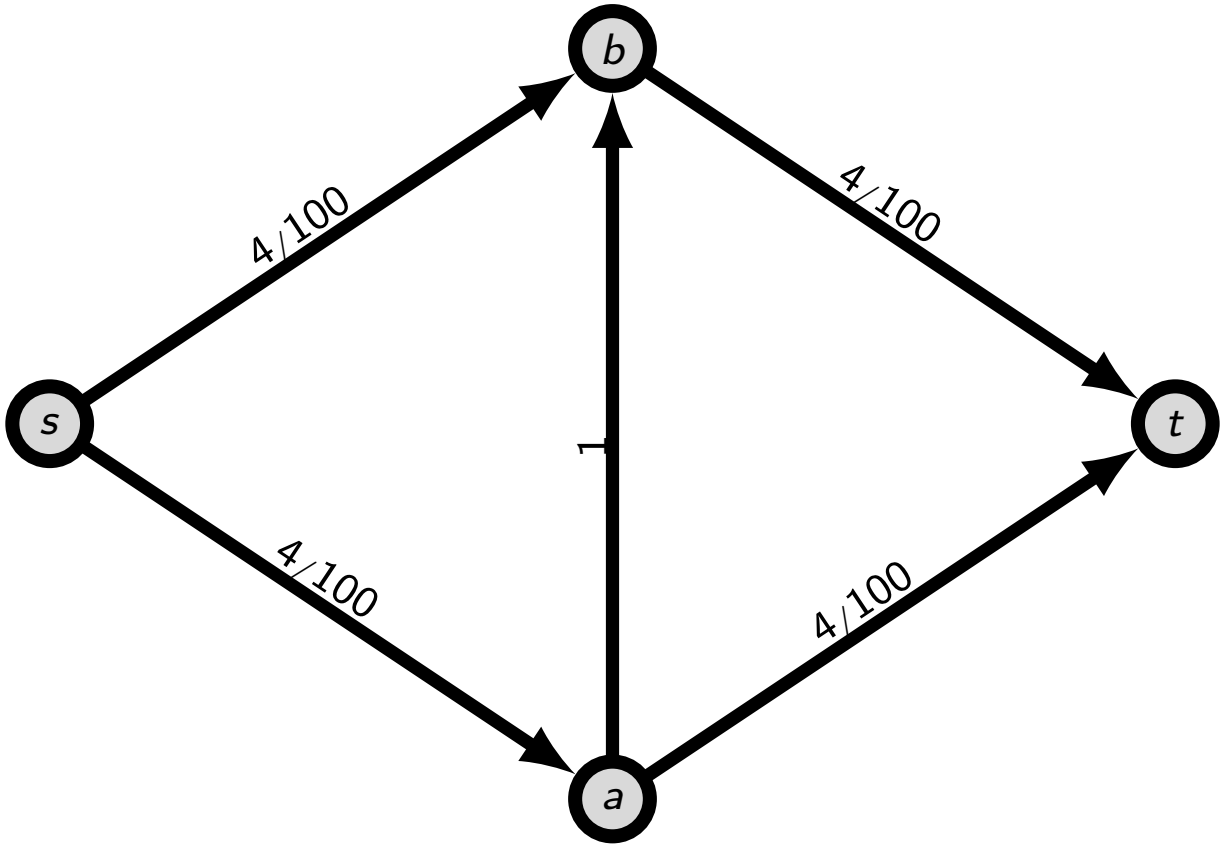
Beispiel

$n = |V|, m = |E|$



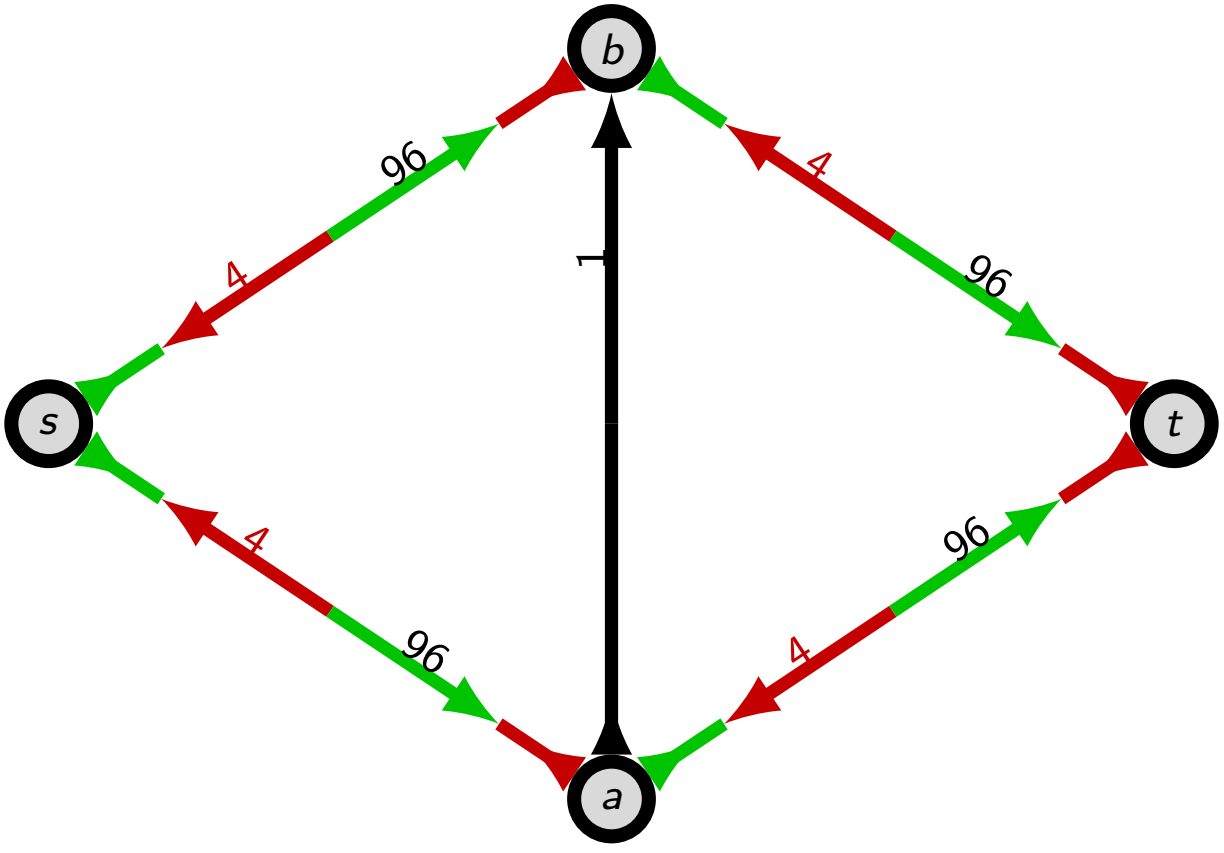
Beispiel

$n = |V|, m = |E|$



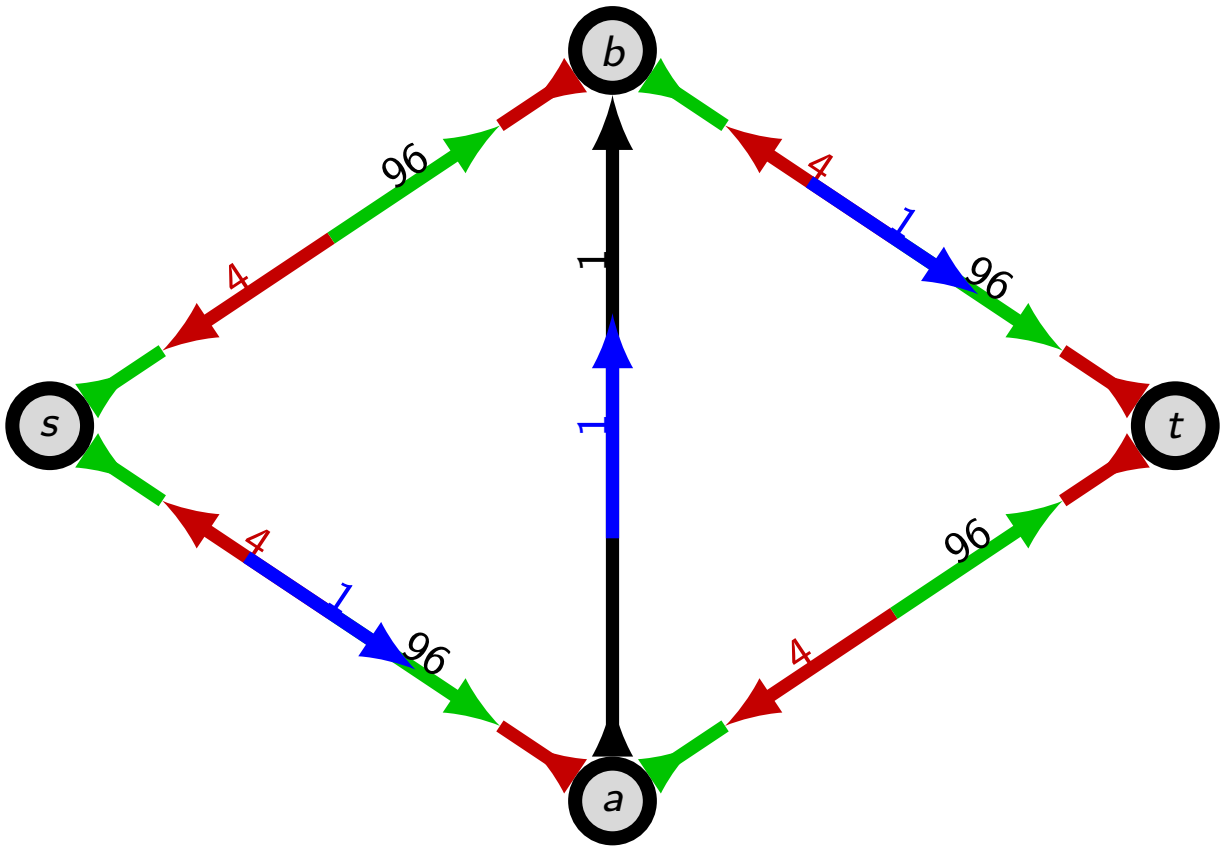
Beispiel

$n = |V|, m = |E|$



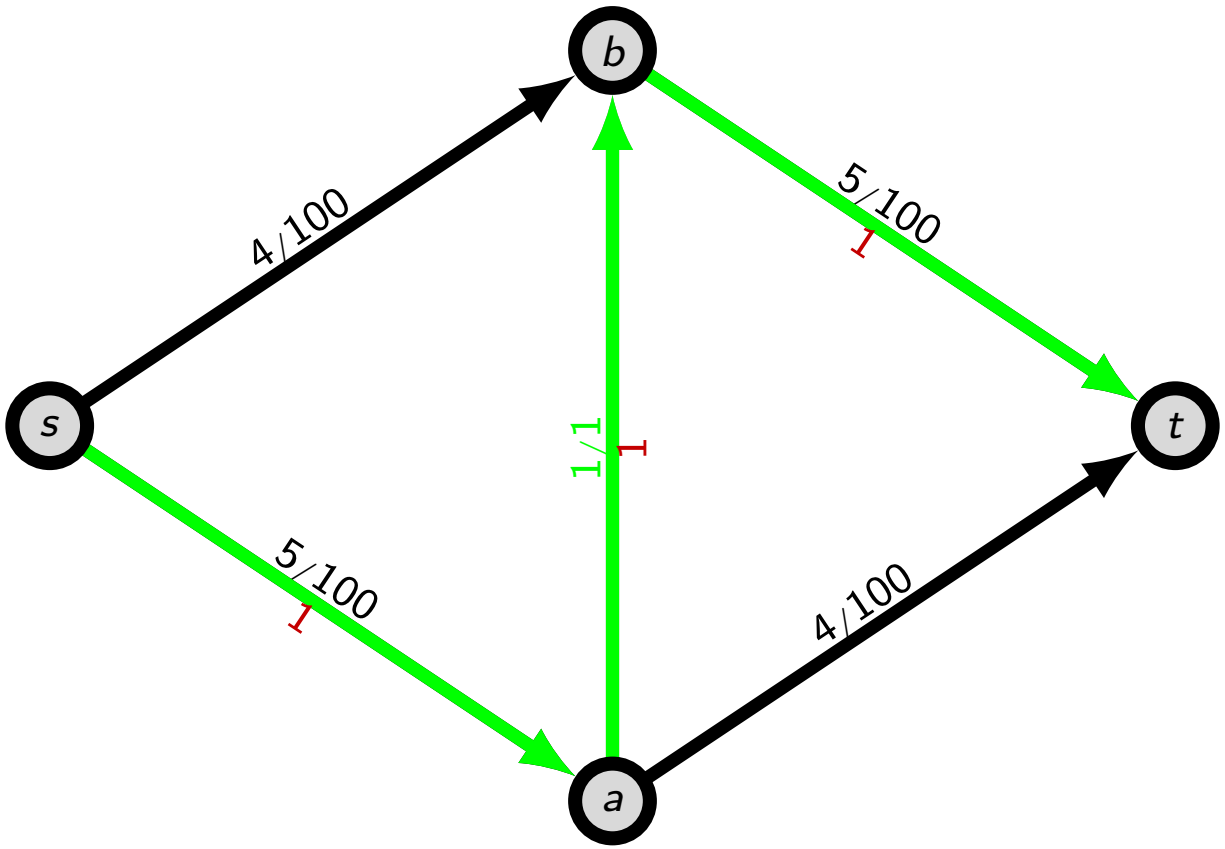
Beispiel

$n = |V|, m = |E|$



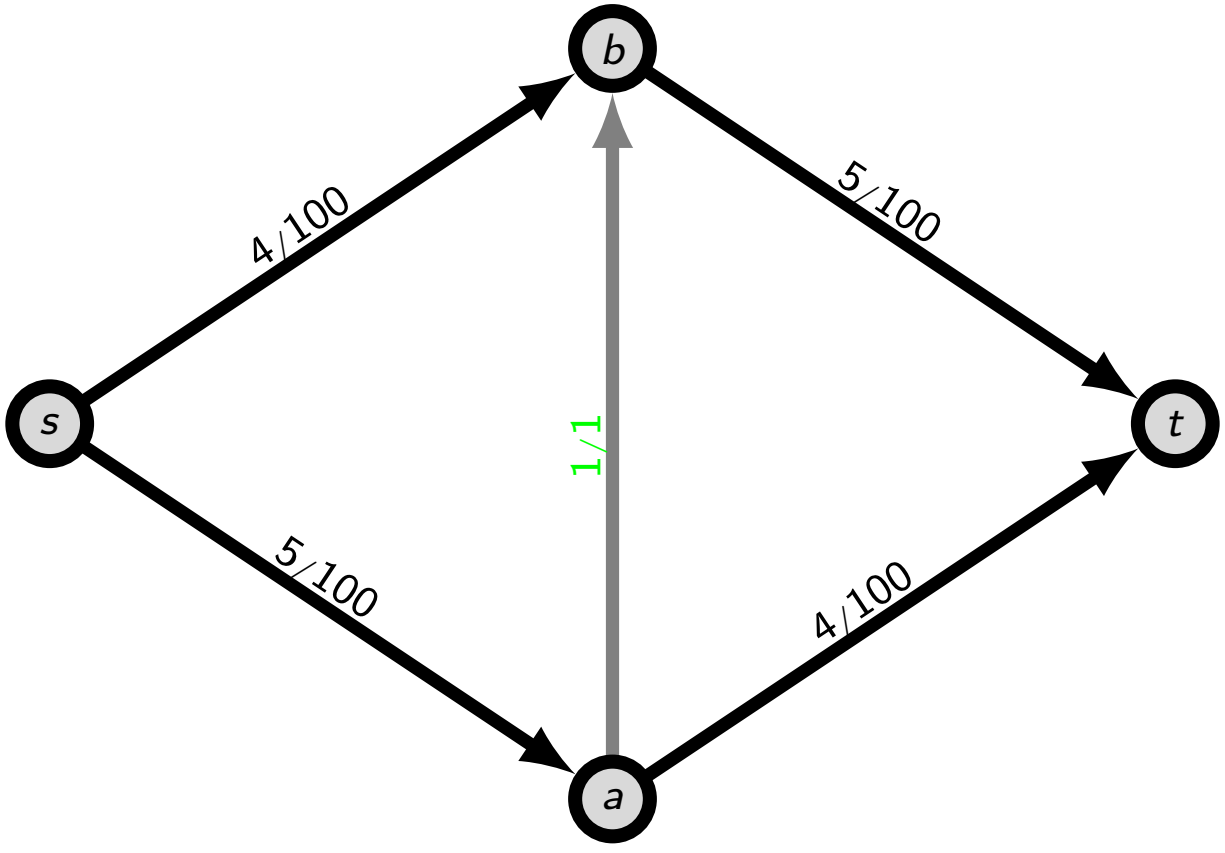
Beispiel

$n = |V|, m = |E|$



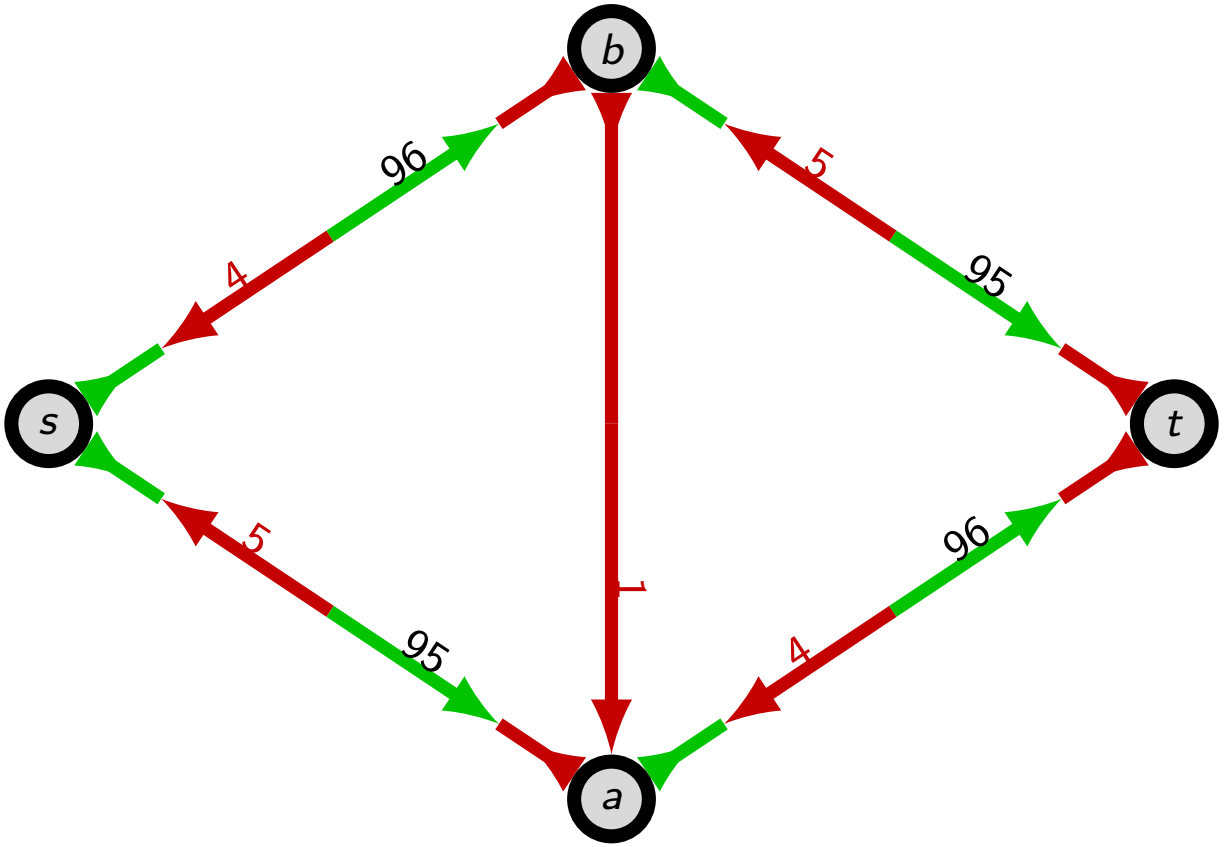
Beispiel

$n = |V|, m = |E|$



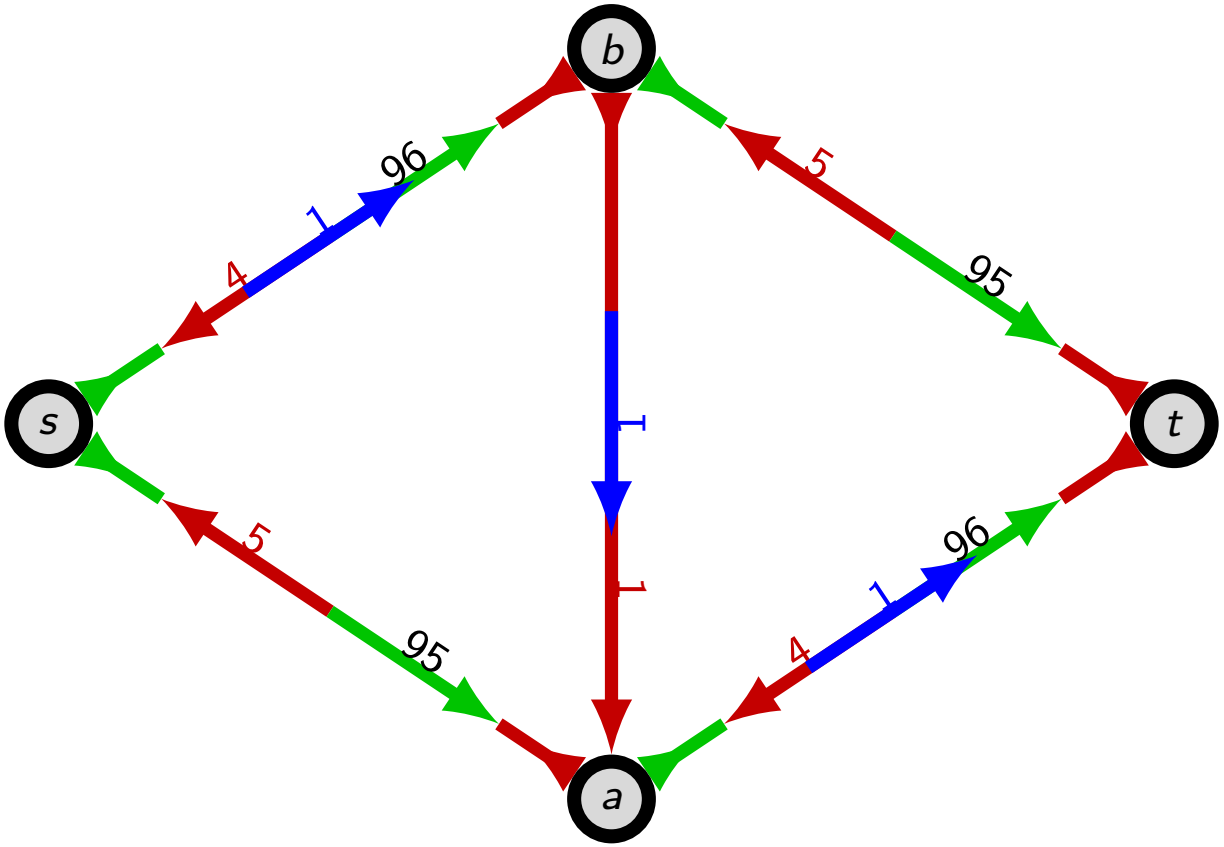
Beispiel

$n = |V|, m = |E|$



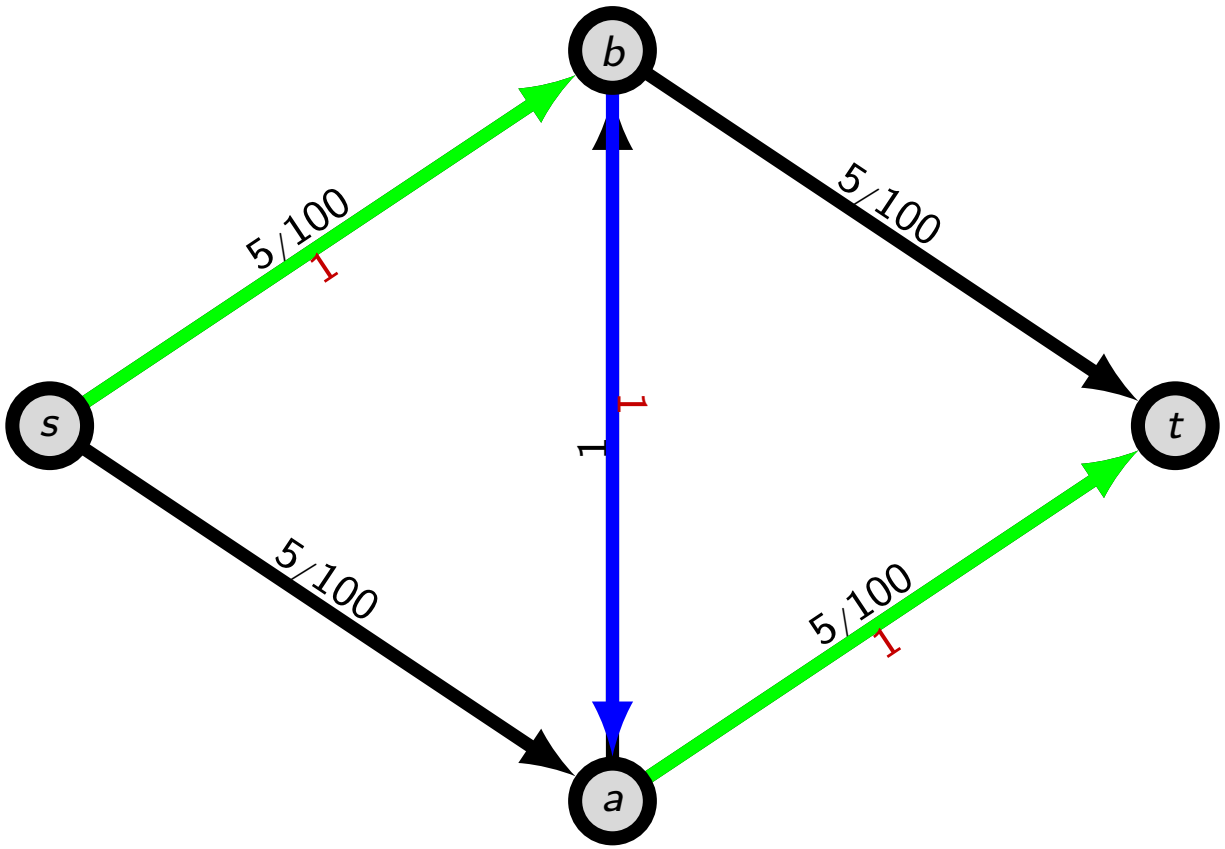
Beispiel

$n = |V|, m = |E|$



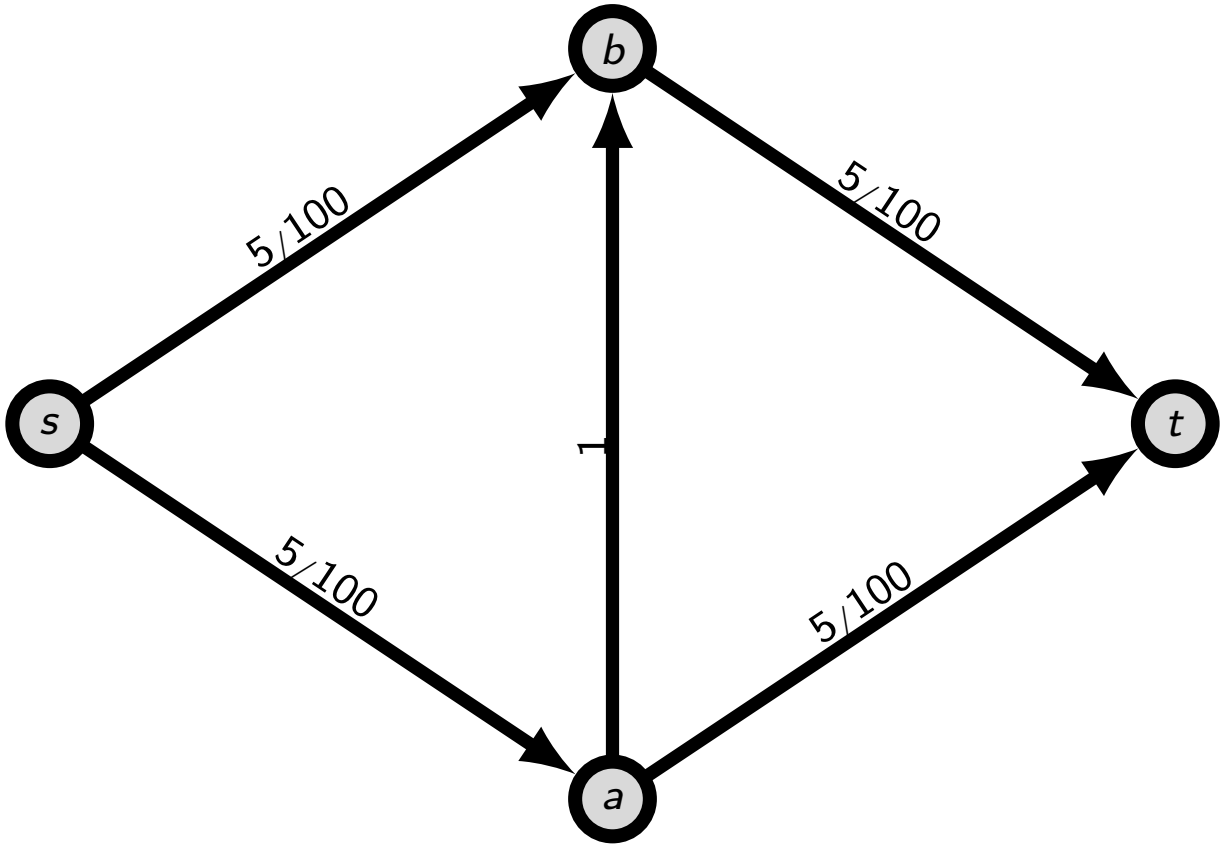
Beispiel

$n = |V|, m = |E|$



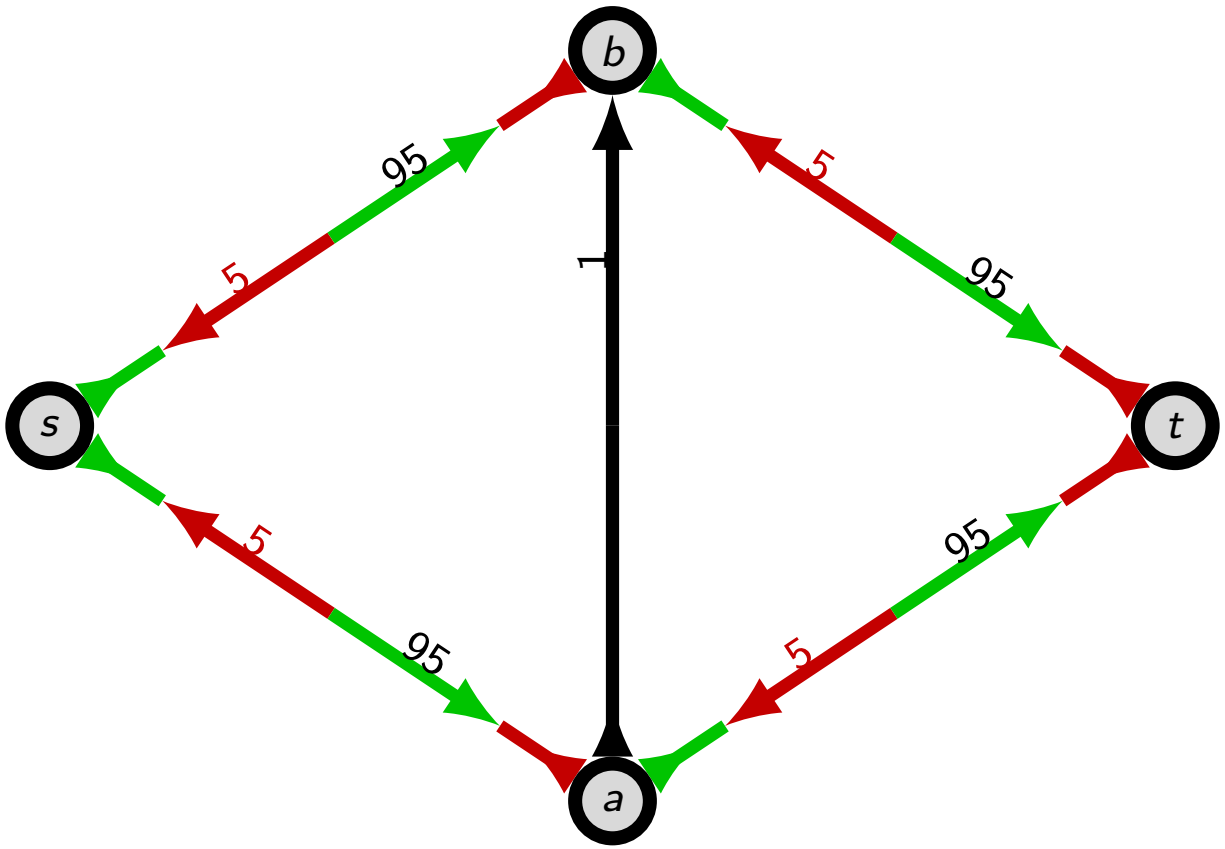
Beispiel

$n = |V|, m = |E|$



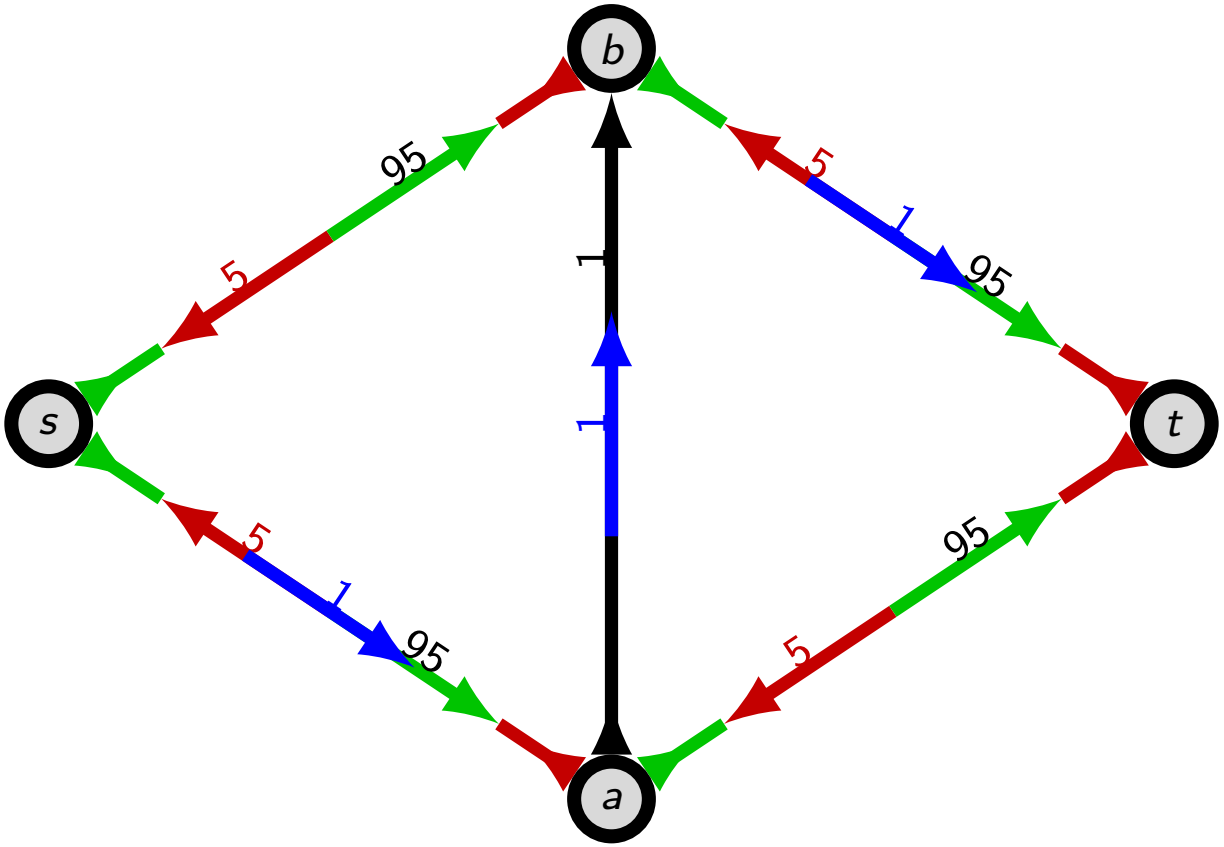
Beispiel

$n = |V|, m = |E|$



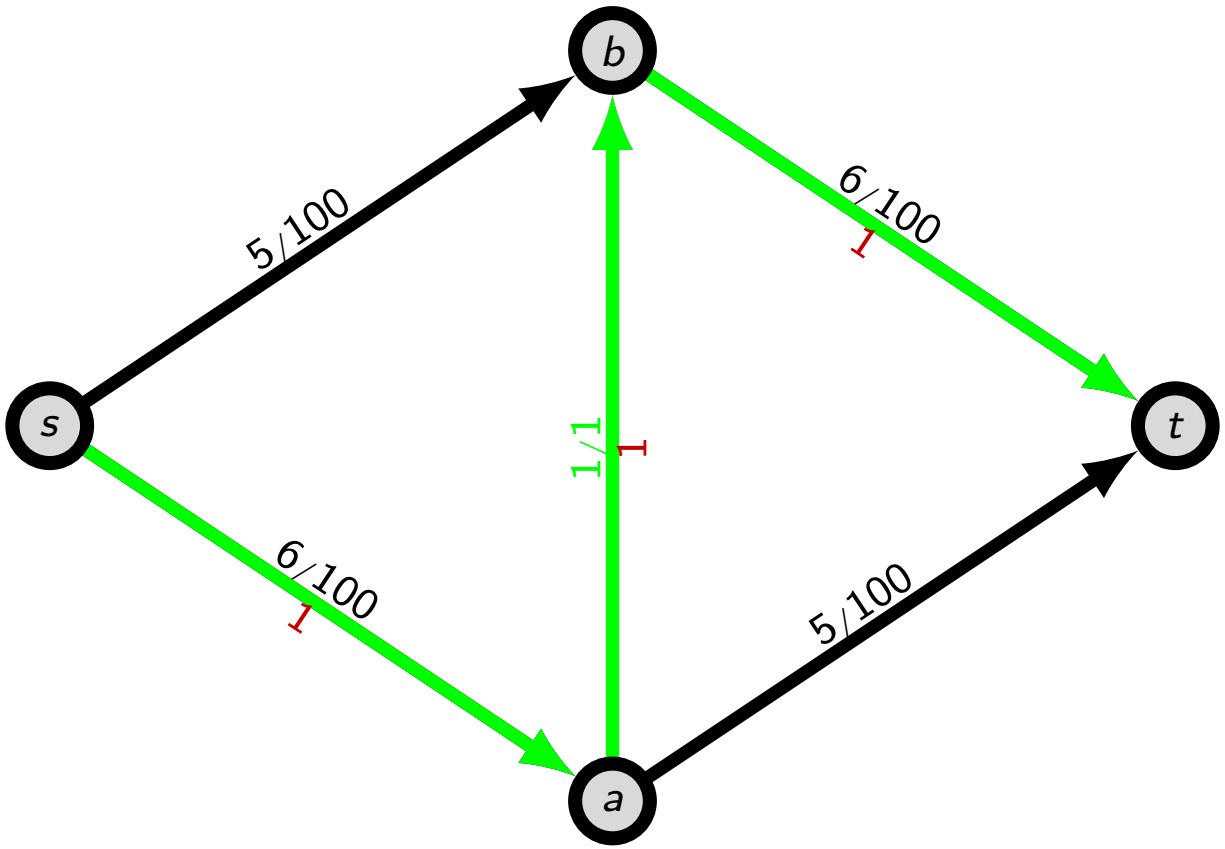
Beispiel

$n = |V|, m = |E|$



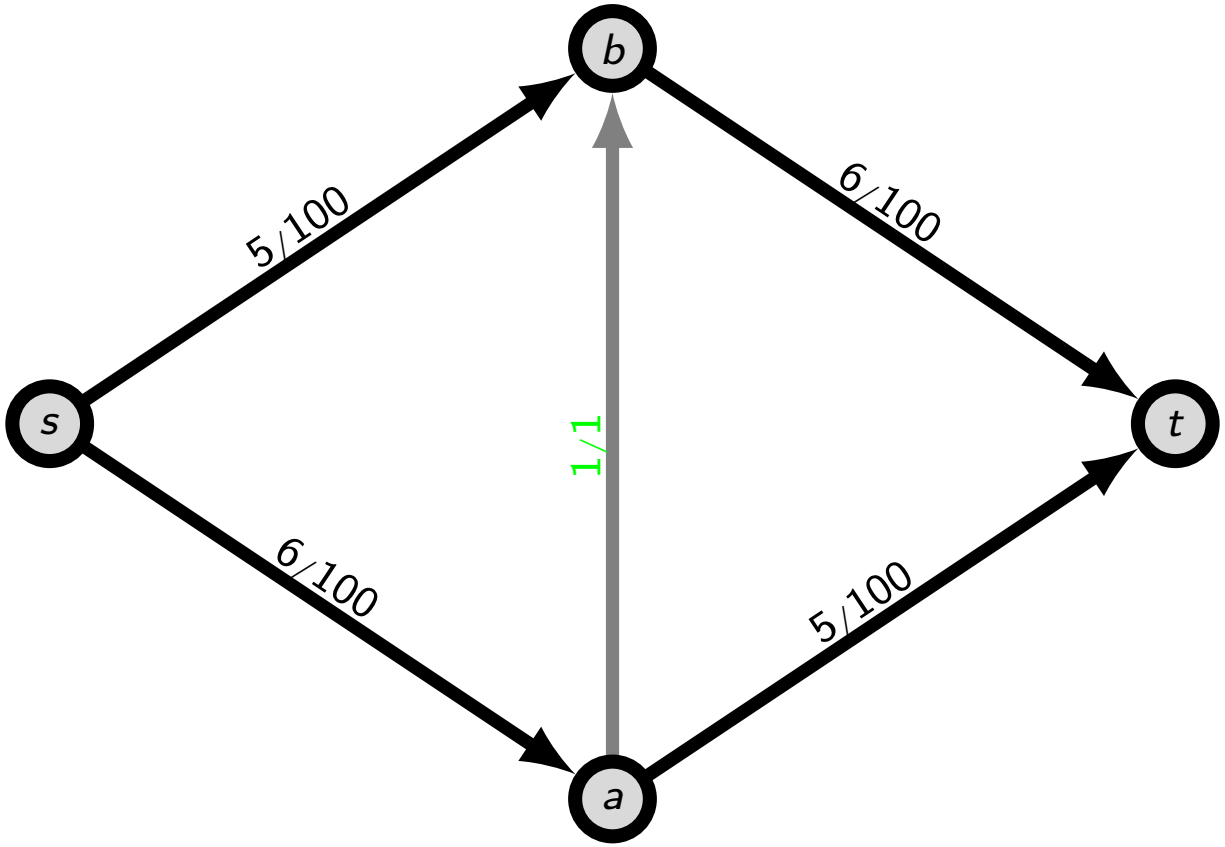
Beispiel

$n = |V|, m = |E|$



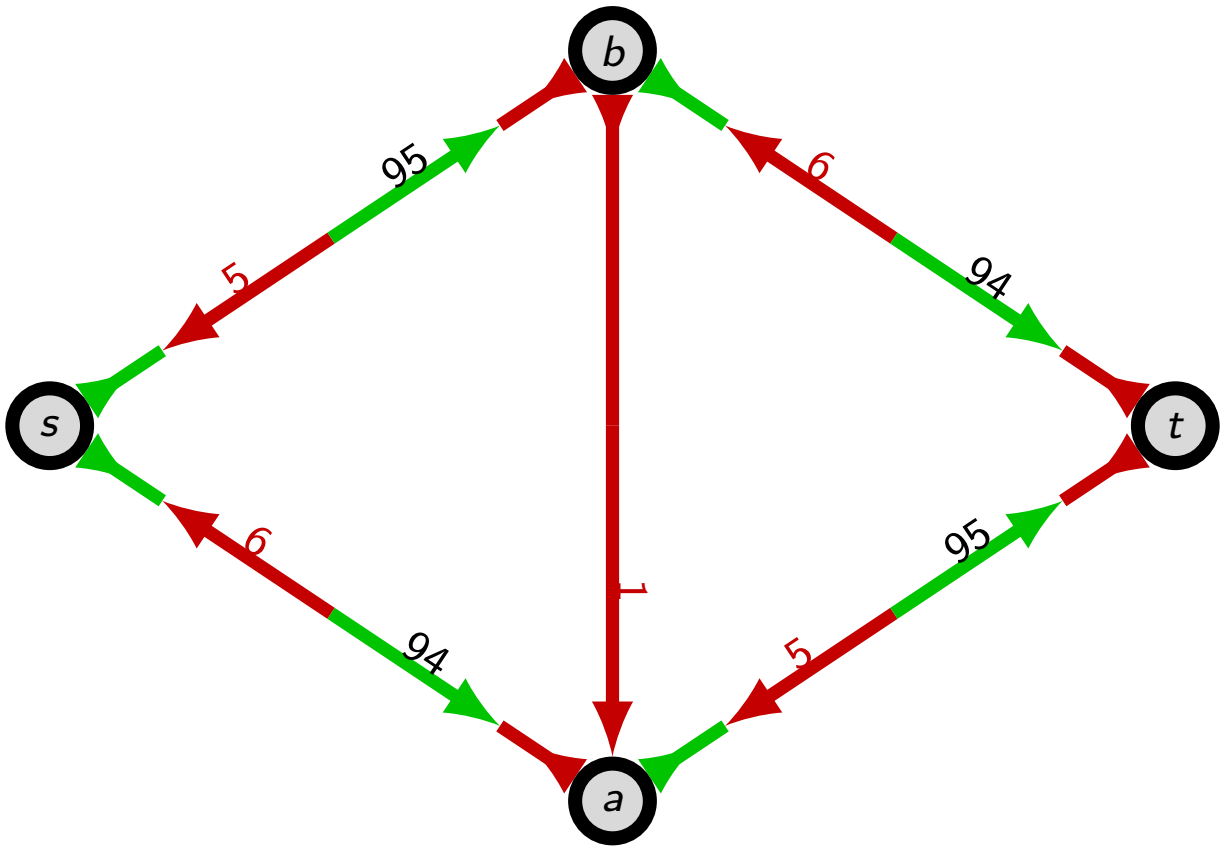
Beispiel

$n = |V|, m = |E|$



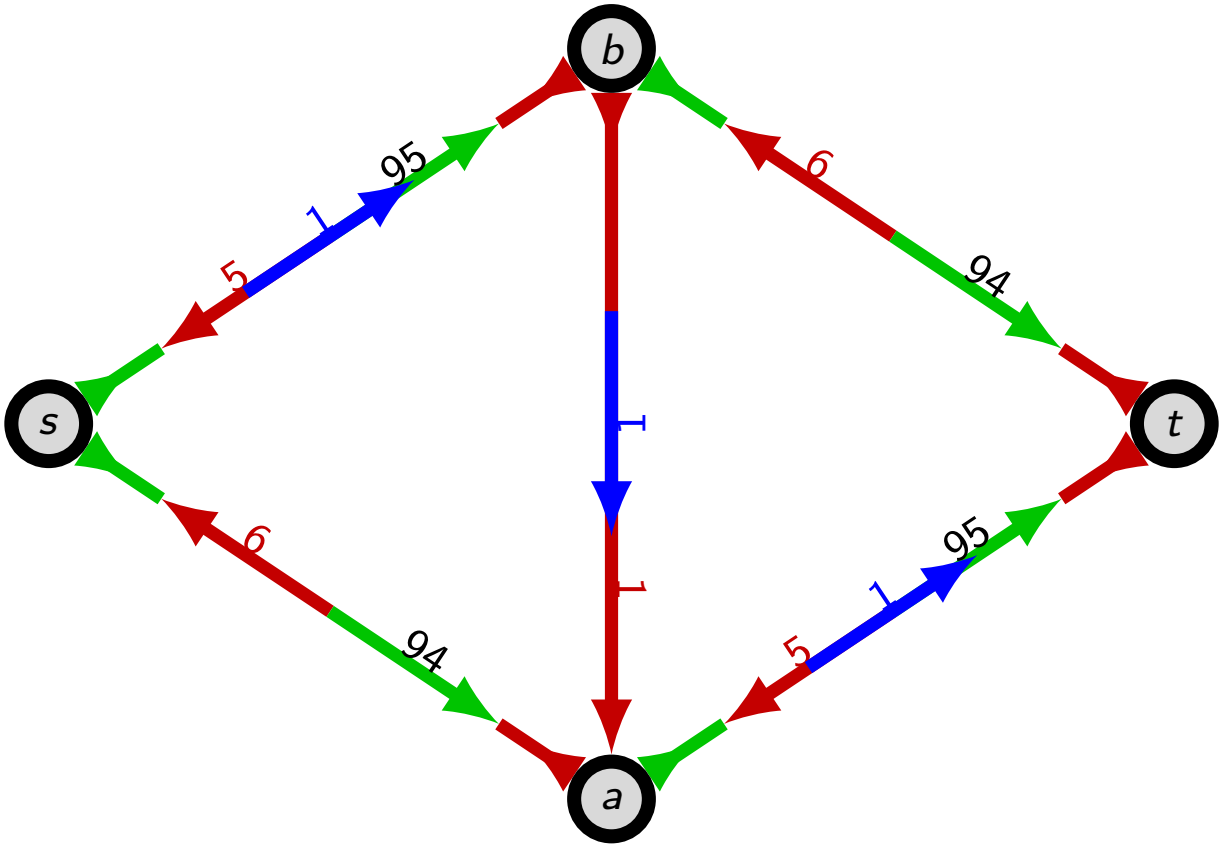
Beispiel

$n = |V|, m = |E|$



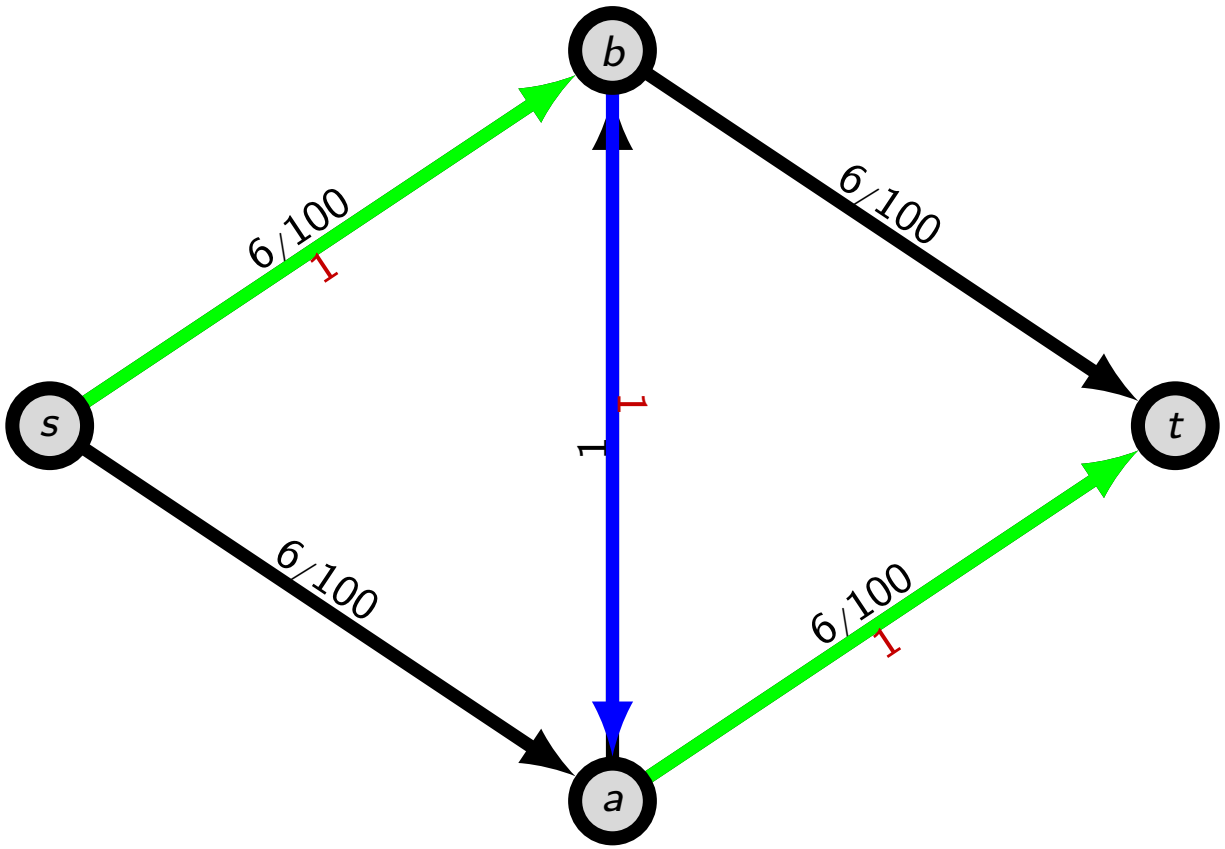
Beispiel

$n = |V|, m = |E|$



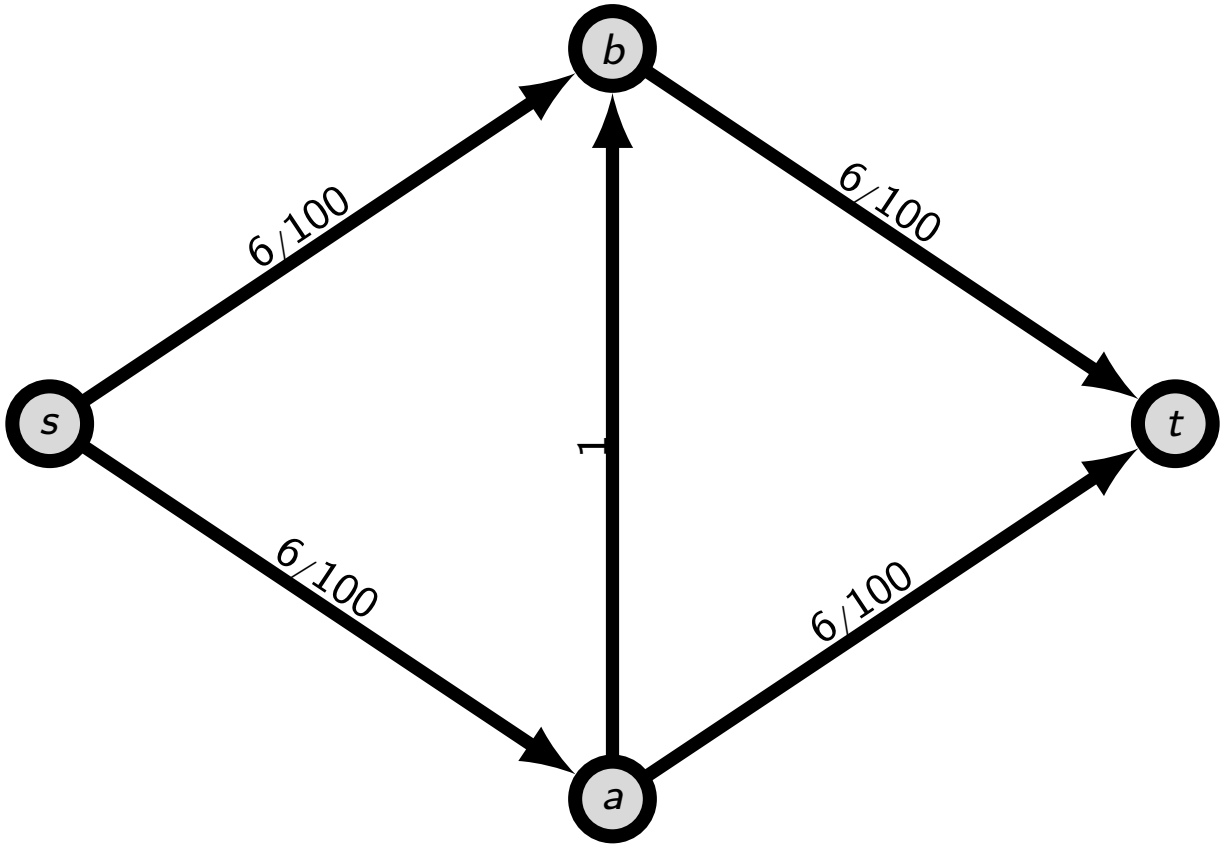
Beispiel

$n = |V|, m = |E|$



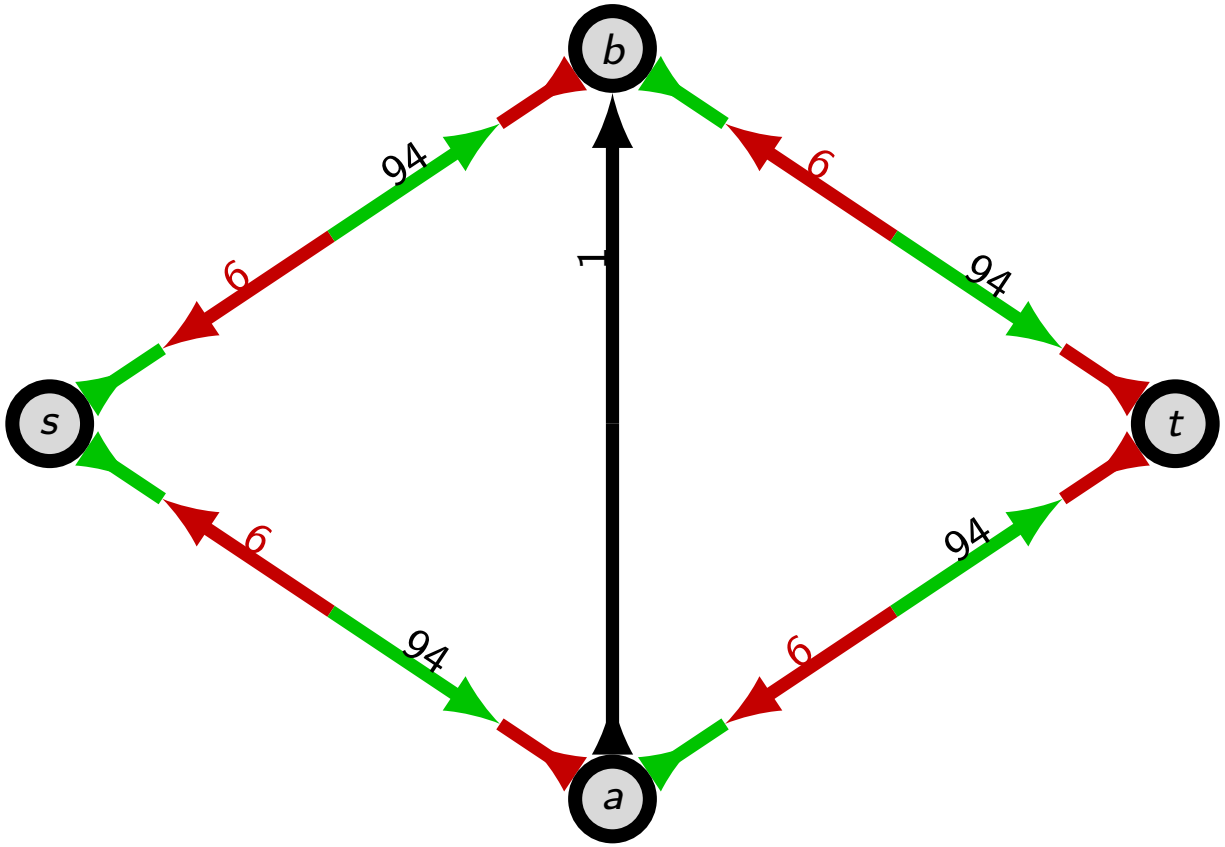
Beispiel

$n = |V|, m = |E|$



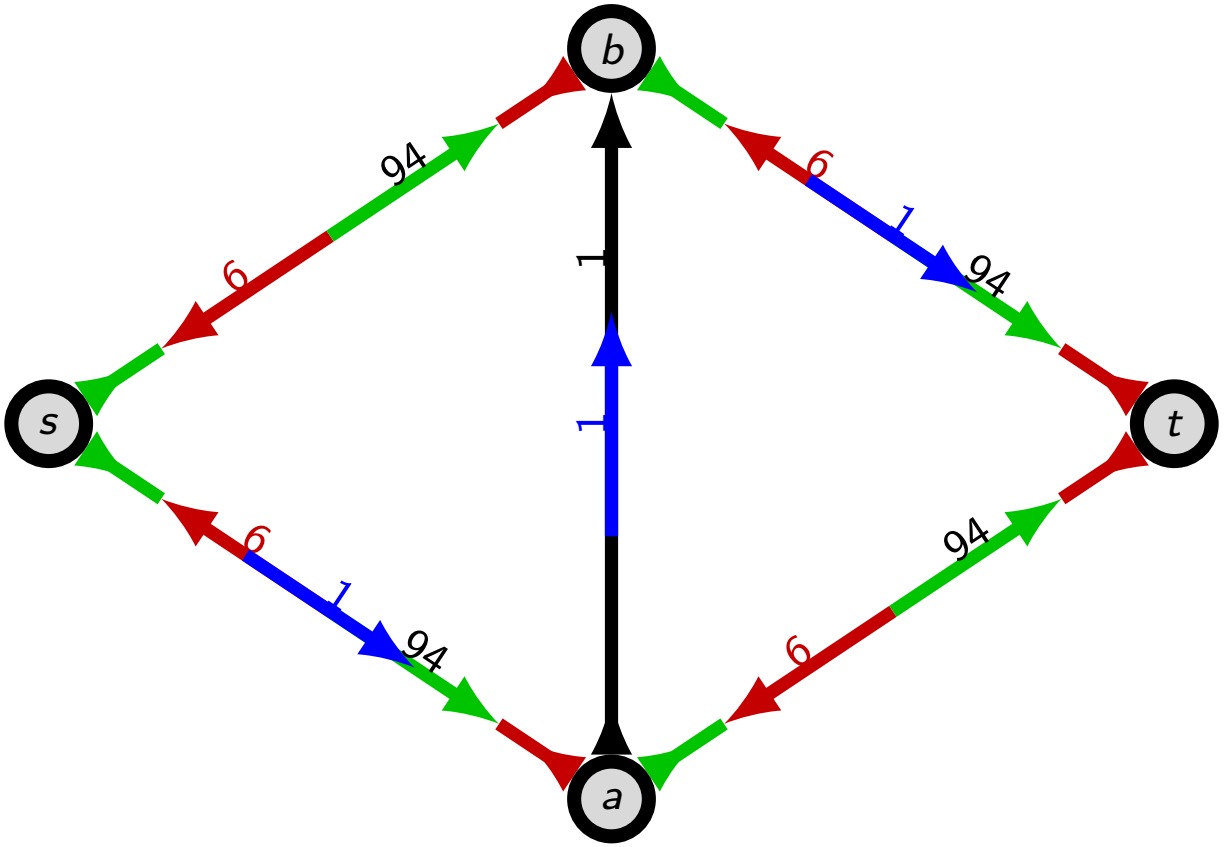
Beispiel

$n = |V|, m = |E|$



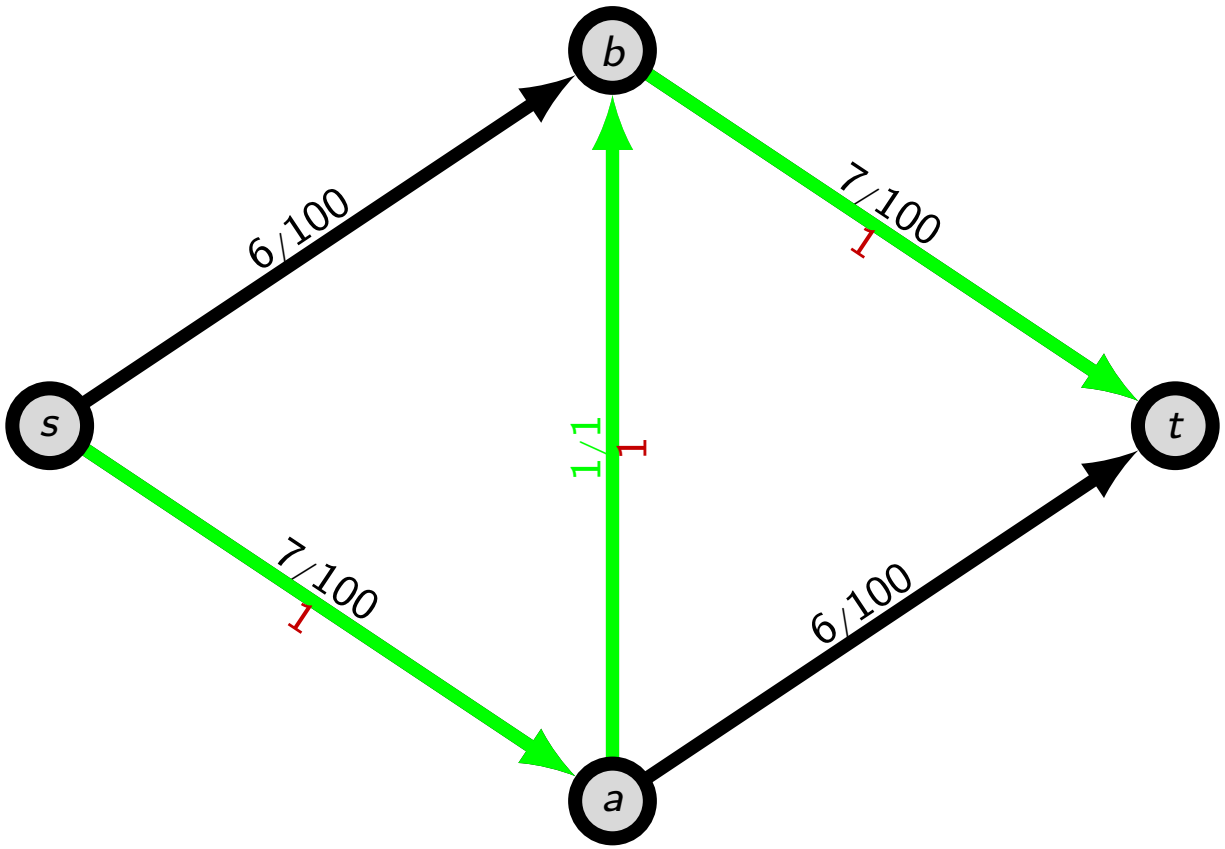
Beispiel

$n = |V|, m = |E|$



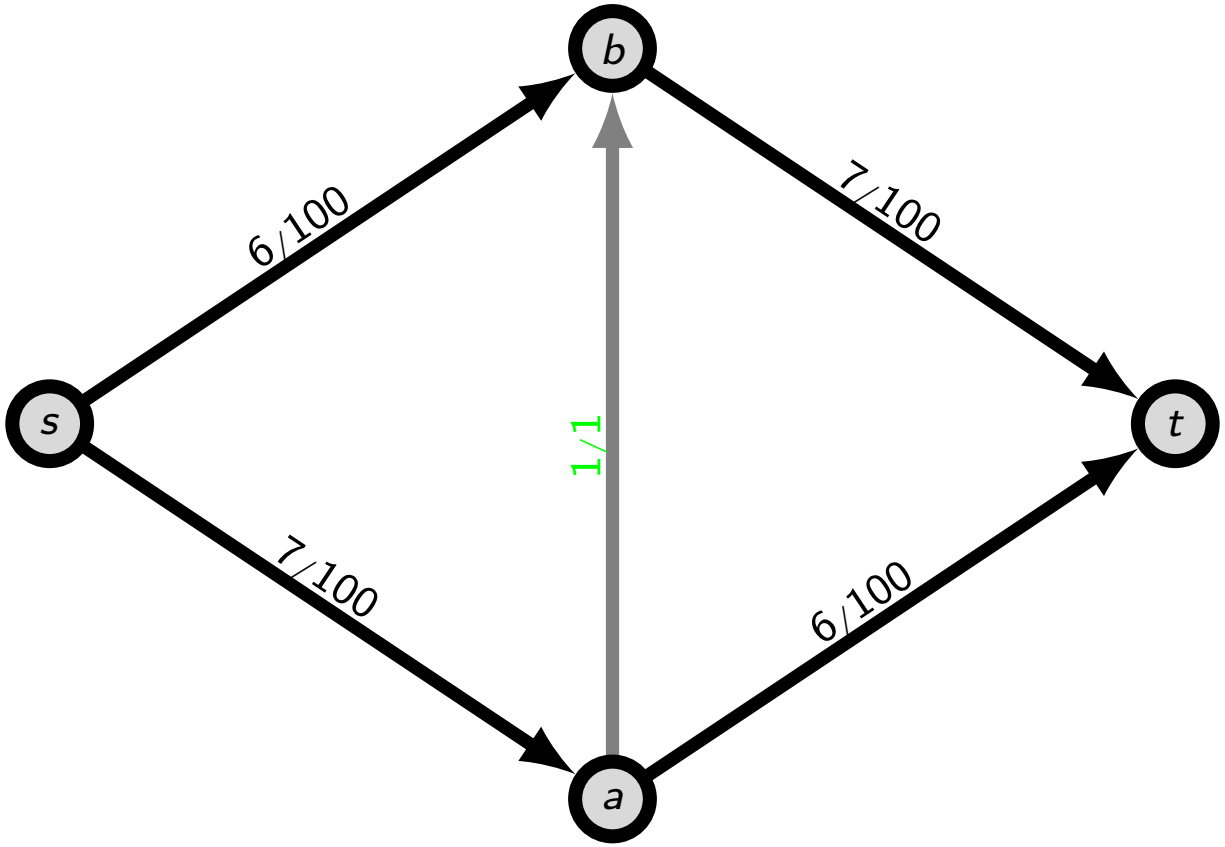
Beispiel

$n = |V|, m = |E|$



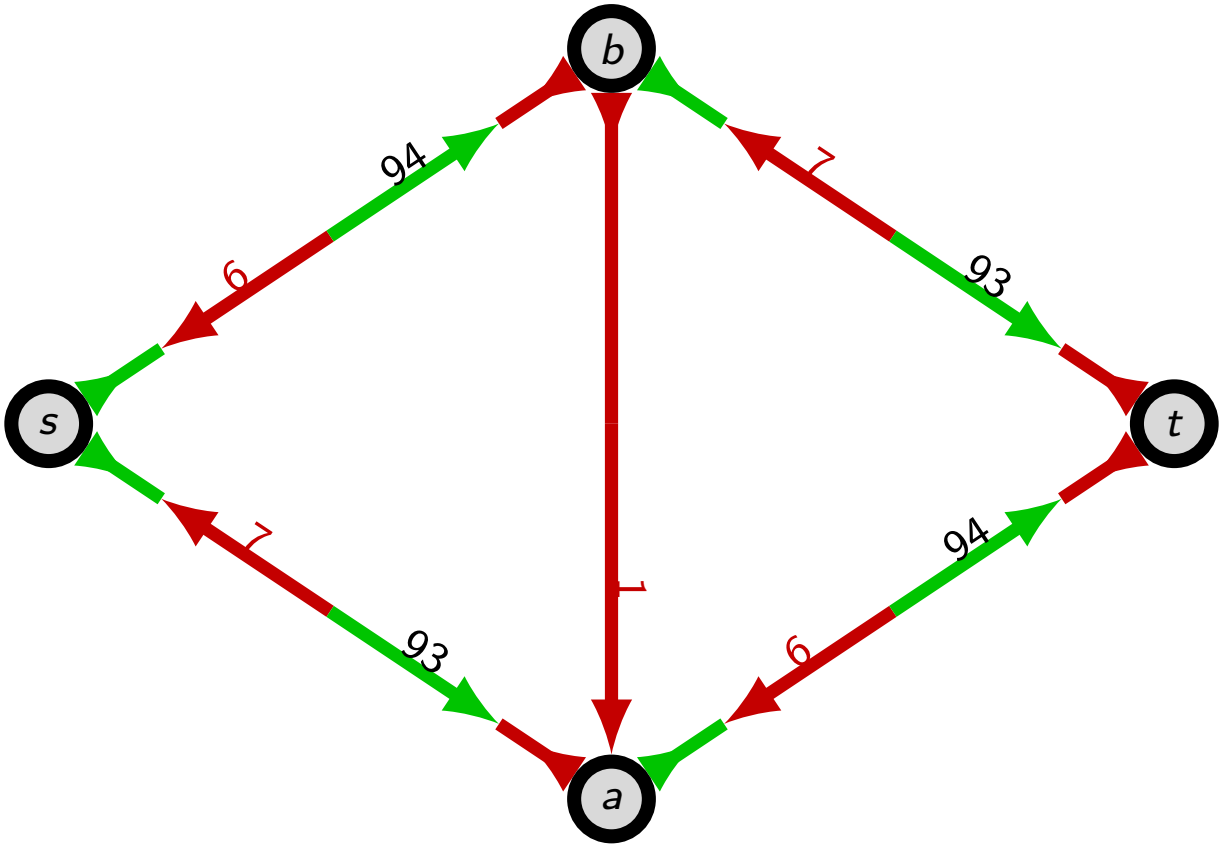
Beispiel

$n = |V|, m = |E|$



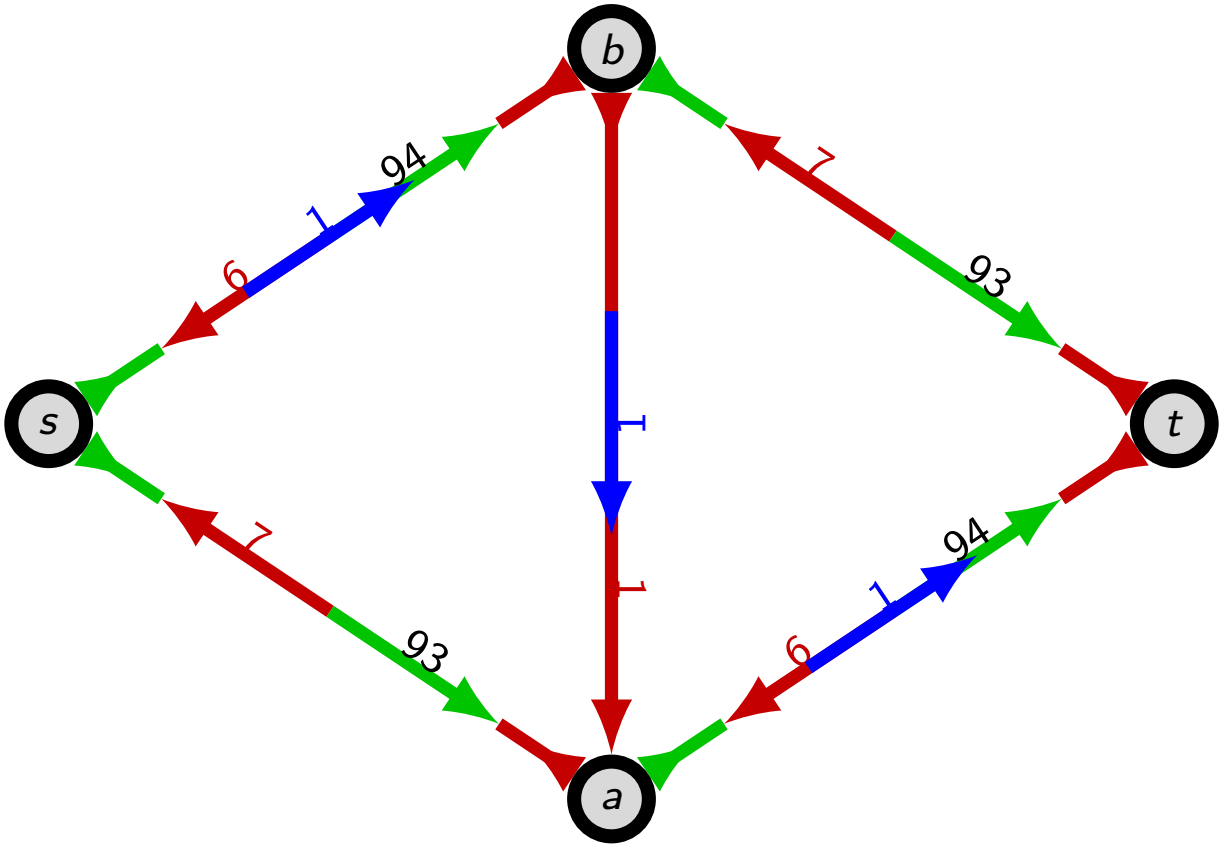
Beispiel

$n = |V|, m = |E|$



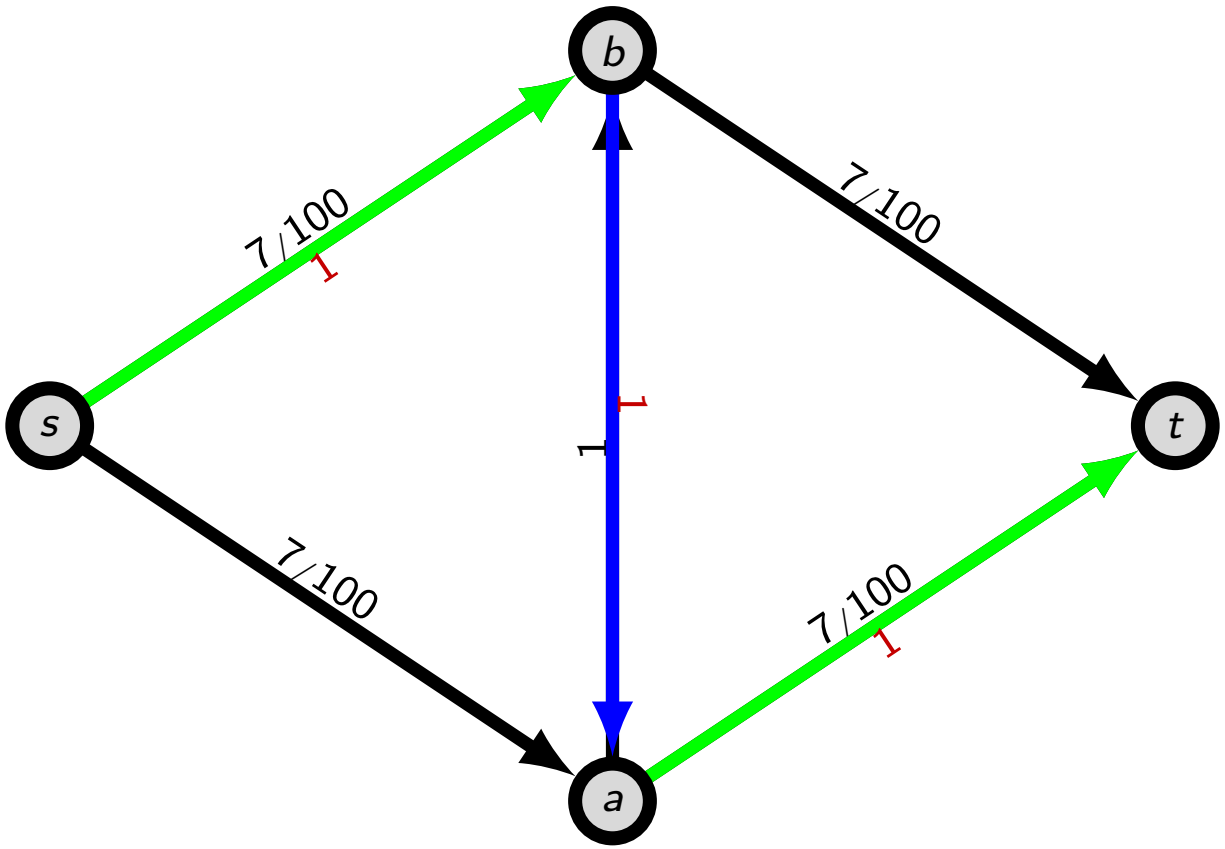
Beispiel

$n = |V|, m = |E|$



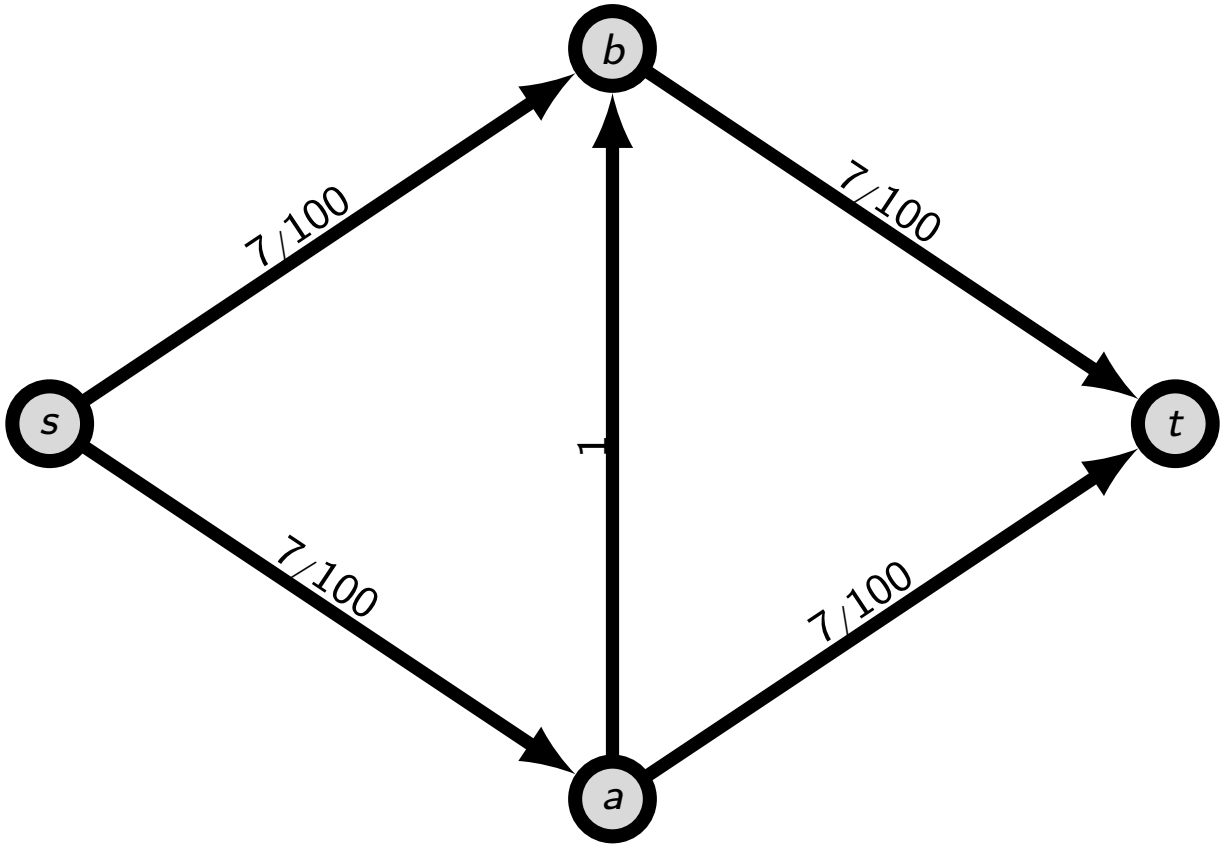
Beispiel

$n = |V|, m = |E|$



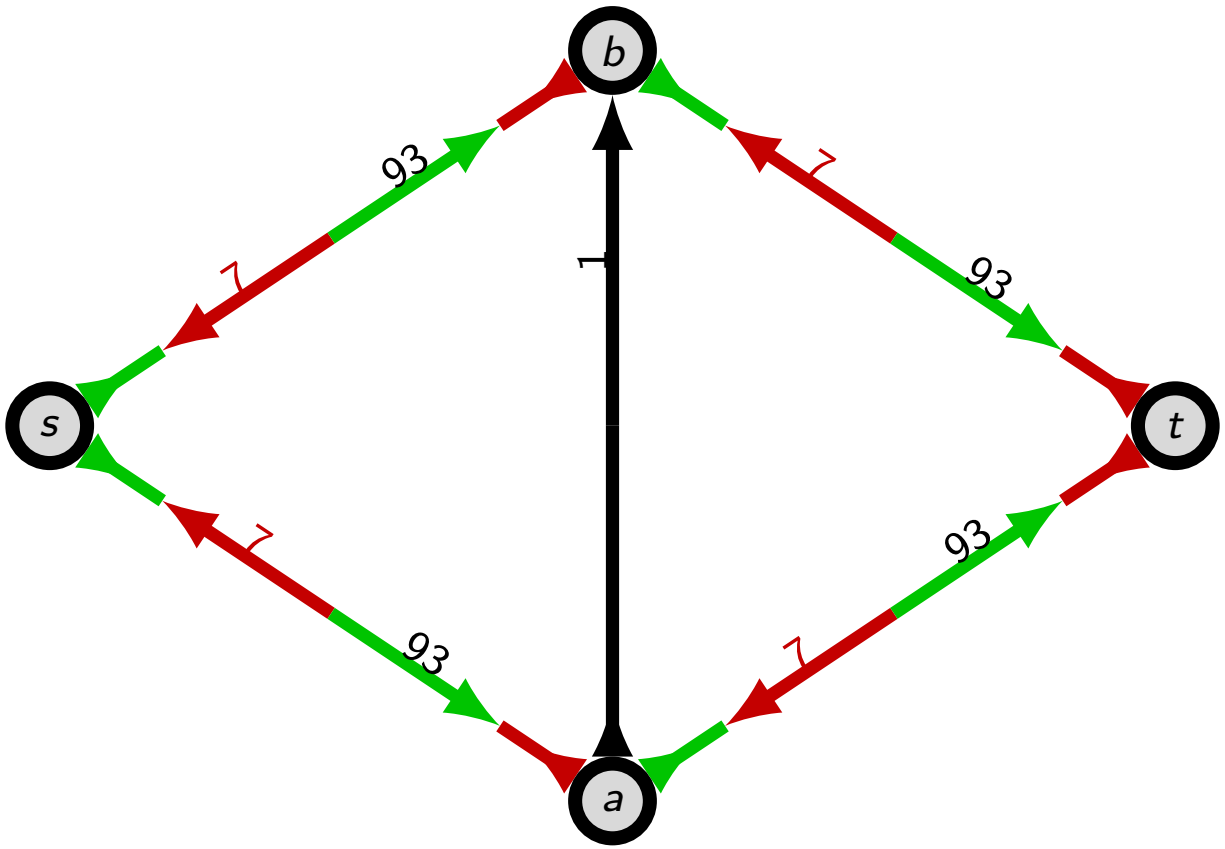
Beispiel

$n = |V|, m = |E|$



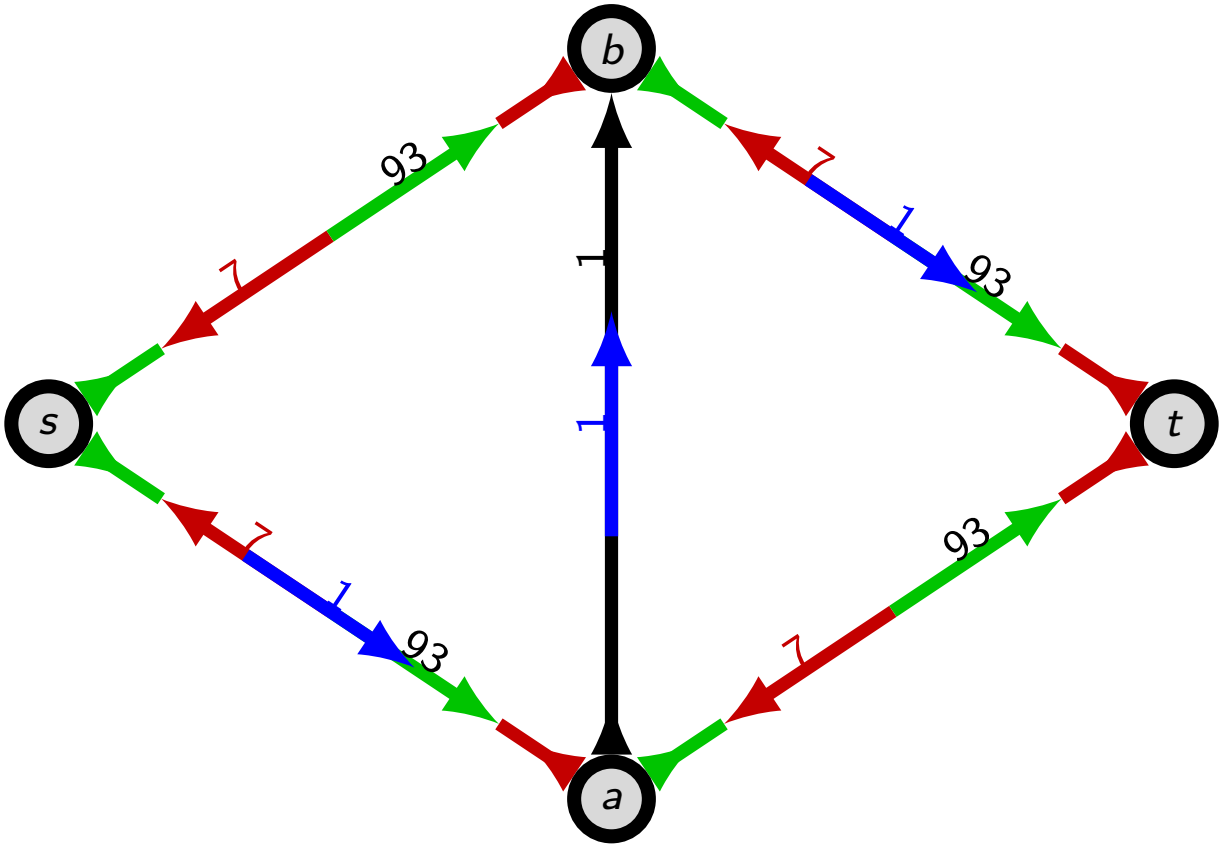
Beispiel

$n = |V|, m = |E|$



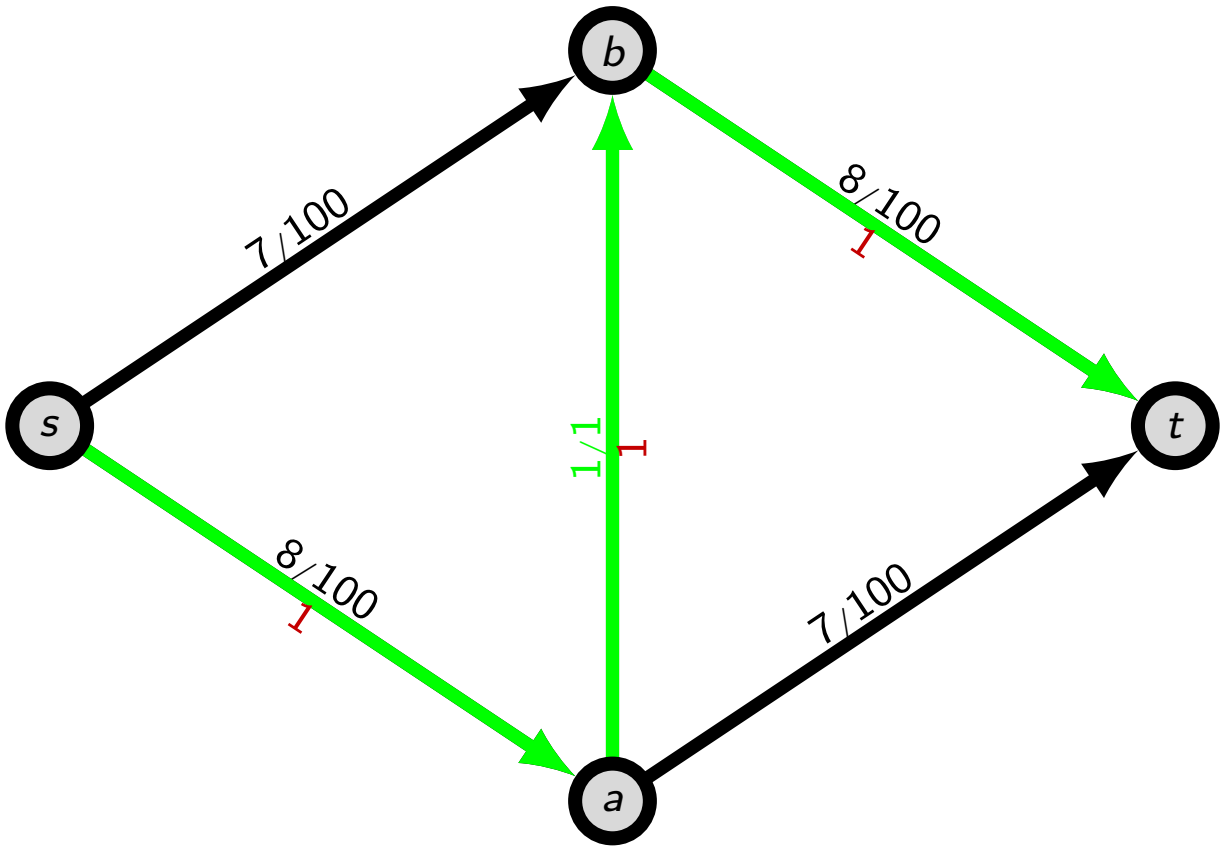
Beispiel

$n = |V|, m = |E|$



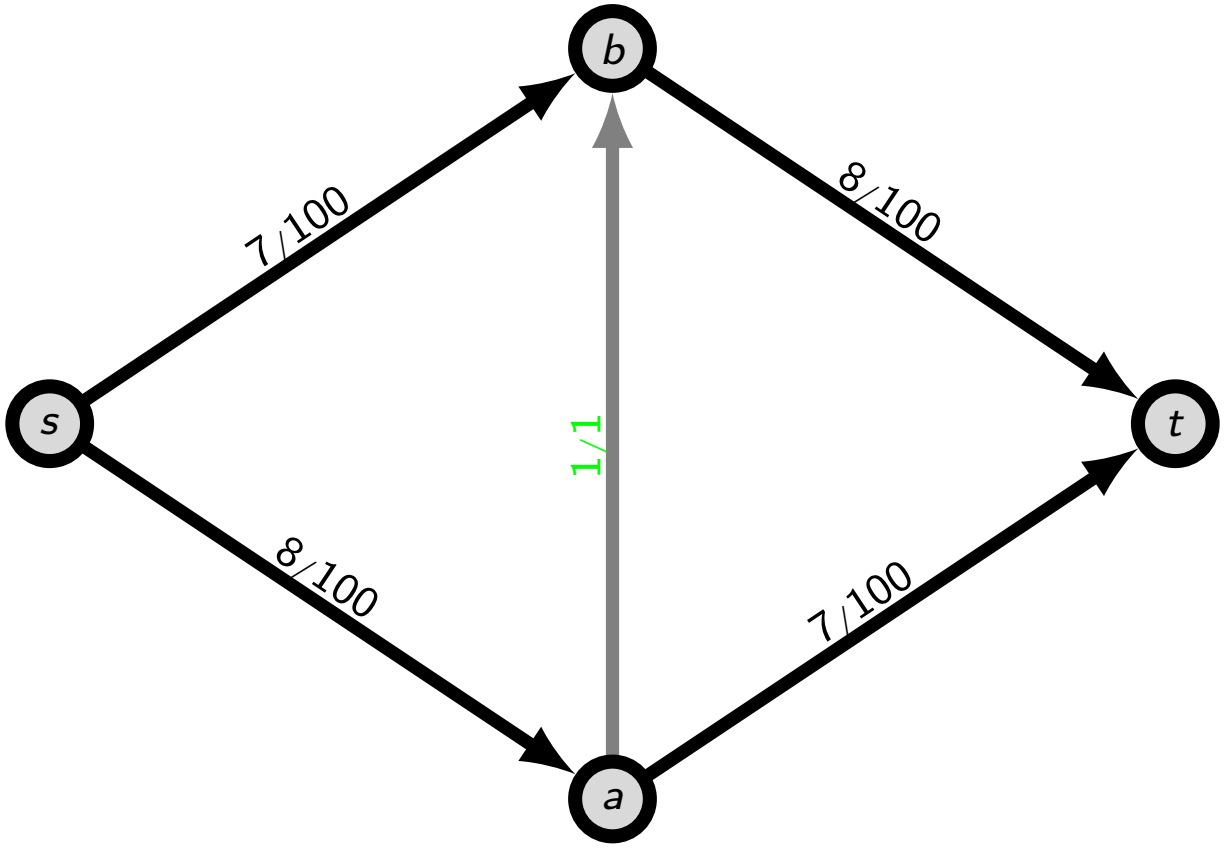
Beispiel

$n = |V|, m = |E|$



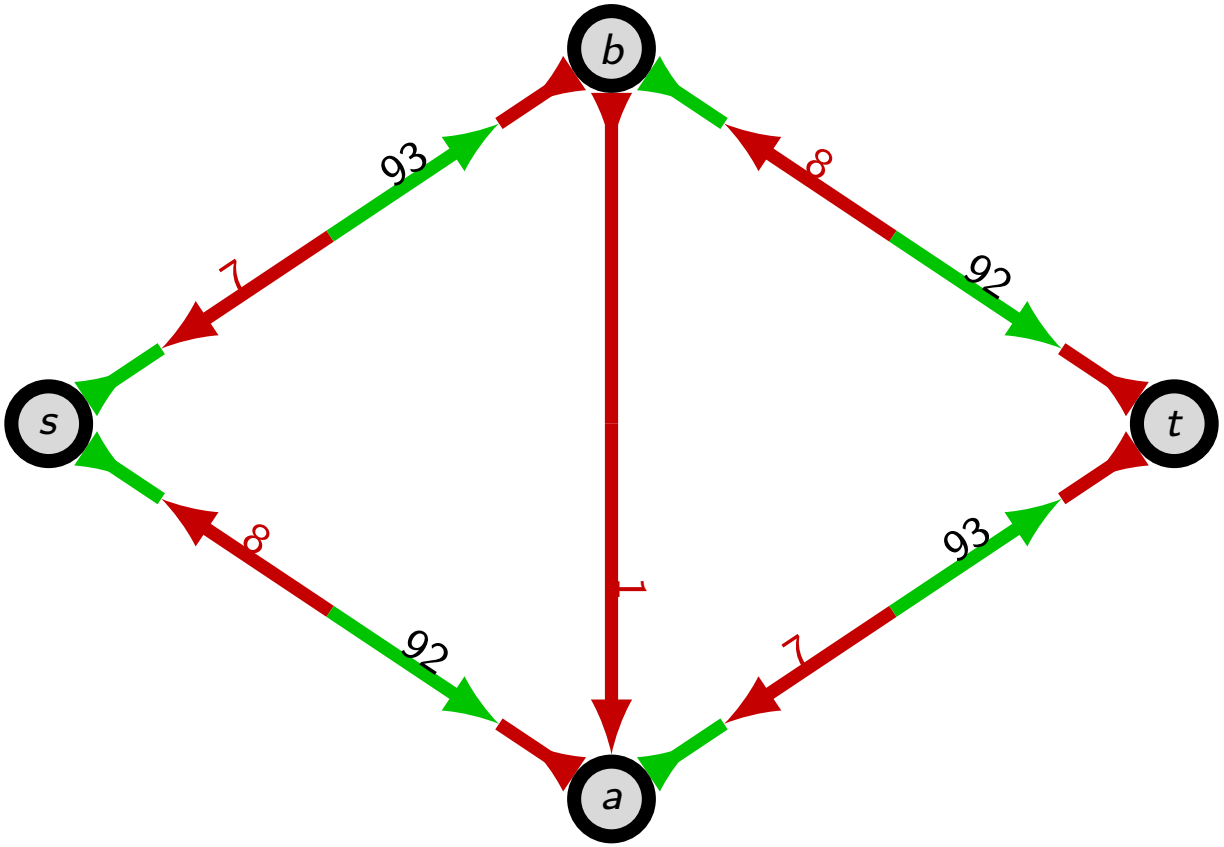
Beispiel

$n = |V|, m = |E|$



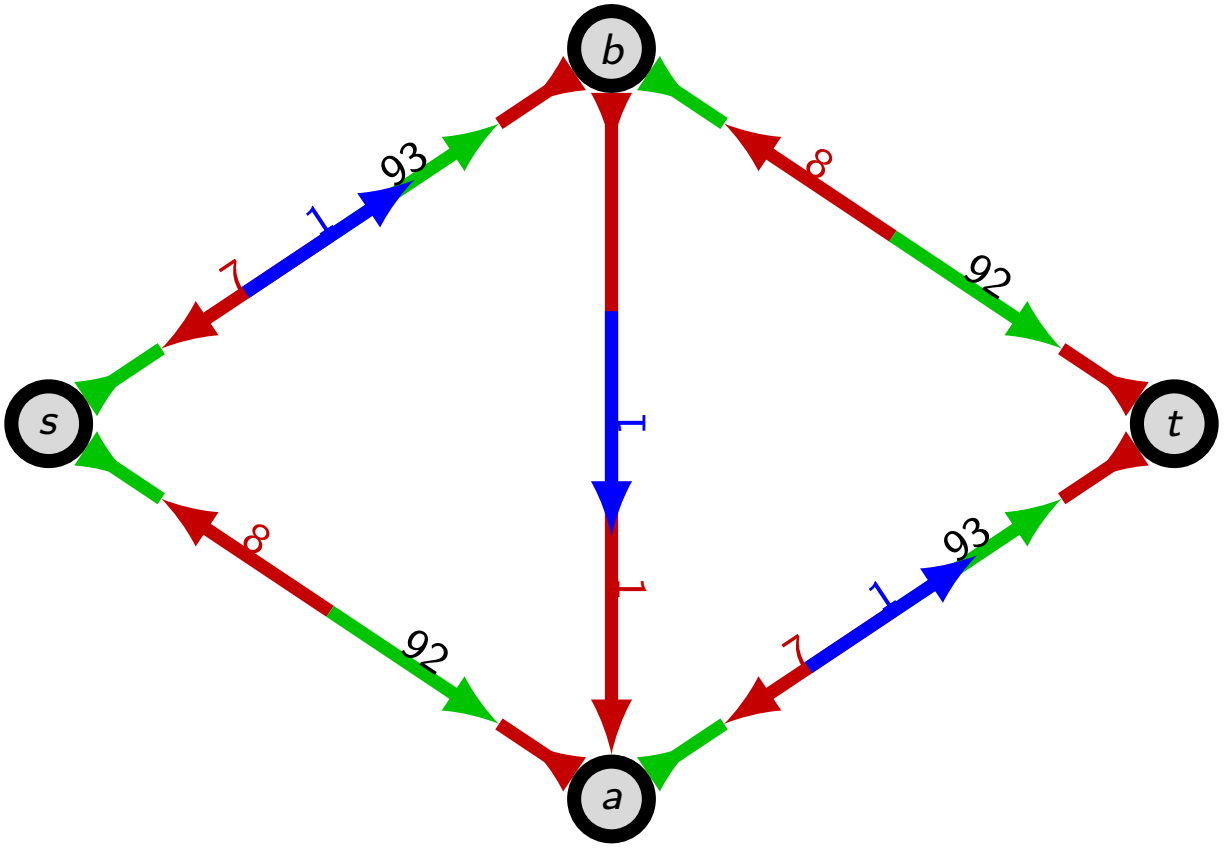
Beispiel

$n = |V|, m = |E|$



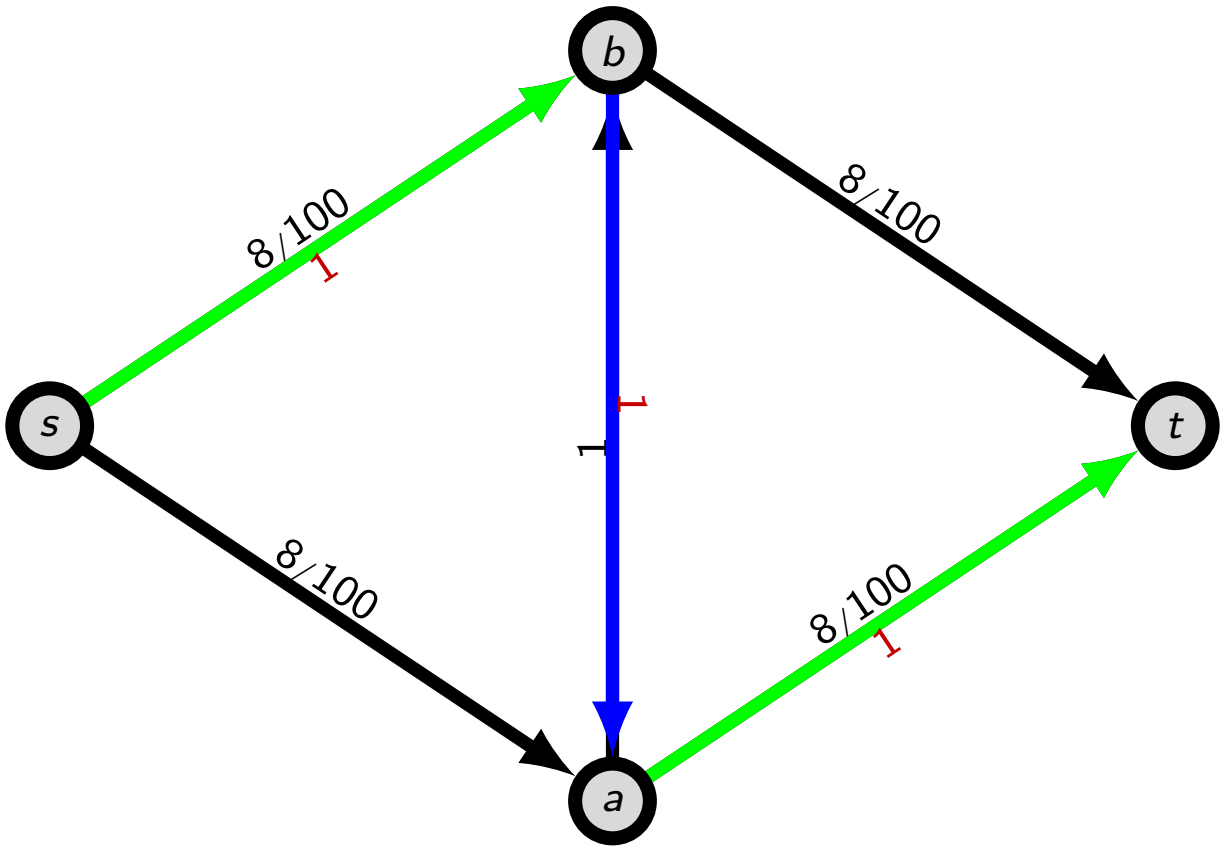
Beispiel

$n = |V|, m = |E|$



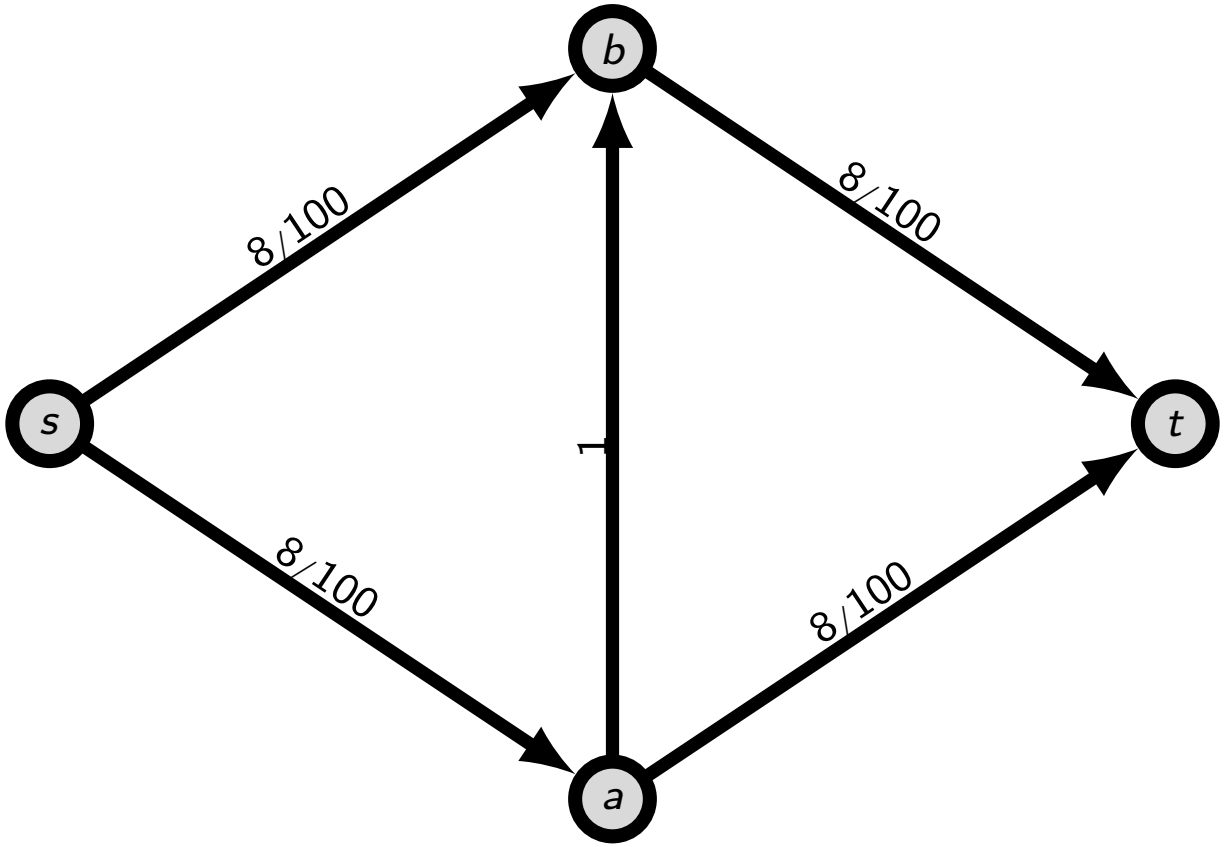
Beispiel

$n = |V|, m = |E|$



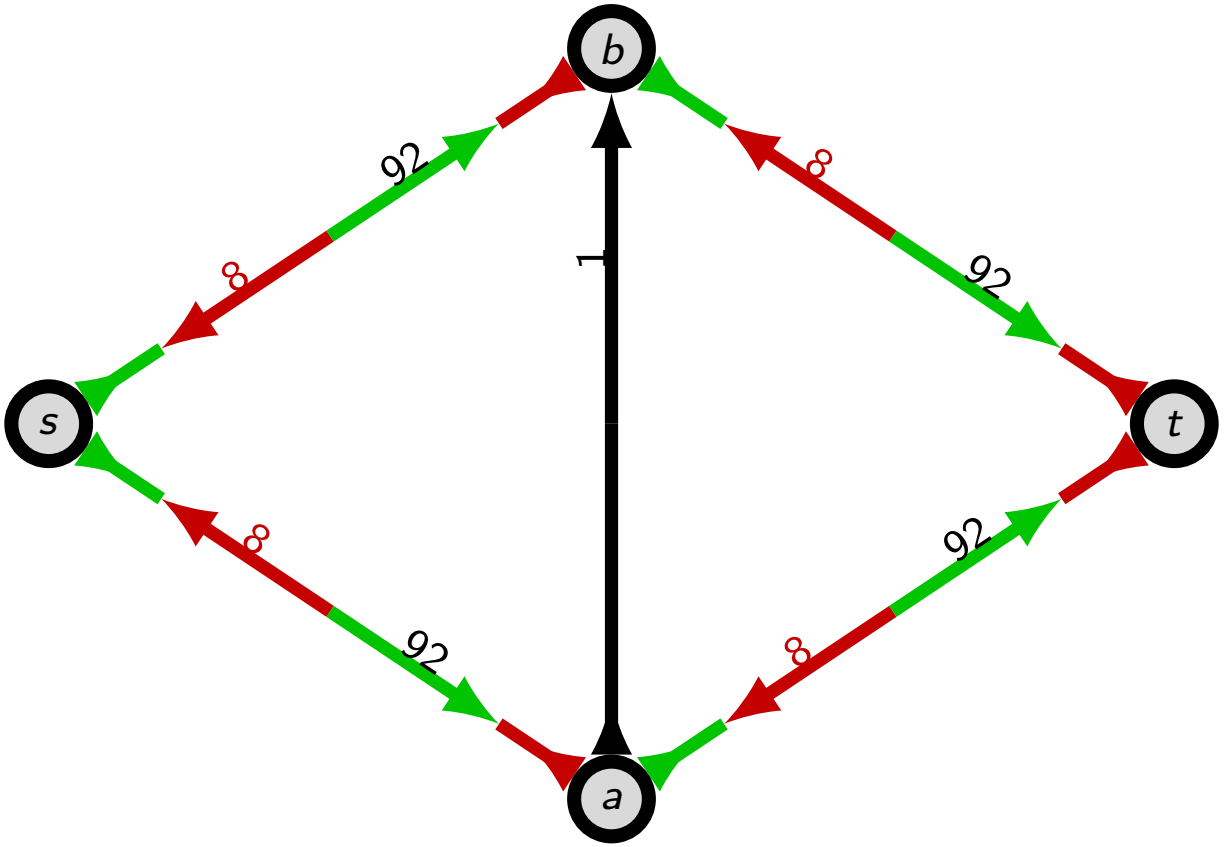
Beispiel

$n = |V|, m = |E|$



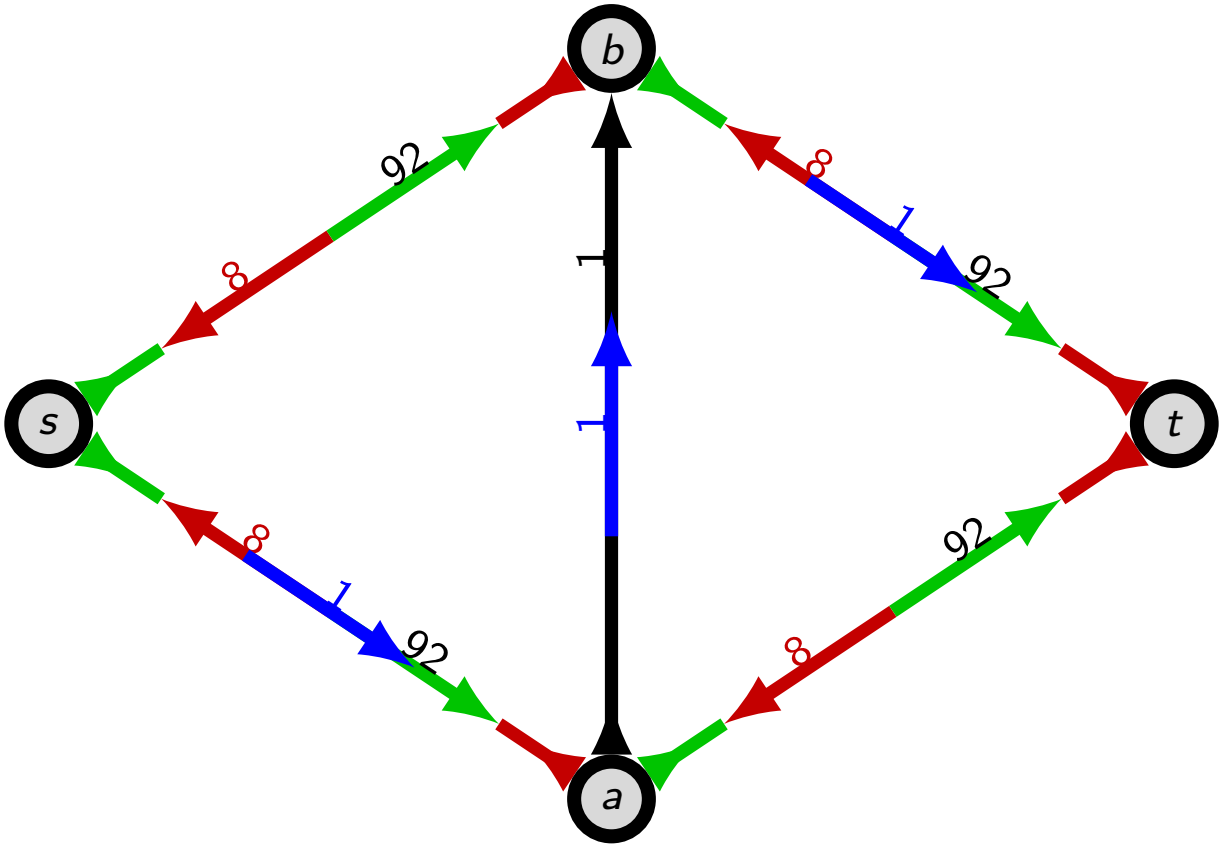
Beispiel

$n = |V|, m = |E|$



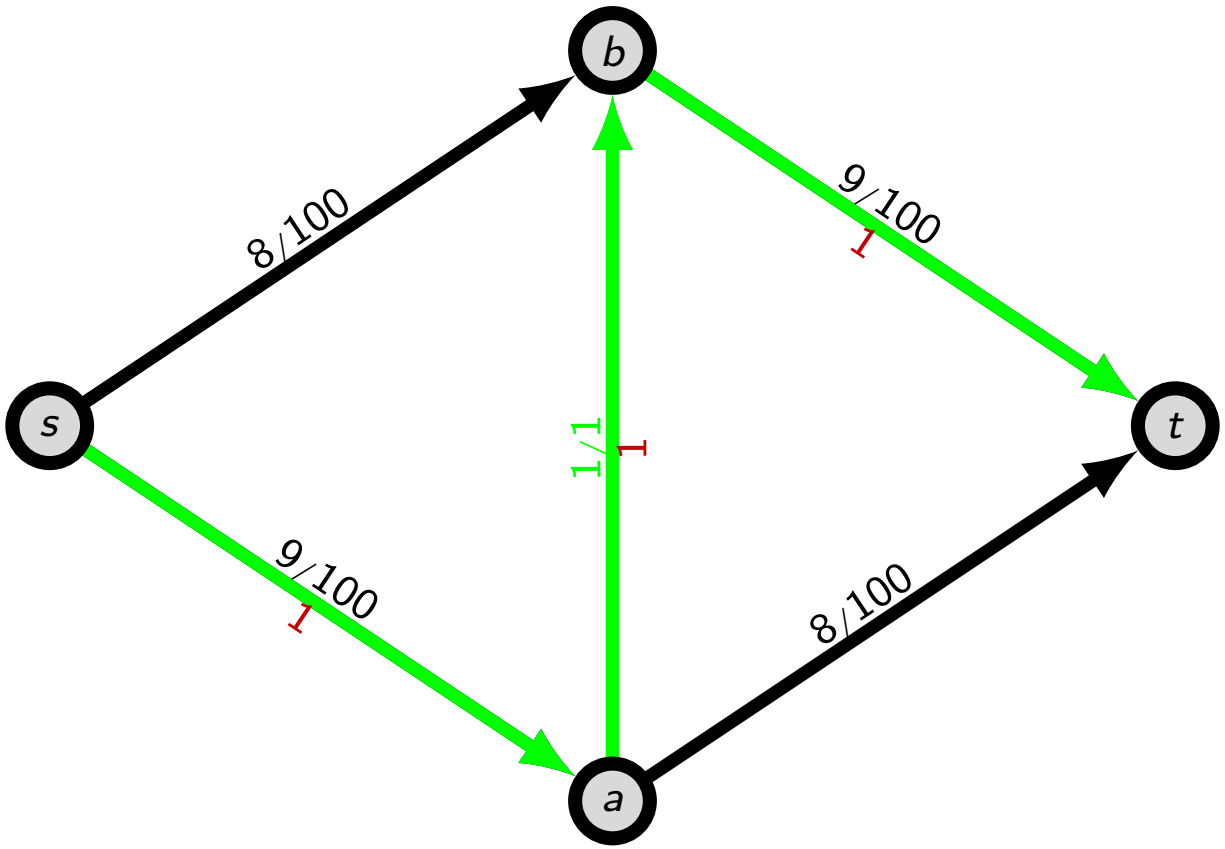
Beispiel

$n = |V|, m = |E|$



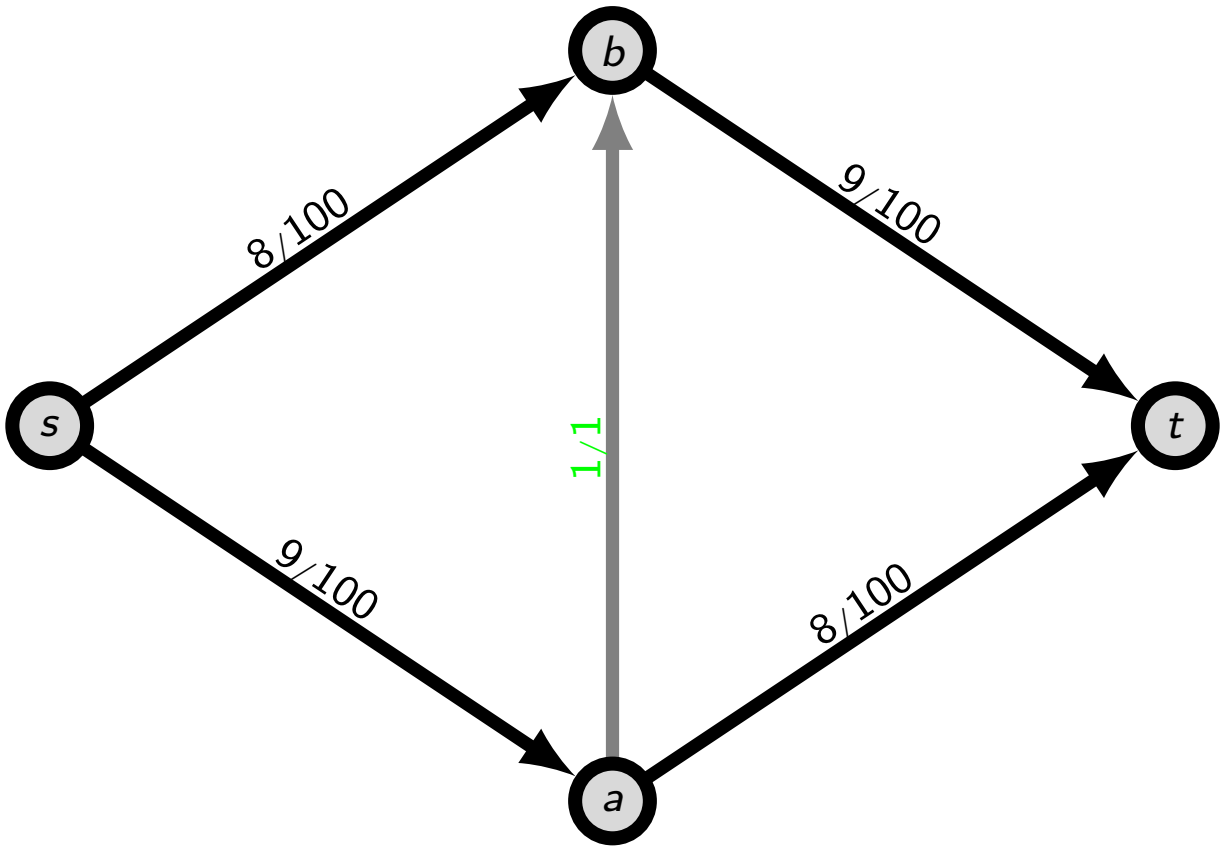
Beispiel

$n = |V|, m = |E|$



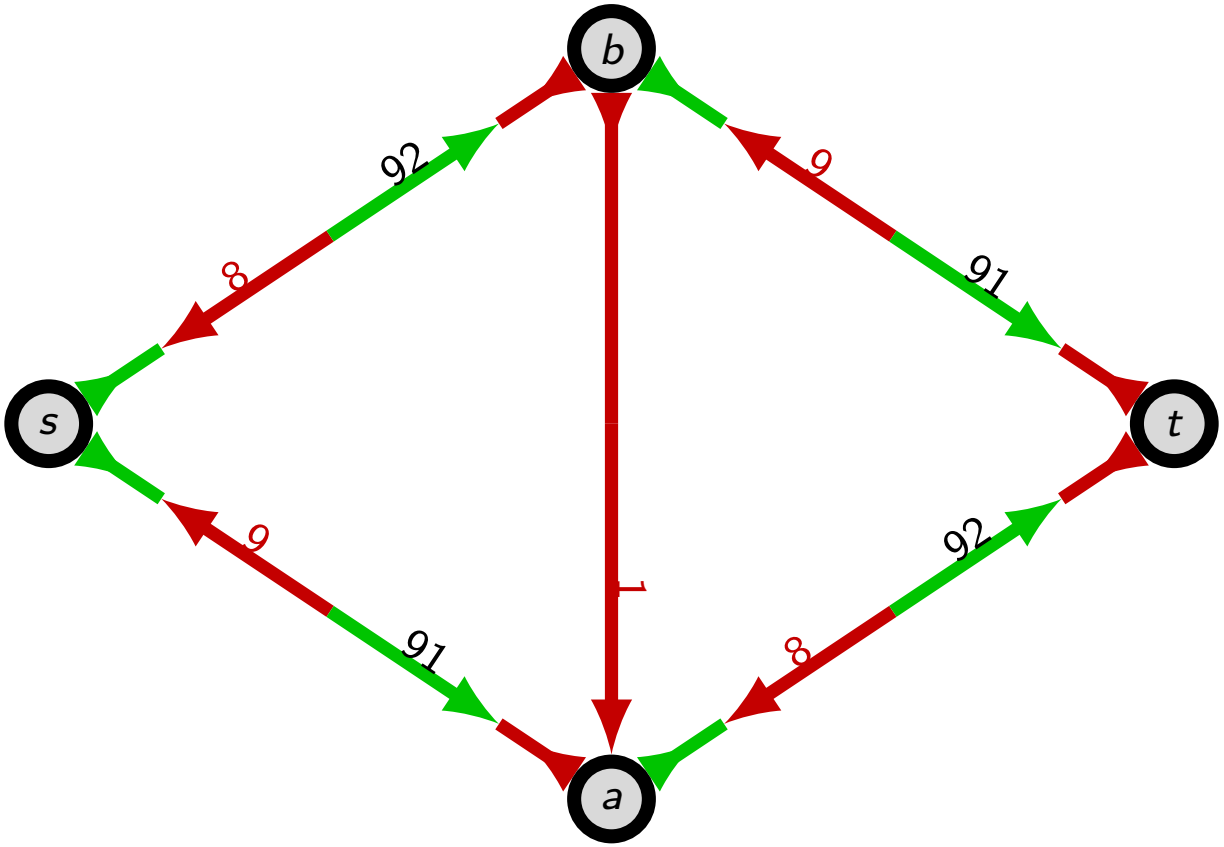
Beispiel

$n = |V|, m = |E|$



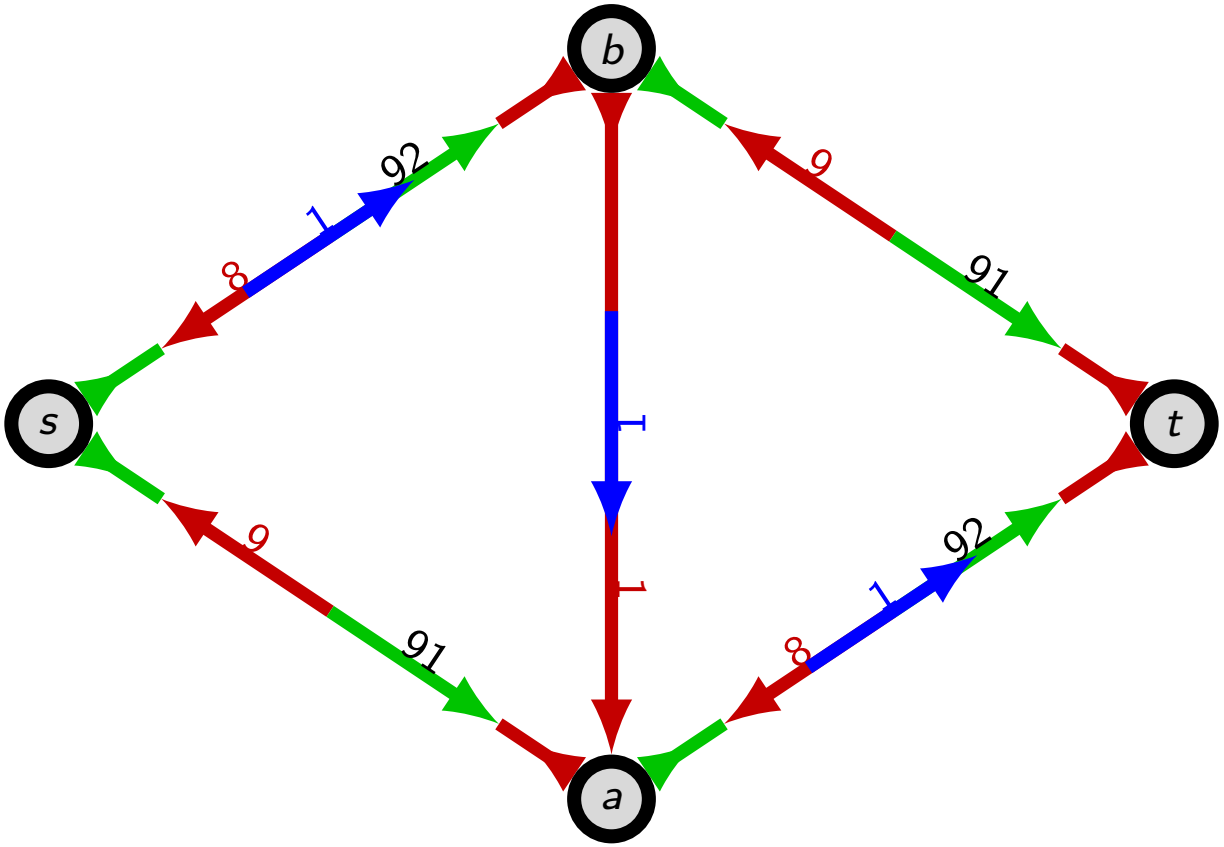
Beispiel

$n = |V|, m = |E|$



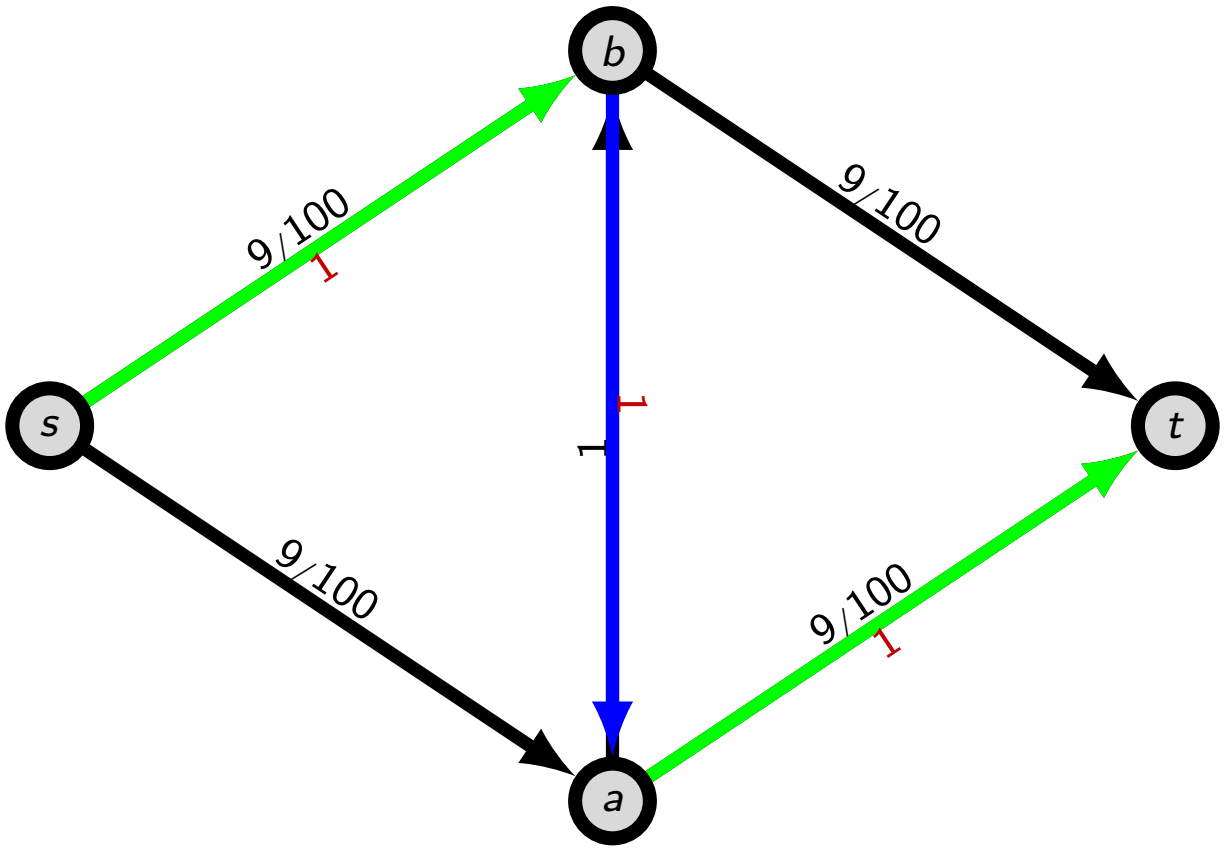
Beispiel

$n = |V|, m = |E|$



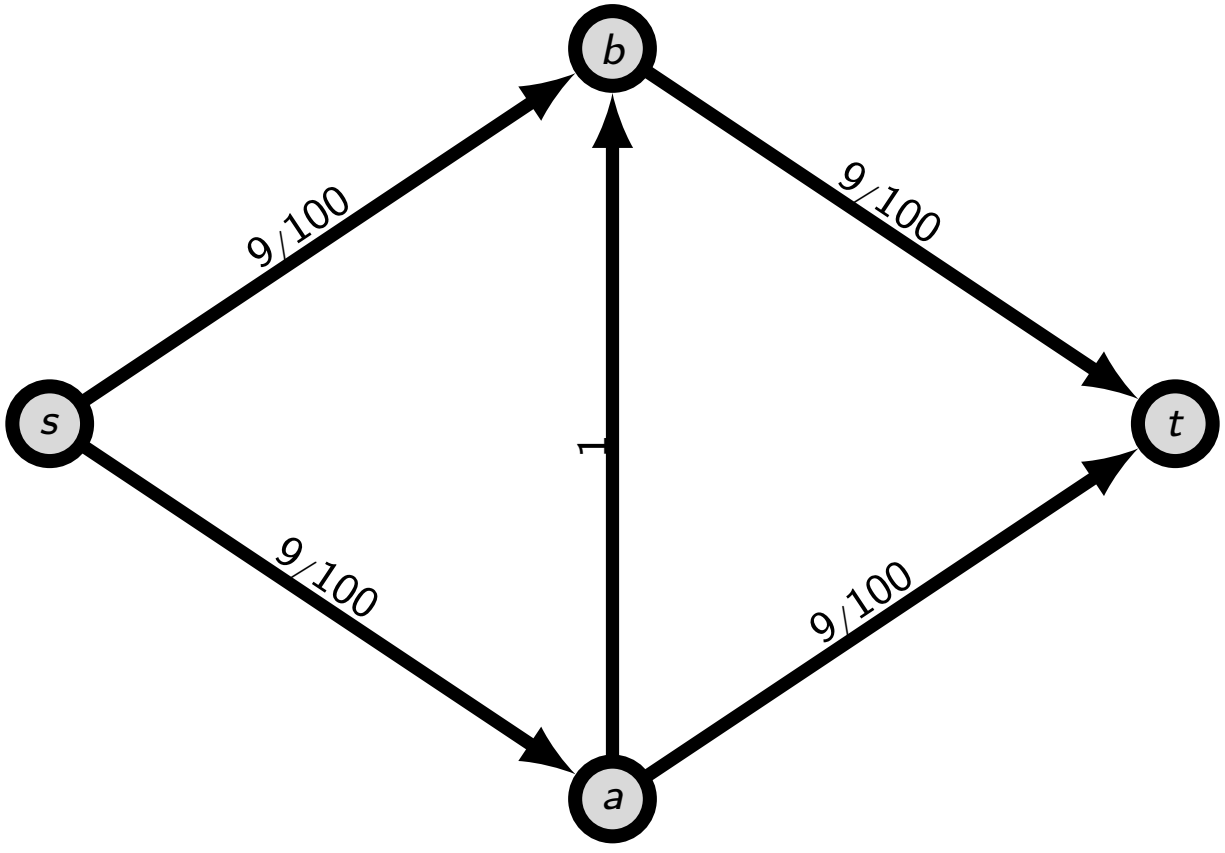
Beispiel

$n = |V|, m = |E|$



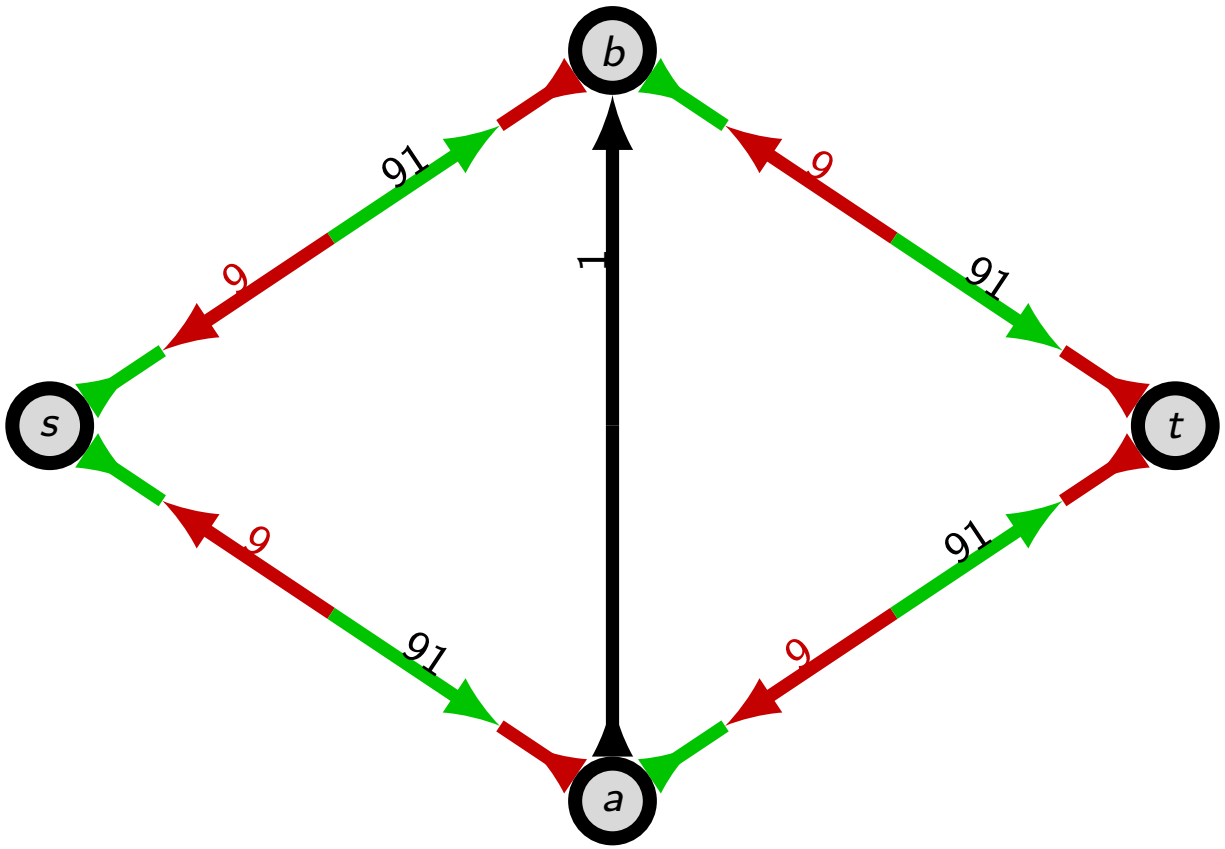
Beispiel

$n = |V|, m = |E|$



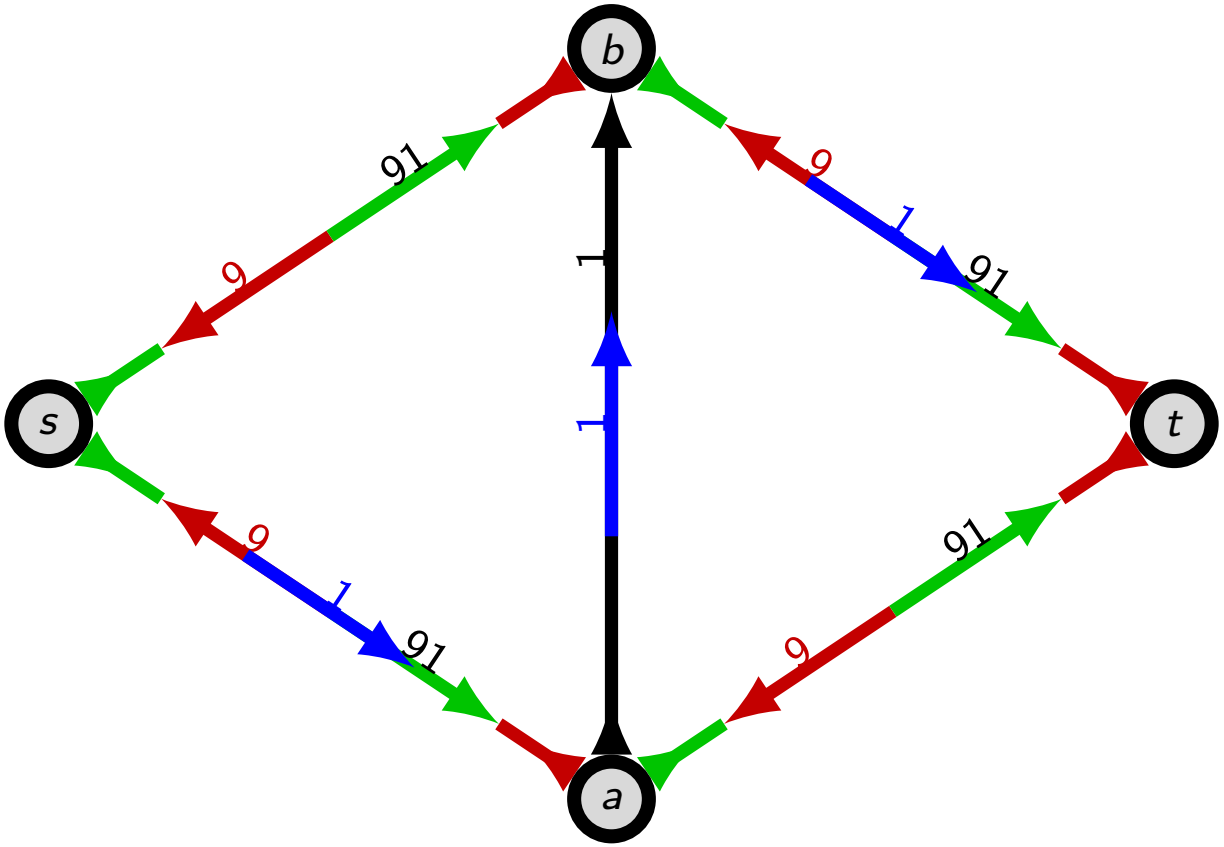
Beispiel

$n = |V|, m = |E|$



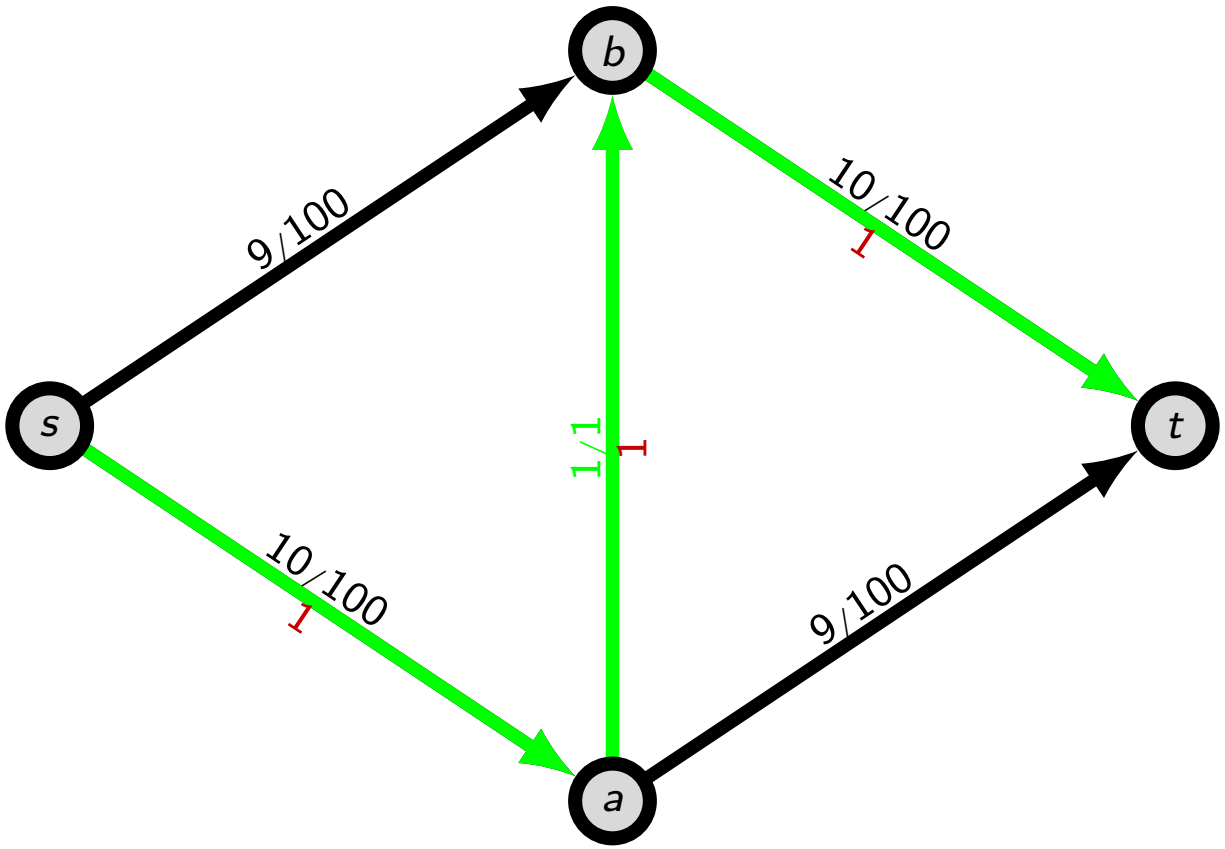
Beispiel

$n = |V|, m = |E|$



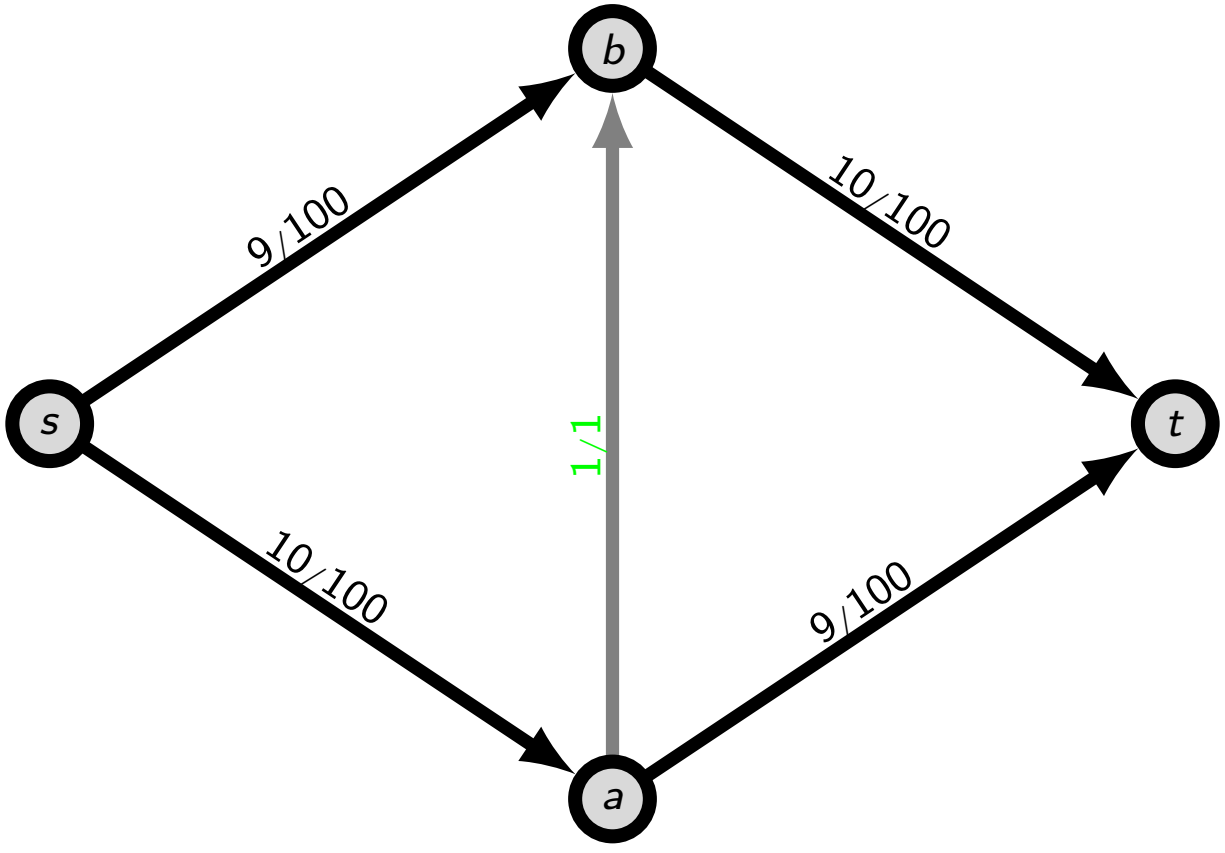
Beispiel

$n = |V|, m = |E|$



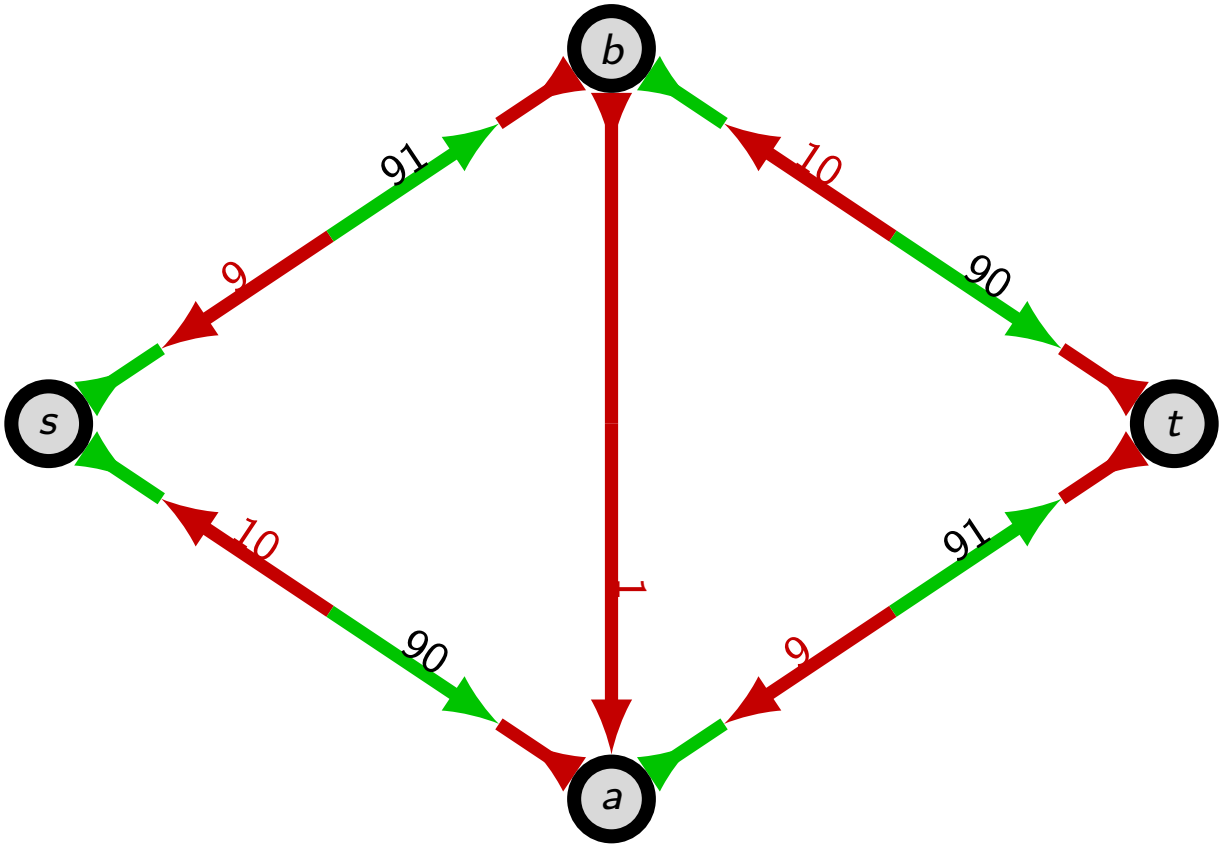
Beispiel

$n = |V|, m = |E|$



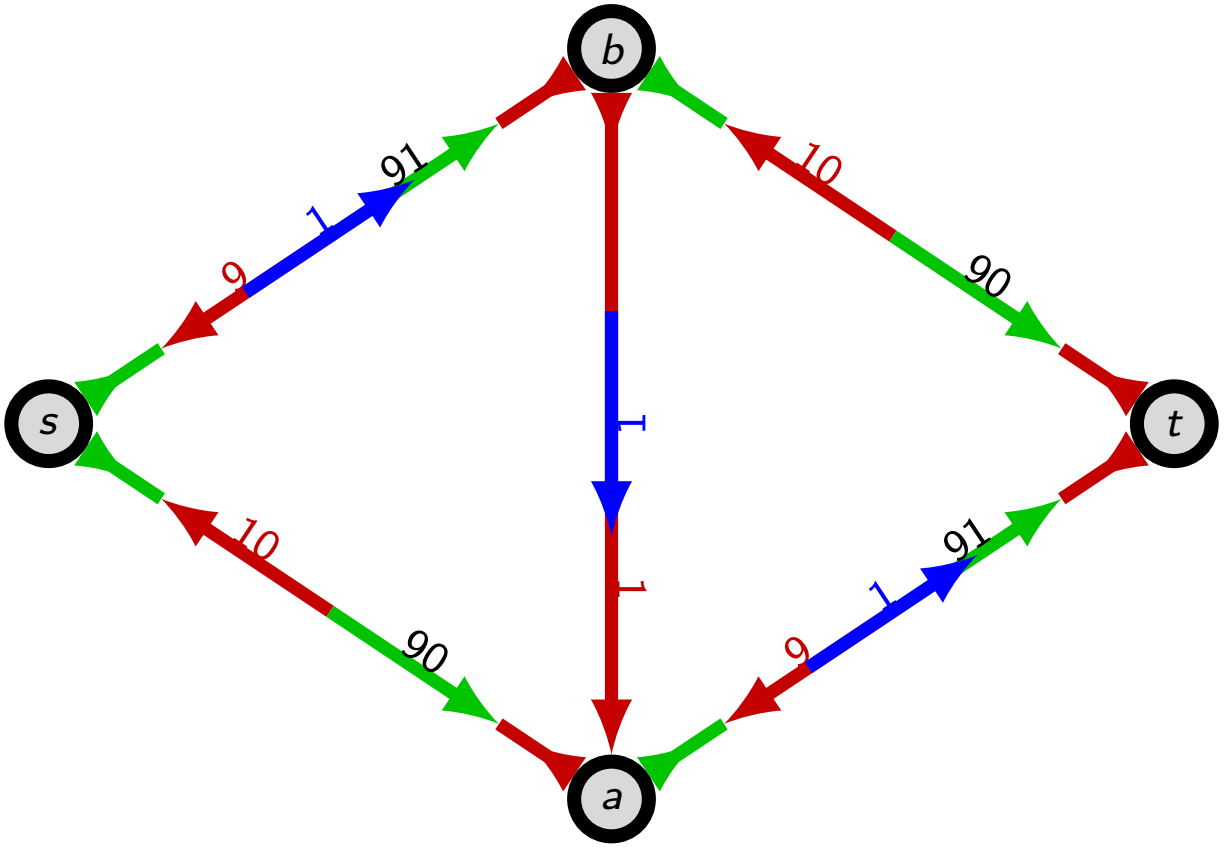
Beispiel

$n = |V|, m = |E|$



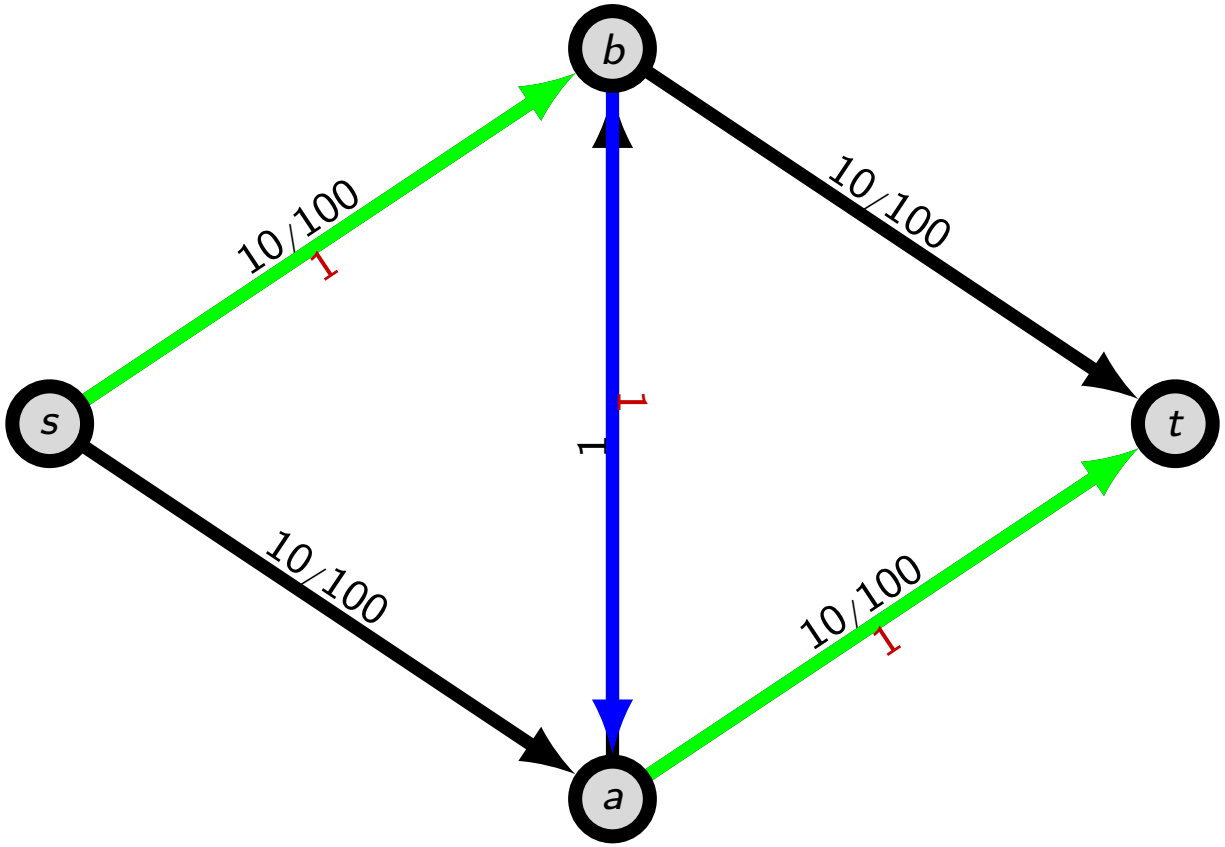
Beispiel

$n = |V|, m = |E|$



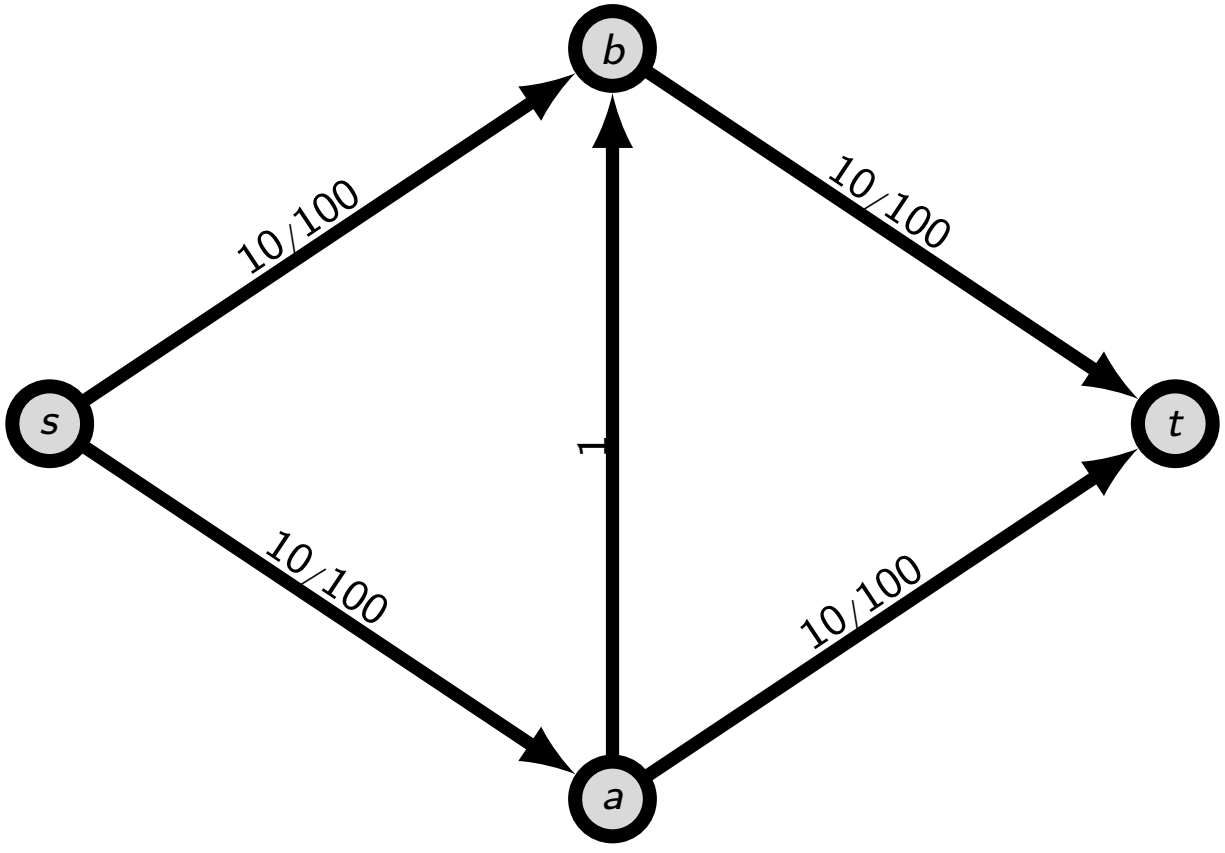
Beispiel

$n = |V|, m = |E|$



Beispiel

$n = |V|, m = |E|$

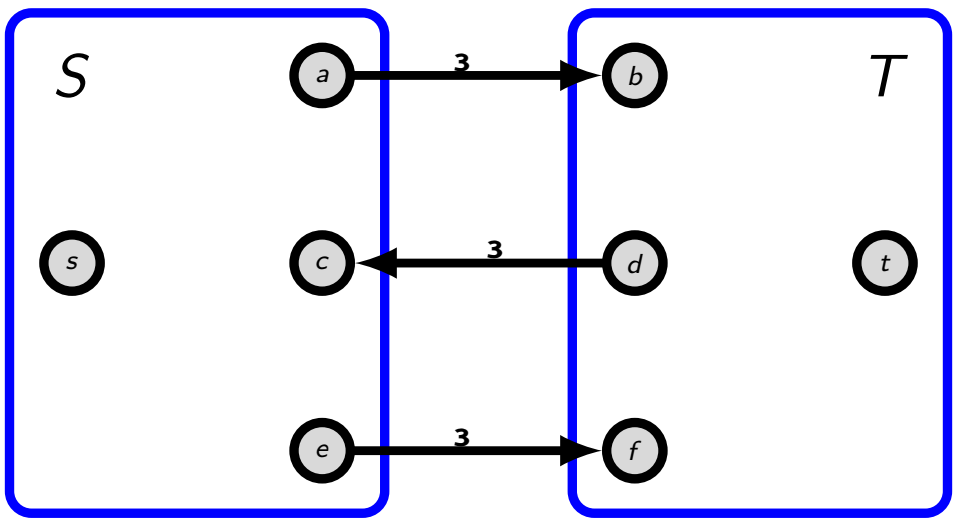


Ein Algorithmus würde nicht die Gedult verlieren.

Rückblick

$n = |V|, m = |E|$

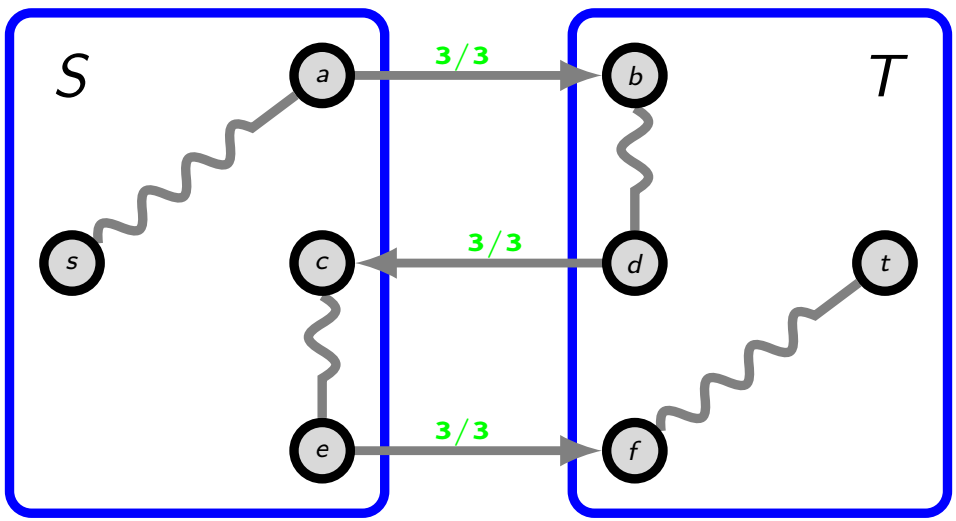
- Sei (S, T) der minimale Schnitt.



Rückblick

$n = |V|, m = |E|$

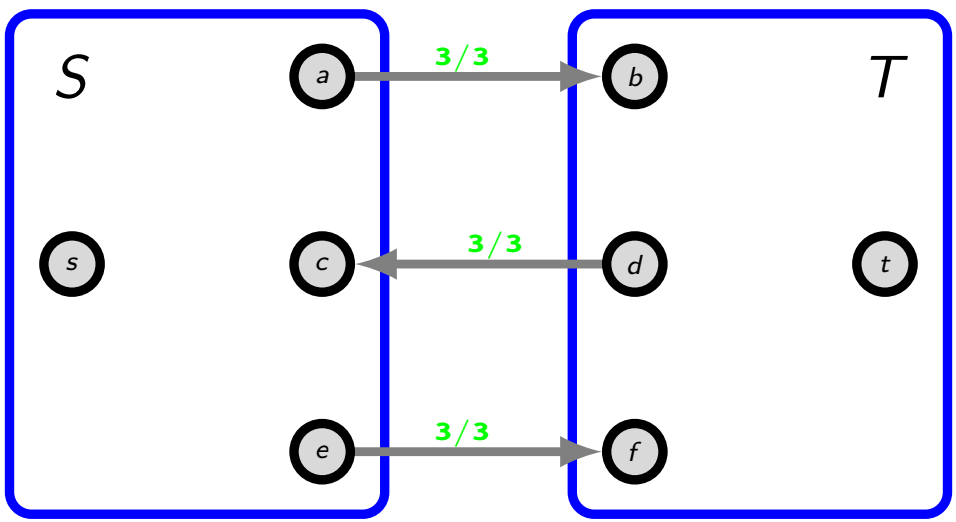
- Sei (S, T) der minimale Schnitt.
- Der Algorithmus kann Kanten von T nach S verwenden.



Rückblick

$n = |V|, m = |E|$

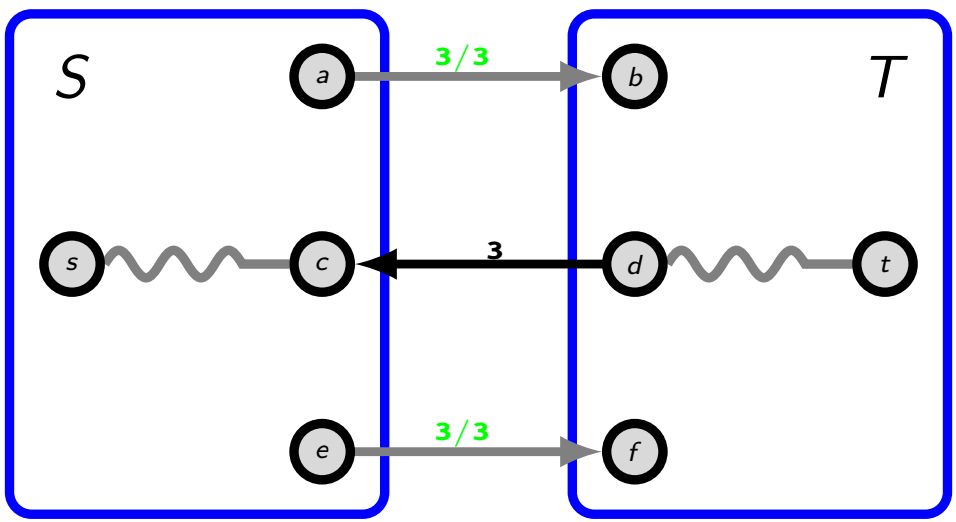
- Sei (S, T) der minimale Schnitt.
- Der Algorithmus kann Kanten von T nach S verwenden.
- Damit ist der aktuelle Fluss über den Schnitt kleiner als die Kapazität des minimalen Schnittes.



Rückblick

$n = |V|, m = |E|$

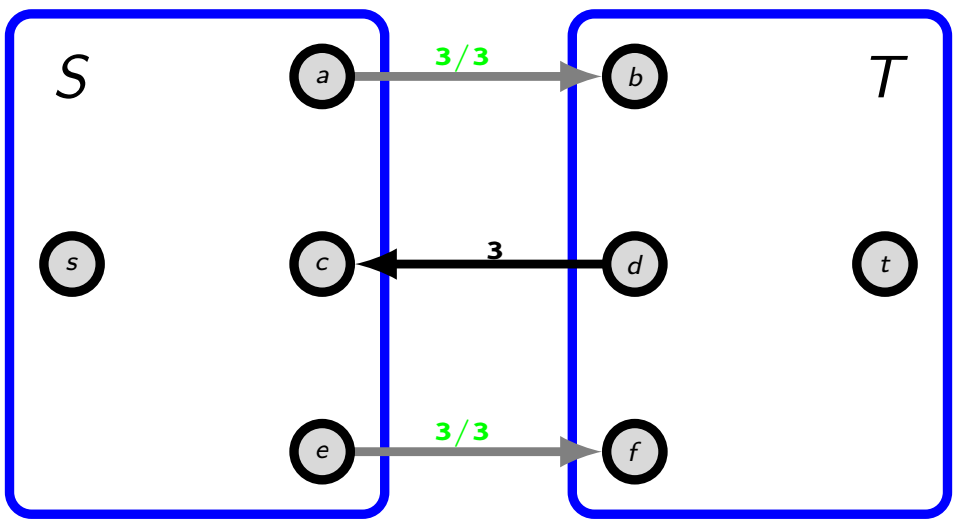
- Sei (S, T) der minimale Schnitt.
- Der Algorithmus kann Kanten von T nach S verwenden.
- Damit ist der aktuelle Fluss über den Schnitt kleiner als die Kapazität des minimalen Schnittes.
- Durch Rückwärtskanten werden diese "Fehler" behoben.



Rückblick

$n = |V|, m = |E|$

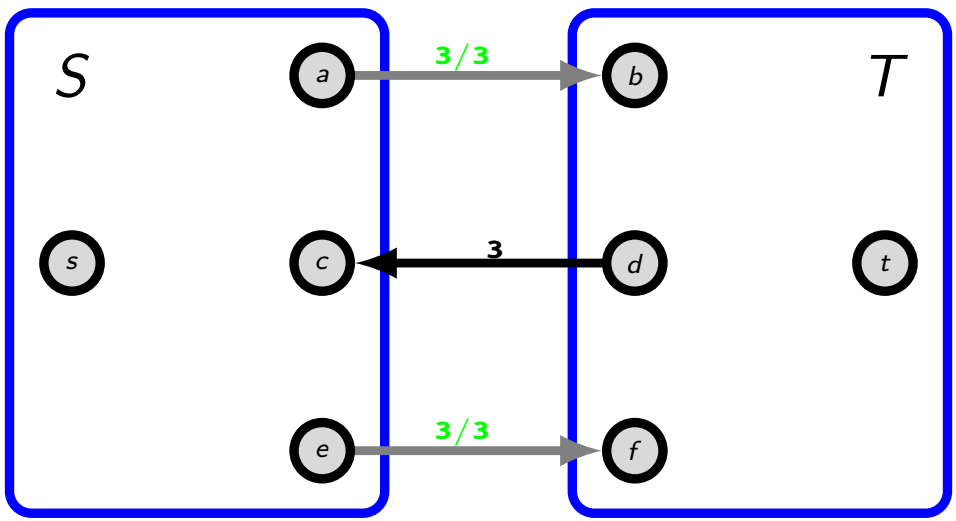
- Sei (S, T) der minimale Schnitt.
- Der Algorithmus kann Kanten von T nach S verwenden.
- Damit ist der aktuelle Fluss über den Schnitt kleiner als die Kapazität des minimalen Schnittes.
- Durch Rückwärtskanten werden diese "Fehler" behoben.
- Damit findet der Algorithmus den minimalen Schnitt.



Rückblick

$n = |V|, m = |E|$

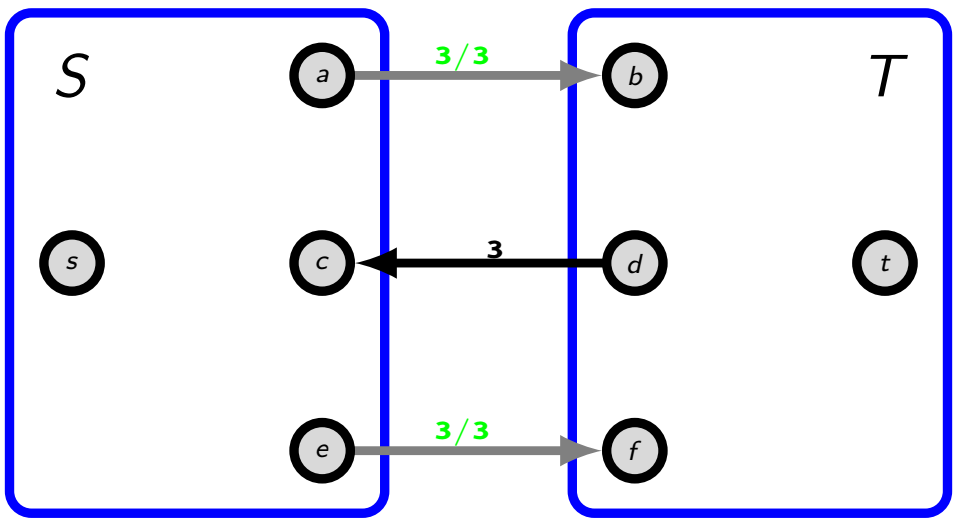
- Sei (S, T) der minimale Schnitt.
- Der Algorithmus kann Kanten von T nach S verwenden.
- Damit ist der aktuelle Fluss über den Schnitt kleiner als die Kapazität des minimalen Schnittes.
- Durch Rückwärtskanten werden diese "Fehler" behoben.
- Damit findet der Algorithmus den minimalen Schnitt.
- Und damit kann das Min-Cut Max-Flow Theorem bewiesen werden.



Rückblick

$n = |V|, m = |E|$

- Sei (S, T) der minimale Schnitt.
- Der Algorithmus kann Kanten von T nach S verwenden.
- Damit ist der aktuelle Fluss über den Schnitt kleiner als die Kapazität des minimalen Schnittes.
- Durch Rückwärtskanten werden diese "Fehler" behoben.
- Damit findet der Algorithmus den minimalen Schnitt.
- Und damit kann das Min-Cut Max-Flow Theorem bewiesen werden.



Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) *f ist ein maximaler Fluss.*

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) *f ist ein maximaler Fluss.*

b) *Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.*

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) *f ist ein maximaler Fluss.*

b) *Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.*

c) *Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.*

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) *f ist ein maximaler Fluss.*

b) *Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.*

c) *Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.*

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

Beweis:

- Zeige c) $\Longrightarrow$ a).

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

Beweis:

- Zeige c) $\Longrightarrow$ a).
- Zeige a) $\Longrightarrow$ b).

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

Beweis:

- Zeige c) $\Longrightarrow$ a).
- Zeige a) $\Longrightarrow$ b).
- Zeige b) $\Longrightarrow$ c).

Min-Cut Max-Flow

$n = |V|, m = |E|$

Theorem (Minimum Cut Maximum Flow)

Sei f ein Fluss auf einem Netzwerk $G = (V, E, s, t, c)$.
Dann sind die folgenden Aussagen äquivalent:

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

Beweis:

- Zeige c) $\Longrightarrow$ a).
- Zeige a) $\Longrightarrow$ b).
- Zeige b) $\Longrightarrow$ c).

Zeige c) $\Longrightarrow$ a)

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

Zeige c) $\Longrightarrow$ a)

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

a) f ist ein maximaler Fluss.

Zeige c) $\Longrightarrow$ a)

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

a) f ist ein maximaler Fluss.

- Wegen Lemma (Max-Flow $\leqslant$ Min-Cut) gilt:

Zeige c) $\Longrightarrow$ a)

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

a) f ist ein maximaler Fluss.

- Wegen Lemma (Max-Flow $\leqslant$ Min-Cut) gilt:
- $w(f) = c(S, T) \geqslant$ Min-Cut $\geqslant$ Max-Flow

Zeige c) $\Longrightarrow$ a)

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

a) f ist ein maximaler Fluss.

- Wegen Lemma (Max-Flow $\leqslant$ Min-Cut) gilt:
- $w(f) = c(S, T) \geqslant$ Min-Cut $\geqslant$ Max-Flow
- Damit muss f ein maximaler Fluss sein.

Zeige c) $\Longrightarrow$ a)

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

a) f ist ein maximaler Fluss.

- Wegen Lemma (Max-Flow $\leqslant$ Min-Cut) gilt:
- $w(f) = c(S, T) \geqslant$ Min-Cut $\geqslant$ Max-Flow
- Damit muss f ein maximaler Fluss sein.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

- Beweis durch Widerspruch.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

- Beweis durch Widerspruch.
- Sei f maximaler Fluss.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

- Beweis durch Widerspruch.
- Sei f maximaler Fluss.
- Sei P ein vergrößernder Weg von s nach t in G_f mit Fluss p.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

- Beweis durch Widerspruch.
- Sei f maximaler Fluss.
- Sei P ein vergrößernder Weg von s nach t in G_f mit Fluss p.
- Dann kann f um p vergrößert werden.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

- Beweis durch Widerspruch.
- Sei f maximaler Fluss.
- Sei P ein vergrößernder Weg von s nach t in G_f mit Fluss p.
- Dann kann f um p vergrößert werden.
- f ist daher nicht maximal.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

- Beweis durch Widerspruch.
- Sei f maximaler Fluss.
- Sei P ein vergrößernder Weg von s nach t in G_f mit Fluss p.
- Dann kann f um p vergrößert werden.
- f ist daher nicht maximal.
- Widerspruch.

Zeige a) $\Longrightarrow$ b)

a) f ist ein maximaler Fluss.

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

- Beweis durch Widerspruch.
- Sei f maximaler Fluss.
- Sei P ein vergrößernder Weg von s nach t in G_f mit Fluss p.
- Dann kann f um p vergrößert werden.
- f ist daher nicht maximal.
- Widerspruch.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

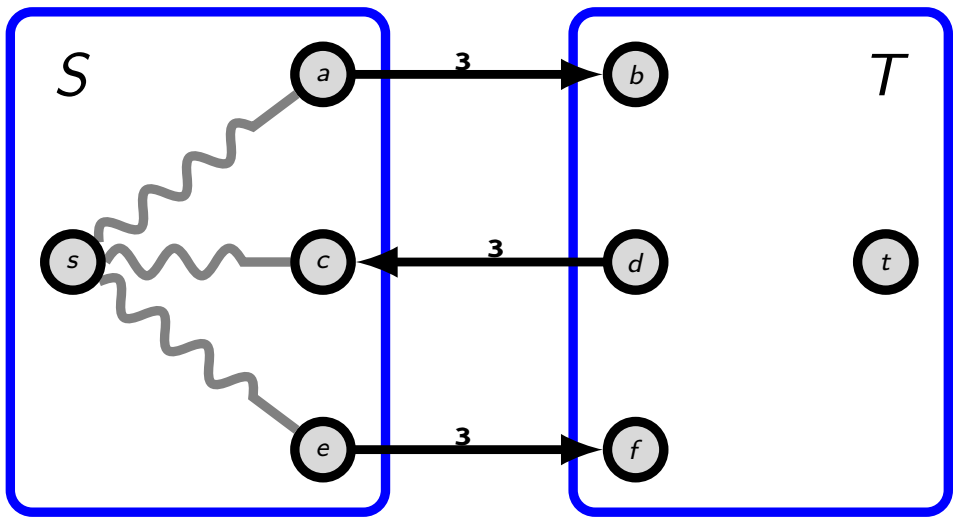
- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.
- Da es keinen vergrößernden Weg in G_f gibt, gilt: $t \in T$.

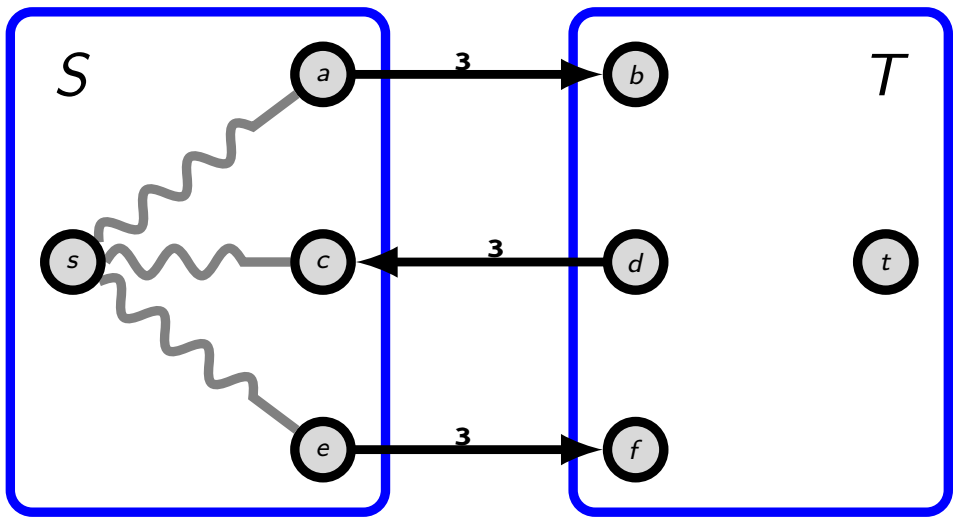


Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.
- Da es keinen vergrößernden Weg in G_f gibt, gilt: $t \in T$.



Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.
- Da es keinen vergrößernden Weg in G_f gibt, gilt: $t \in T$.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.
- Da es keinen vergrößernden Weg in G_f gibt, gilt: $t \in T$.
- $\forall (v, w) \in E \cap (S, T) : f(v, w) = c(v, w)$, denn $\text{rest}_f(v, w) = 0$.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.
- Da es keinen vergrößernden Weg in G_f gibt, gilt: $t \in T$.
- $\forall (v, w) \in E \cap (S, T) : f(v, w) = c(v, w)$, denn $\text{rest}_f(v, w) = 0$.
- $\forall (v, w) \in E \cap (T, S) : f(v, w) = 0$, denn $\text{rest}_f(v, w) = 0$.

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.
- Da es keinen vergrößernden Weg in G_f gibt, gilt: $t \in T$.
- $\forall (v, w) \in E \cap (S, T) : f(v, w) = c(v, w)$, denn $\text{rest}_f(v, w) = 0$.
- $\forall (v, w) \in E \cap (T, S) : f(v, w) = 0$, denn $\text{rest}_f(v, w) = 0$.
- Nach Definition von $c(S, T)$ und $f(S, T)$ gilt:

$$\begin{aligned} c(S,T) &= \sum_{(v,w)\in E, v\in S, w\in T} c(v,w) \\ &= \sum_{(v,w)\in E, v\in S, w\in T} f(v,w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w,v) \\ &= f(S,T) \\ &= w(f) \end{aligned}$$

Zeige b) $\Longrightarrow$ c)

b) Das Restnetzwerk G_f enthält keinen vergrößernden Weg von s nach t.

c) Es gibt einen Schnitt (S, T) mit: $w(f) = c(S, T)$.

- Setze $S = \{v \in V \mid \exists \text{ Weg von } s \text{ nach } v \text{ in } G_f\}$
- Setze $T = V \setminus S$.
- Da es keinen vergrößernden Weg in G_f gibt, gilt: $t \in T$.
- $\forall (v, w) \in E \cap (S, T) : f(v, w) = c(v, w)$, denn $\text{rest}_f(v, w) = 0$.
- $\forall (v, w) \in E \cap (T, S) : f(v, w) = 0$, denn $\text{rest}_f(v, w) = 0$.
- Nach Definition von $c(S, T)$ und $f(S, T)$ gilt:

$$\begin{aligned} c(S, T) &= \sum_{(v,w)\in E, v\in S, w\in T} c(v, w) \\ &= \sum_{(v,w)\in E, v\in S, w\in T} f(v, w) - \sum_{(w,v)\in E, v\in S, w\in T} f(w, v) \\ &= f(S, T) \\ &= w(f) \end{aligned}$$

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

- Zeige: es reichen $O(m \cdot n)$ Iterationen.

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

- Zeige: es reichen $O(m \cdot n)$ Iterationen.
- Durch einen flussvergrößernden Weg werden Kanten gelöscht und ggf. hinzugefügt.

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

- Zeige: es reichen $O(m \cdot n)$ Iterationen.
- Durch einen flussvergrößernden Weg werden Kanten gelöscht und ggf. hinzugefügt.
- Sei nun G_f^i das Restnetzwerk G_f, welches in Iteration i verwendet wird.

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

- Zeige: es reichen $O(m \cdot n)$ Iterationen.
- Durch einen flussvergrößernden Weg werden Kanten gelöscht und ggf. hinzugefügt.
- Sei nun G_f^i das Restnetzwerk G_f, welches in Iteration i verwendet wird.
- Zeige: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

- Zeige: es reichen $O(m \cdot n)$ Iterationen.
- Durch einen flussvergrößernden Weg werden Kanten gelöscht und ggf. hinzugefügt.
- Sei nun G_f^i das Restnetzwerk G_f, welches in Iteration i verwendet wird.
- Zeige: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Dies gilt sicher für Kanten, die in G_f^i gelöscht wurden.

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

- Zeige: es reichen $O(m \cdot n)$ Iterationen.
- Durch einen flussvergrößernden Weg werden Kanten gelöscht und ggf. hinzugefügt.
- Sei nun G_f^i das Restnetzwerk G_f, welches in Iteration i verwendet wird.
- Zeige: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Dies gilt sicher für Kanten, die in G_f^i gelöscht wurden.
- Die hinzugefügten Kanten von G_f^i zu G_f^{i+1} werden auf der folgenden Folie behandelt.

Ford-Fulkerson mit BFS

$n = |V|, m = |E|$

Theorem (Edmonds und Karp, 1969)

Die Laufzeit der Ford-Fulkerson Methode mit BFS ist $O(m^2 \cdot n) = O(n^5)$.

Beweis:

- Zeige: es reichen $O(m \cdot n)$ Iterationen.
- Durch einen flussvergrößernden Weg werden Kanten gelöscht und ggf. hinzugefügt.
- Sei nun G_f^i das Restnetzwerk G_f, welches in Iteration i verwendet wird.
- Zeige: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Dies gilt sicher für Kanten, die in G_f^i gelöscht wurden.
- Die hinzugefügten Kanten von G_f^i zu G_f^{i+1} werden auf der folgenden Folie behandelt.

$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.

$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

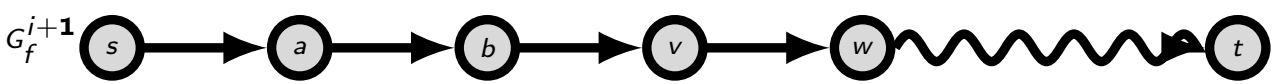
$n = |V|, m = |E|$

- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.

$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

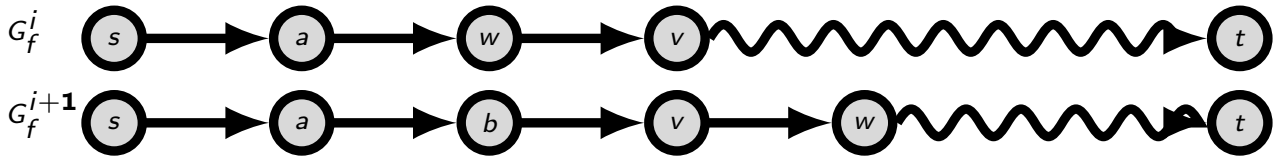
- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.
- Sei (v, w) eine Kante mit $(v, w) \in E(G_f^{i+1})$ und $(v, w) \notin E(G_f^i)$.



$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

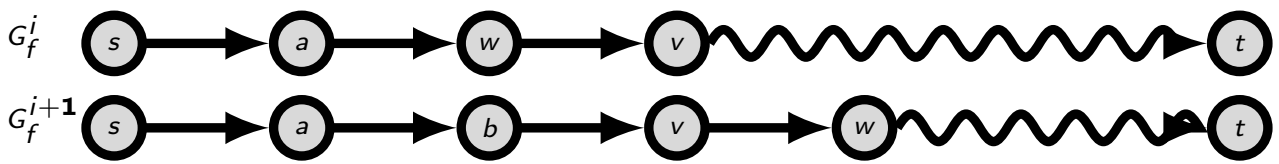
- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.
- Sei (v, w) eine Kante mit $(v, w) \in E(G_f^{i+1})$ und $(v, w) \notin E(G_f^i)$.
- Kante (v, w) wird eingefügt, wenn sie vorher als Kante (w, v) auf flussvergrößerndem Weg in G_f^i lag.



$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

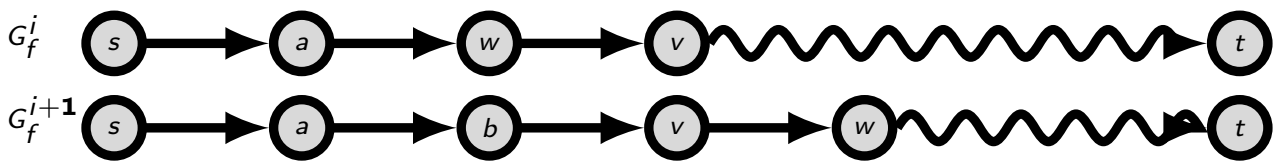
- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.
- Sei (v, w) eine Kante mit $(v, w) \in E(G_f^{i+1})$ und $(v, w) \notin E(G_f^i)$.
- Kante (v, w) wird eingefügt, wenn sie vorher als Kante (w, v) auf flussvergrößerndem Weg in G_f^i lag.
- Für den Knoten v gilt damit: $\text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$, denn nur für w kann die Distanz verbessert werden.



$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

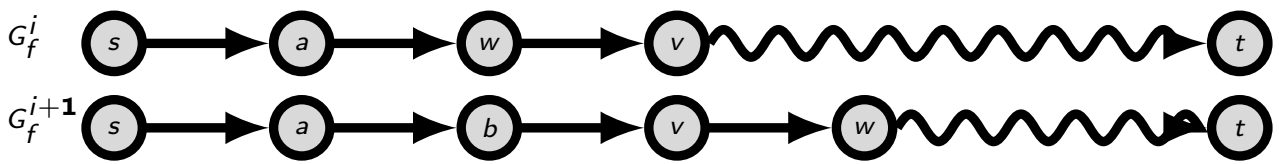
- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.
- Sei (v, w) eine Kante mit $(v, w) \in E(G_f^{i+1})$ und $(v, w) \notin E(G_f^i)$.
- Kante (v, w) wird eingefügt, wenn sie vorher als Kante (w, v) auf flussvergrößerndem Weg in G_f^i lag.
- Für den Knoten v gilt damit: $\text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$, denn nur für w kann die Distanz verbessert werden.
- Damit gilt: $\text{dist}_{G_f^i}(s, v) + 1 = \text{dist}_{G_f^{i+1}}(s, w)$.



$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

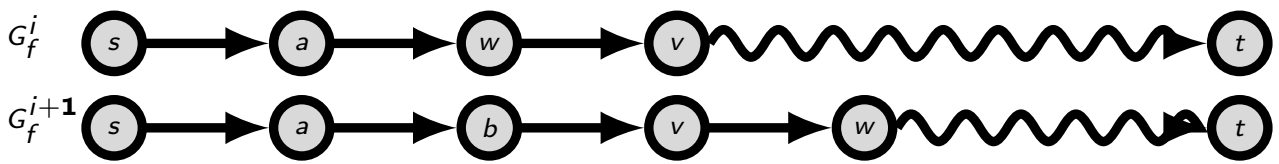
- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.
- Sei (v, w) eine Kante mit $(v, w) \in E(G_f^{i+1})$ und $(v, w) \notin E(G_f^i)$.
- Kante (v, w) wird eingefügt, wenn sie vorher als Kante (w, v) auf flussvergrößerndem Weg in G_f^i lag.
- Für den Knoten v gilt damit: $\text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$, denn nur für w kann die Distanz verbessert werden.
- Damit gilt: $\text{dist}_{G_f^i}(s, v) + 1 = \text{dist}_{G_f^{i+1}}(s, w)$.
- Es kann sich also der Abstand von w nicht verbessern,



$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

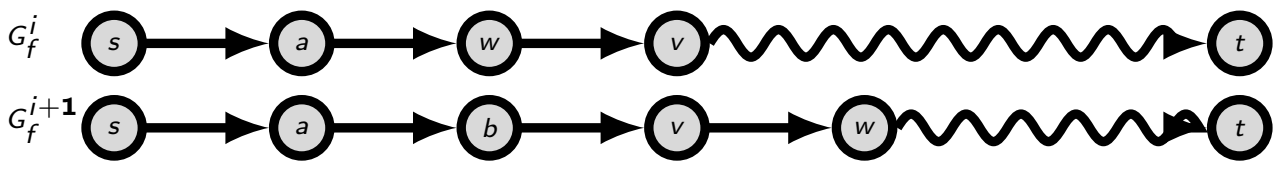
- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.
- Sei (v, w) eine Kante mit $(v, w) \in E(G_f^{i+1})$ und $(v, w) \notin E(G_f^i)$.
- Kante (v, w) wird eingefügt, wenn sie vorher als Kante (w, v) auf flussvergrößerndem Weg in G_f^i lag.
- Für den Knoten v gilt damit: $\text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$, denn nur für w kann die Distanz verbessert werden.
- Damit gilt: $\text{dist}_{G_f^i}(s, v) + 1 = \text{dist}_{G_f^{i+1}}(s, w)$.
- Es kann sich also der Abstand von w nicht verbessern,
- es gilt sogar: $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.



$\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$ (neue Kanten)

$n = |V|, m = |E|$

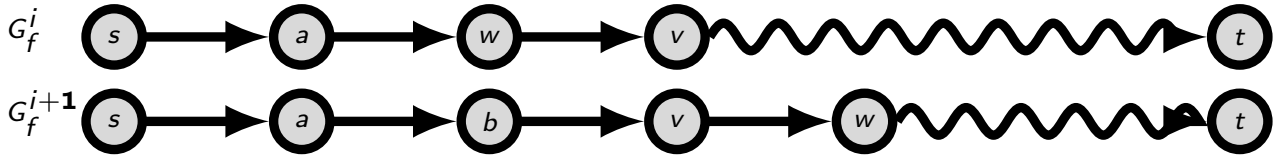
- Wir fügen kantenweise die neuen Kanten zu G_f^{i+1} hinzu.
- D.h. zeigen induktiv (über die Anzahl der eingefügten Kanten): $\forall x \in V : \text{dist}_{G_f^i}(s, x) \leqslant \text{dist}_{G_f^{i+1}}(s, x)$.
- Sei (v, w) eine Kante mit $(v, w) \in E(G_f^{i+1})$ und $(v, w) \notin E(G_f^i)$.
- Kante (v, w) wird eingefügt, wenn sie vorher als Kante (w, v) auf flussvergrößerndem Weg in G_f^i lag.
- Für den Knoten v gilt damit: $\text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$, denn nur für w kann die Distanz verbessert werden.
- Damit gilt: $\text{dist}_{G_f^i}(s, v) + 1 = \text{dist}_{G_f^{i+1}}(s, w)$.
- Es kann sich also der Abstand von w nicht verbessern,
- es gilt sogar: $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.



Beweis der Laufzeit

$n = |V|, m = |E|$

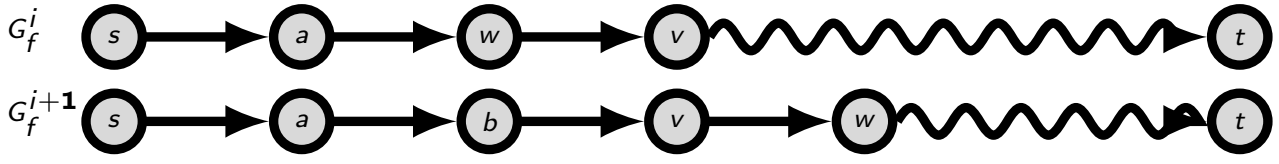
- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.



Beweis der Laufzeit

$n = |V|, m = |E|$

- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Falls (v, w) gelöscht wird, so gilt: $\text{dist}_{G_f^i}(s, v) + 1 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Wir sprechen hier von einer Flaschenhalskante.

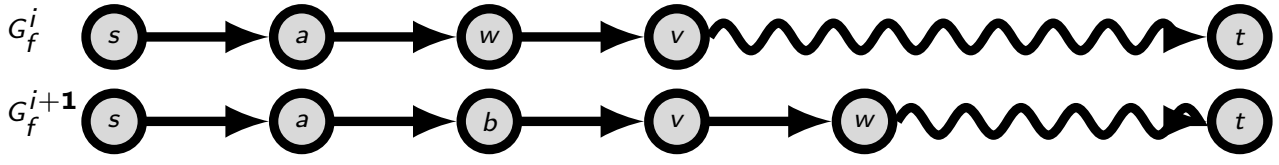


Beweis der Laufzeit

$n = |V|, m = |E|$

- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Falls (v, w) gelöscht wird, so gilt: $\text{dist}_{G_f^i}(s, v) + 1 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Wir sprechen hier von einer Flaschenhalskante.
- Falls (v, w) wieder eingefügt wird, so gilt:
  $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.

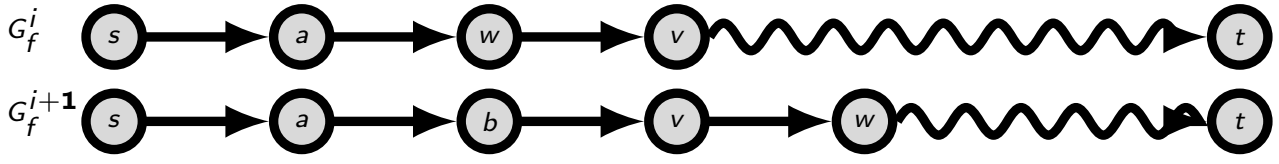
  Beachte: (v, w) lag auf einem kürzesten Weg in G_f^i.



Beweis der Laufzeit

$n = |V|, m = |E|$

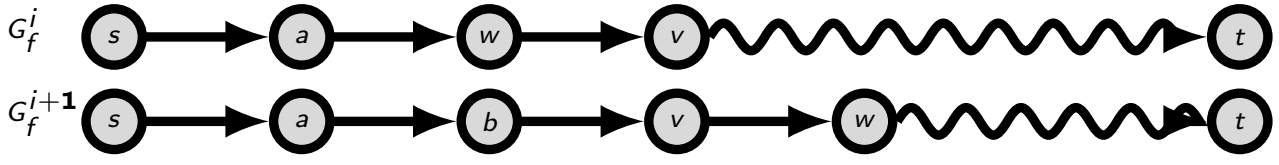
- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Falls (v, w) gelöscht wird, so gilt: $\text{dist}_{G_f^i}(s, v) + 1 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Wir sprechen hier von einer Flaschenhalskante.
- Falls (v, w) wieder eingefügt wird, so gilt:
  $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Beachte: (v, w) lag auf einem kürzesten Weg in G_f^i.
- Wegen $\text{dist}_{g_f^i}(s, t) \leqslant n - 1$ gilt:
  Jede Kante (v, w) kann höchstens $\lfloor (n-1)/2 \rfloor$ mal gelöscht werden.



Beweis der Laufzeit

$n = |V|, m = |E|$

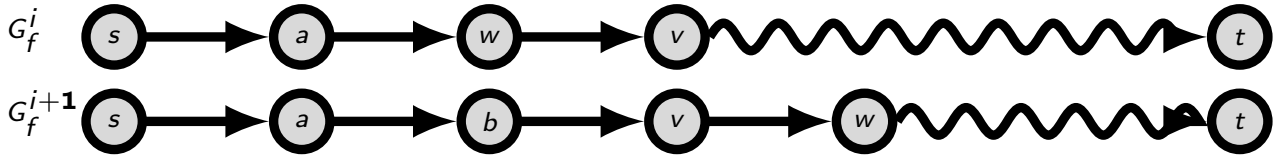
- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Falls (v, w) gelöscht wird, so gilt: $\text{dist}_{G_f^i}(s, v) + 1 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Wir sprechen hier von einer Flaschenhalskante.
- Falls (v, w) wieder eingefügt wird, so gilt:
  $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Beachte: (v, w) lag auf einem kürzesten Weg in G_f^i.
- Wegen $\text{dist}_{g_f^i}(s, t) \leqslant n - 1$ gilt:
  Jede Kante (v, w) kann höchstens $\lfloor (n-1)/2 \rfloor$ mal gelöscht werden.
- In jeder Iteration wird mindestens eine Kante entfernt.



Beweis der Laufzeit

$n = |V|, m = |E|$

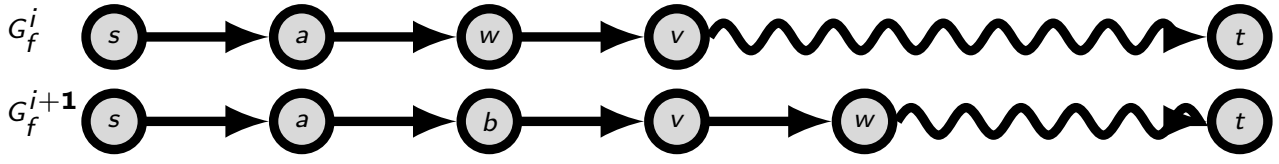
- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Falls (v, w) gelöscht wird, so gilt: $\text{dist}_{G_f^i}(s, v) + 1 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Wir sprechen hier von einer Flaschenhalskante.
- Falls (v, w) wieder eingefügt wird, so gilt:
  $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Beachte: (v, w) lag auf einem kürzesten Weg in G_f^i.
- Wegen $\text{dist}_{g_f^i}(s, t) \leqslant n - 1$ gilt:
  Jede Kante (v, w) kann höchstens $\lfloor (n-1)/2 \rfloor$ mal gelöscht werden.
- In jeder Iteration wird mindestens eine Kante entfernt.
- Im Restnetzwerk sind höchstens $2 \cdot m$ Kanten



Beweis der Laufzeit

$n = |V|, m = |E|$

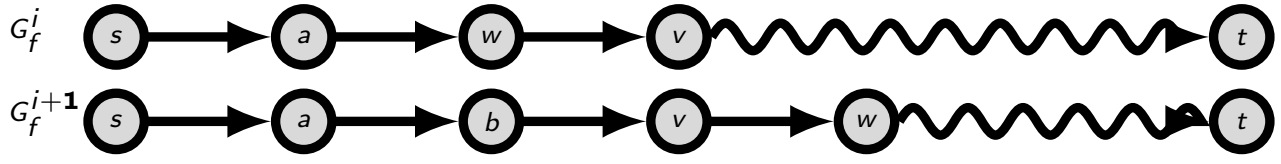
- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Falls (v, w) gelöscht wird, so gilt: $\text{dist}_{G_f^i}(s, v) + 1 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Wir sprechen hier von einer Flaschenhalskante.
- Falls (v, w) wieder eingefügt wird, so gilt:
  $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Beachte: (v, w) lag auf einem kürzesten Weg in G_f^i.
- Wegen $\text{dist}_{g_f^i}(s, t) \leqslant n - 1$ gilt:
  Jede Kante (v, w) kann höchstens $\lfloor (n-1)/2 \rfloor$ mal gelöscht werden.
- In jeder Iteration wird mindestens eine Kante entfernt.
- Im Restnetzwerk sind höchstens $2 \cdot m$ Kanten
- Anzahl der Iterationen damit höchstens: $\frac{n}{2} \cdot 2 \cdot m = n \cdot m$.



Beweis der Laufzeit

$n = |V|, m = |E|$

- Es gilt: $\forall v \in V : \text{dist}_{G_f^i}(s, v) \leqslant \text{dist}_{G_f^{i+1}}(s, v)$.
- Falls (v, w) gelöscht wird, so gilt: $\text{dist}_{G_f^i}(s, v) + 1 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Wir sprechen hier von einer Flaschenhalskante.
- Falls (v, w) wieder eingefügt wird, so gilt:
  $\text{dist}_{G_f^i}(s, w) + 2 \leqslant \text{dist}_{G_f^{i+1}}(s, w)$.
  Beachte: (v, w) lag auf einem kürzesten Weg in G_f^i.
- Wegen $\text{dist}_{g_f^i}(s, t) \leqslant n - 1$ gilt:
  Jede Kante (v, w) kann höchstens $\lfloor (n-1)/2 \rfloor$ mal gelöscht werden.
- In jeder Iteration wird mindestens eine Kante entfernt.
- Im Restnetzwerk sind höchstens $2 \cdot m$ Kanten
- Anzahl der Iterationen damit höchstens: $\frac{n}{2} \cdot 2 \cdot m = n \cdot m$.



Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.
- Cormen, Leiserson, Rives: Introduction to Algorithms, First Edition, MIT Press, 1990.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.
- Cormen, Leiserson, Rives: Introduction to Algorithms, First Edition, MIT Press, 1990.
- Cormen, Leiserson, Rives: Introduction to Algorithms, Second Edition, MIT Press, 2001.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.
- Cormen, Leiserson, Rives: Introduction to Algorithms, First Edition, MIT Press, 1990.
- Cormen, Leiserson, Rives: Introduction to Algorithms, Second Edition, MIT Press, 2001.
- Ottmann, Widmayer: Algorithmen und Datenstrukturen. BI-Wiss.-Verl. 1990.

Literatur

- Ahuja, Magnanti, Orlin: Network Flows: Theory, Algorihms, and Applications, Prentice Hall, 1993.
- Cormen, Leiserson, Rives: Introduction to Algorithms, First Edition, MIT Press, 1990.
- Cormen, Leiserson, Rives: Introduction to Algorithms, Second Edition, MIT Press, 2001.
- Ottmann, Widmayer: Algorithmen und Datenstrukturen. BI-Wiss.-Verl. 1990.