Algorithms and Probability

Week 13

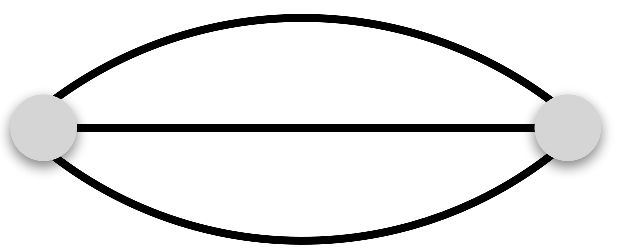
15/05/2025 — Georg Hasebe

Theory Recap

Definitions

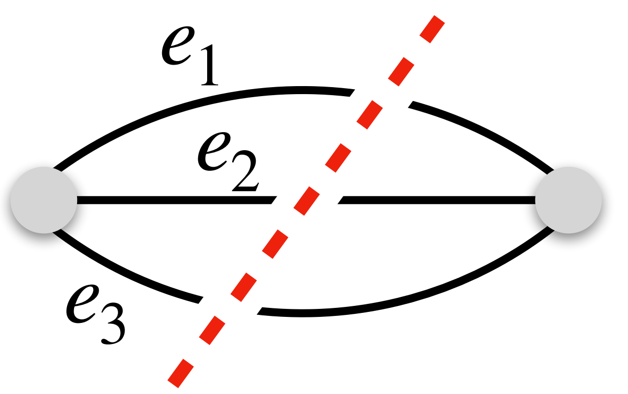
MIN-CUT Problem. Gegeben ein Multigraph G, bestimme die Kardinalität eines *minimalen Kantenschnitts*.

Multigraph: Ungerichteter, ungewichteter Graph $G = (V, E)$ ohne Schleifen, aber möglicherweise mit mehreren Kanten zwischen demselben Knotenpaar. (Kann auch durch positiv ganzzahlige Kantengewichte realisiert werden.)



Kantenschnitt in einem Multigraph $G = (V, E)$: Menge von Kanten C, so dass $(V, E \setminus C)$ unzusammenhängend ist.

(Kantenmenge $C \subseteq E$ vs. Partition (S, T) von V)



$C = \{e_1, e_2, e_3\}$

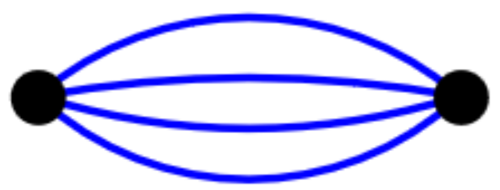
Mit $\mu(G)$ bezeichnen wir die Kardinalität eines kleinstmöglichen Kantenschnitts in G, d.h.

$$\mu(G) := \min_{C \subseteq E, (V, E \setminus C) \text{ unzusammenhängend}} |C|$$

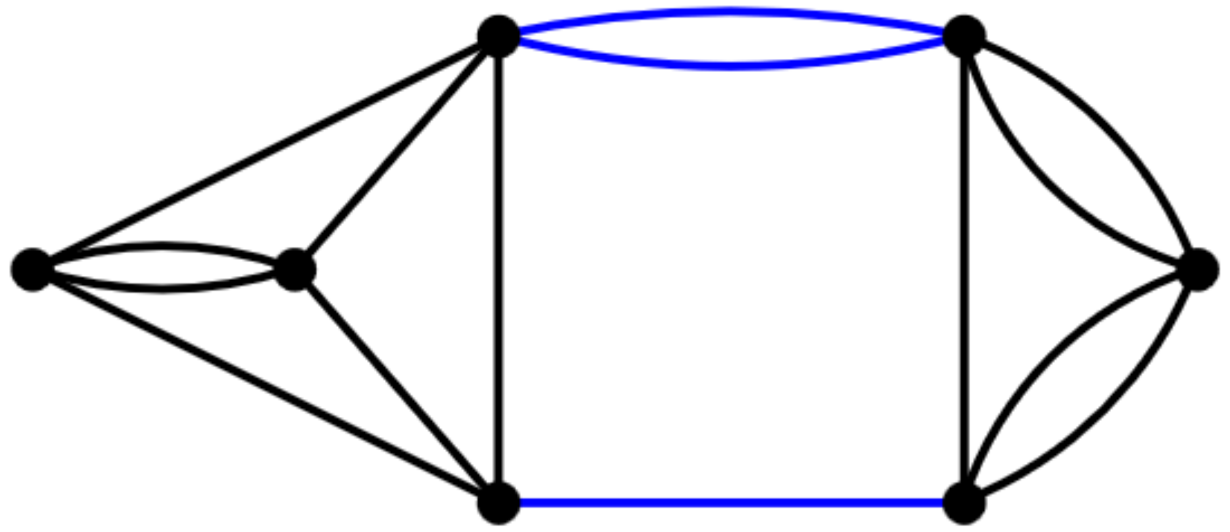
Beispiele

MIN-CUT Problem. Gegeben ein Multigraph G, bestimme $\mu(G)$.

Für G nicht zusammenhängend gilt $\mu(G) = 0$.



$\mu = 4$

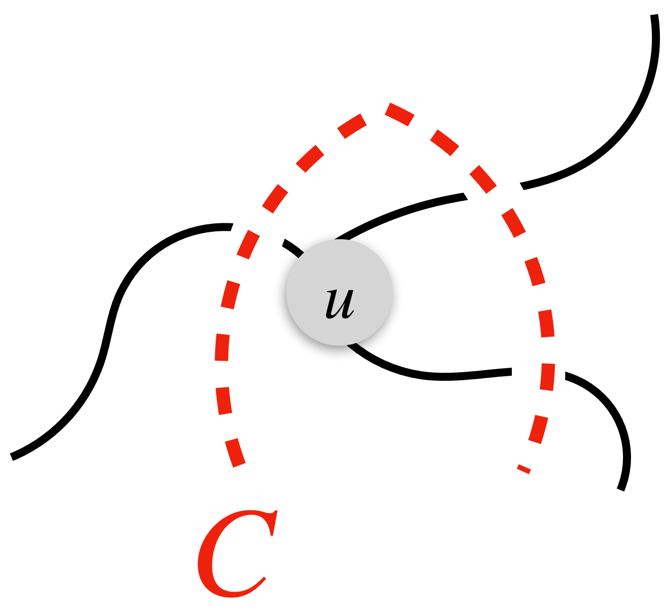


$\mu = 3$

Bemerkungen

In einem Multigraph $G = (V, E)$ ist der **Grad** $\deg(v) = \deg_G(v)$ eines Knoten v die #inzidenter Kanten (nicht die #Nachbarn). So gilt

$$|E| = \frac{1}{2}\sum_{v \in V} \deg(v) \quad \text{und} \quad \mu(G) \leq \min_{v \in V} \deg(v)\ .$$



Assuming u has the smallest degree in G, we can just remove all edges incident with u.

Kantenkontraktion

Gegeben $G = (V, E)$, $e = \{u, v\} \in E$.

Kontraktion von e: Verschmilzt u und v zu einem neuen Knoten $x_{u,v}$, der nun zu allen Kanten inzident ist, zu denen u oder v inzident war. Die Kanten zwischen u und v verschwinden.

Den entstehenden Graph bezeichnen wir mit G/e.

Zufällige Kantenkontraktion

$\textsc{Cut}(G)$ — G zusammenhängender Multigraph

while $|V(G)| > 2$ **do**
  $e \leftarrow$ gleichverteilt zufällige Kante in G
  $G \leftarrow G/e$
return Grösse des eindeutigen Schnitts in G

Zufällige Kantenkontraktion

CUT(G)	G zusammenhängender Multigraph

while $|V(G)| > 2$ **do**
  $e \leftarrow$ gleichverteilt zufällige Kante in G
  $G \leftarrow G/e$
return Grösse des eindeutigen Schnitts in G

Wir wollen den kleinstmöglichen Kantenschnitt $\mu(G)$.

Wie verändert sich dieser? Mit anderen Worten, in welcher Relation stehen $\mu(G)$ und $\mu(G/e)$?

Es stellt sich heraus: $\mu(G/e) \geq \mu(G)$.

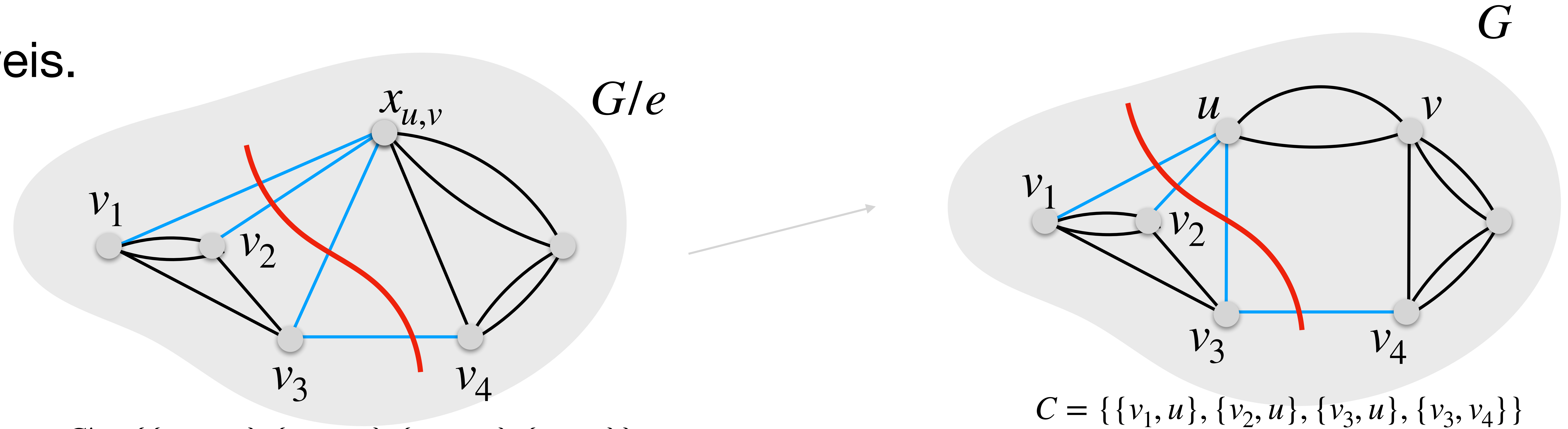
Beweis an der Tafel.

Zufällige Kantenkontraktion

In welcher Relation stehen $\mu(G)$ und $\mu(G/e)$?

Bemerkung 1: $\forall C$ in G/e $\exists C'$ in G such that $|C| = |C'|$.

Beweis.



$C' = \{\{v_1, x_{u,v}\}, \{v_2, x_{u,v}\}, \{v_3, x_{u,v}\}, \{v_3, v_4\}\}$

$C = \{\{v_1, u\}, \{v_2, u\}, \{v_3, u\}, \{v_3, v_4\}\}$

Es gibt eine natürliche Bijektion zwischen $\{$Kanten in G ausser denen zwischen u und $v\}$ und $\{$Kanten in $G/e\}$.

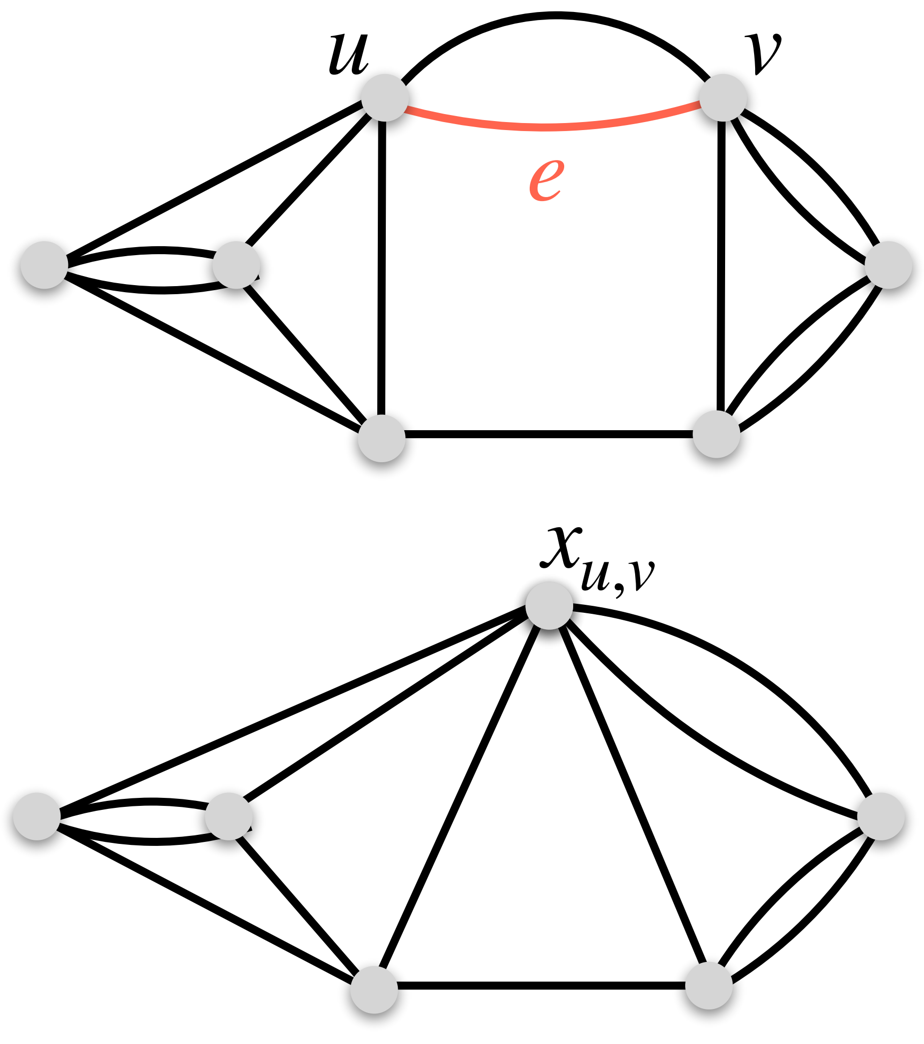
Zufällige Kantenkontraktion

In welcher Relation stehen $\mu(G)$ und $\mu(G/e)$?

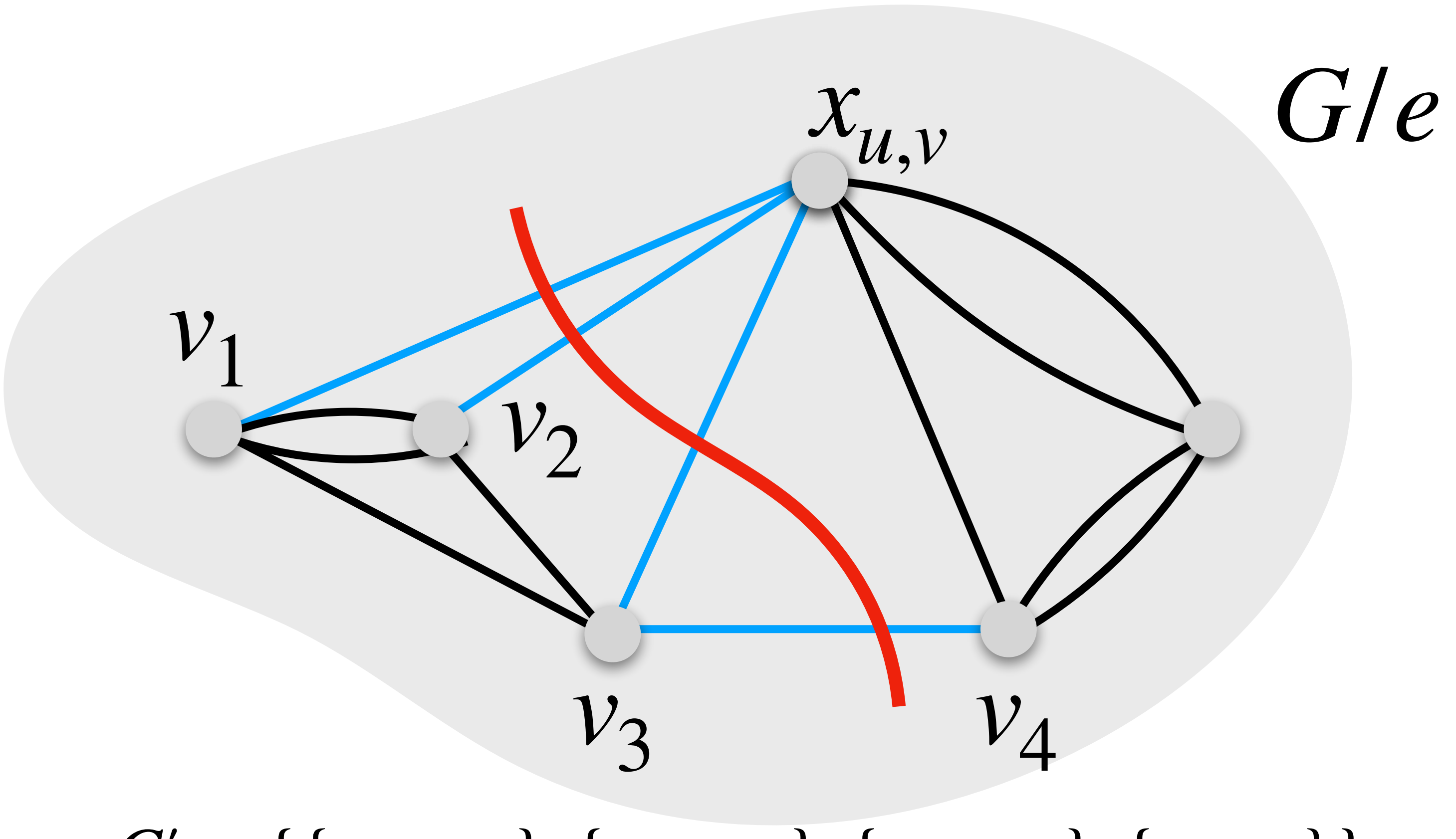
Bemerkung 2: ~~$\forall C$ in $G\,\exists C'$ in G/e such that $|C| = |C'|$.~~

<u>Stimmt nicht!</u>

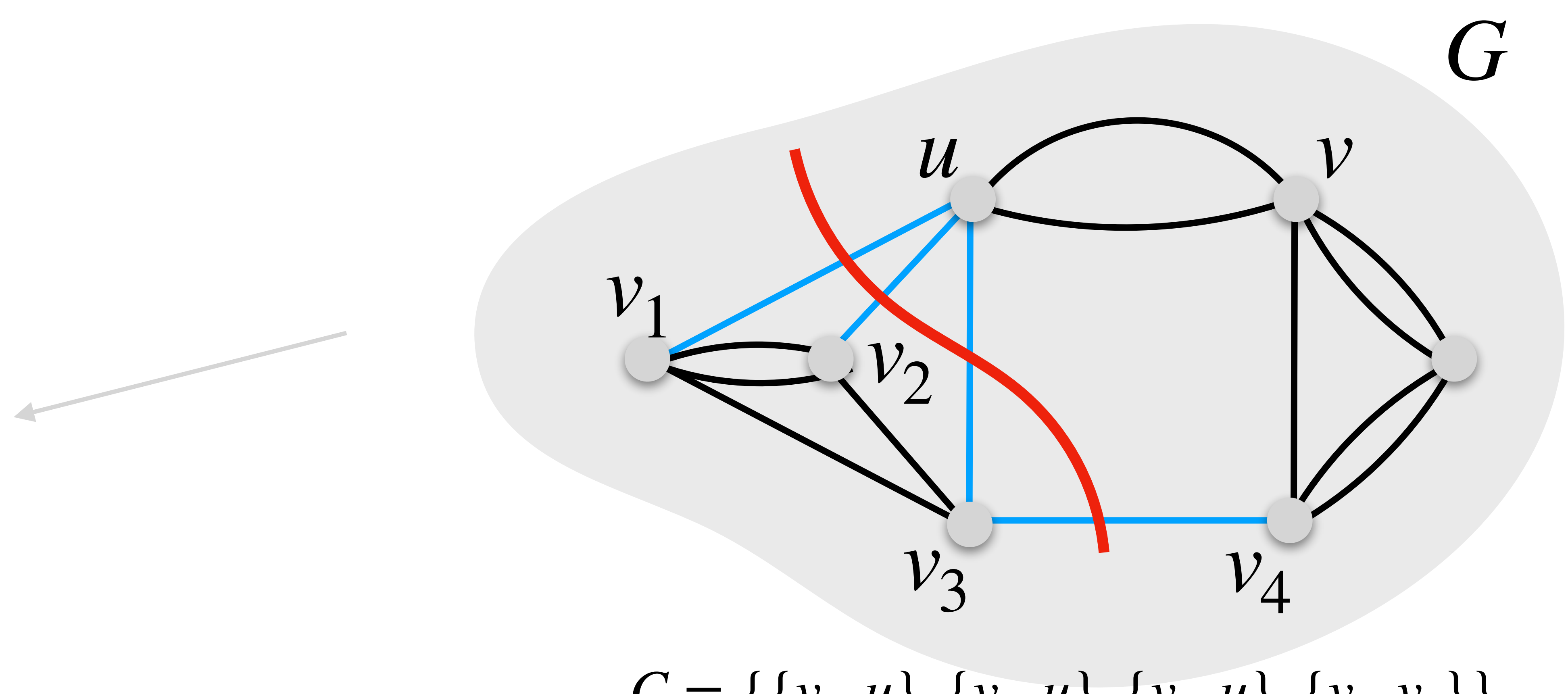
Bemerkung 3: $\forall C$ *ohne* e in $G\,\exists C'$ in G/e such that $|C| = |C'|$.



G mit $\mu(G) = 3$, aber G/e mit $\mu(G/e) = 4$.



$C' = \{\{v_1, x_{u,v}\}, \{v_2, x_{u,v}\}, \{v_3, x_{u,v}\}, \{v_3, v_4\}\}$



$C = \{\{v_1, u\}, \{v_2, u\}, \{v_3, u\}, \{v_3, v_4\}\}$

Zufällige Kantenkontraktion

Lemma 3.21. Sei $G = (V, E)$ ein Multigraph mit n Knoten. Falls e gleichverteilt zufällig unter den Kanten in G gewählt wird, dann gilt

$$\Pr\left[\mu(G) = \mu(G/e)\right] \geq 1 - \frac{2}{n} \, .$$

Beweis an der Tafel.

Erfolgswahrscheinlichkeit

$\hat{p}(G) :=$ Wahrscheinlichkeit, dass Cut(G) den Wert $\mu(G)$ ausgibt

$$\hat{p}(n) := \inf_{G=(V,E),\,|V|=n} \hat{p}(G) \;.$$

Lemma 3.22. Es gilt für alle $n \geq 3$

$$\hat{p}(n) \geq (1 - 2/n) \cdot \hat{p}(n-1).$$

Beweis im Skript.

Erfolgswahrscheinlichkeit

Es gilt also $\hat{p}(n) \geq \frac{n-2}{n} \cdot \hat{p}(n-1)$. Wir erhalten so

$$\hat{p}(n) \geq \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdots \frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3} \cdot \underbrace{\hat{p}(2)}_{=1} = \frac{2}{n(n-1)}$$

Lemma 3.23. Es gilt $\hat{\mathfrak{p}}(n) \geq \frac{2}{n(n-1)} = 1/\binom{n}{2}$ für alle $n \geq 2$.

Laufzeit

$\textsc{Cut}(G)$	G zusammenhängender Multigraph

```
1: while |V(G)| > 2 do
2:     e ← gleichverteilt zufällige Kante in G
3:     G ← G/e
4: return Grösse des eindeutigen Schnitts in G
```

Sei n Anzahl der Konten in G. Für die Implementierung setzen wir folgendes voraus:

- Eine Kantenkontraktion kann in $O(n)$ Zeit durchgeführt werden.
- Eine gleichverteilt zufällige Kante in G kann in $O(n)$ gewählt werden.

(Das ist nicht ganz offensichtlich, und erfordert u.a., dass Mehrfachkanten mittels Kantengewichten dargestellt werden.) Mit dieser Voraussetzung können wir $\textsc{Cut}(G)$ mit einer Laufzeit von $O(n^2)$ implementieren.

Resultat

Satz 3.24. Für den Algorithmus der $\lambda\binom{n}{2}$-maligen Wiederholung von $\textsc{Cut}(G)$ gilt:

(1) Der Algorithmus hat eine Laufzeit von $O(\lambda n^4)$.

(2) Der kleinste angetroffene Wert ist mit einer Wahrscheinlichkeit von mindestens $1 - e^{-\lambda}$ gleich $\mu(G)$.

Zu (1): eine Ausführung von Cut(G) kann in $O(n^2)$ ausgeführt werden.
Zu (2):

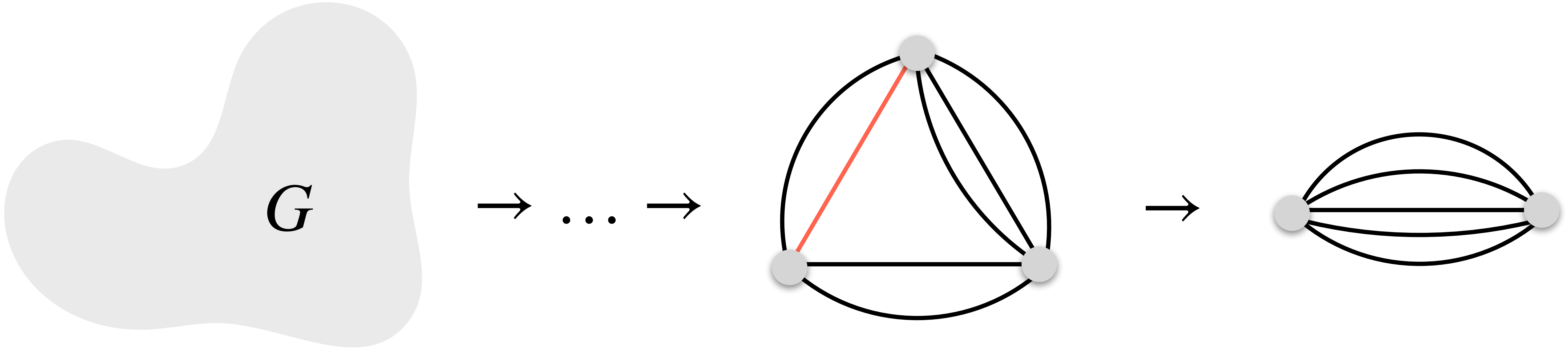
$$(1 - \hat{p}(n))^{\lambda\binom{n}{2}} \leq \left(1 - 1/\binom{n}{2}\right)^{\lambda\binom{n}{2}} \leq e^{-\lambda}$$

wobei $\forall x \in \mathbb{R} : 1 + x \leq e^x$ verwendet wurde.

Bootstrapping

Wir erinnern uns daran, dass, wenn C ein minimaler Schnitt ist, dann gilt

$$e \notin C \implies \mu(G/e) = \mu(G).$$



Ein Multigraph mit vielen Knoten und Kanten. Die Wahrscheinlichkeit, hier zufällig eine Kante zu wählen, die Teil des minimalen Schnittes ist, ist sehr gering.

Falls wir den minimalen Schnitt bis hier hin erhalten konnten, ist die Wahrscheinlichkeit, hier zufällig eine Kante zu wählen, die Teil des minimalen Schnittes ist, ist sehr hoch.

Bootstrapping

Wir erinnern uns daran, dass $\hat{p}(n)$ die Erfolgswahrscheinlichkeit ist. Wir wollen also, dass $\hat{p}(n)$ möglichst nahe an 1 ist.

$$\hat{p}(n) \geq \underbrace{\frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2}}_{\text{Für sehr grosse } n\text{, sehr nahe an } 1.} \cdots \underbrace{\frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3}}_{0.6,\, 0.5,\, 0.33\ldots} \cdot \underbrace{\hat{p}(2)}_{=1} = \frac{2}{n(n-1)}$$

Bootstrapping

<u>Idee:</u> Wir brechen Cut(G) ab, wenn wir bei einem Multigraphen mit t Knoten angelangt sind.

Danach lösen wir den Rest mit einem anderen Algorithmus in Laufzeit $z(t)$.

$\textsc{Cut1}(\mathrm{G})$	G zusammenhängender Multigraph
while $\lvert \mathrm{V(G)} \rvert > \mathrm{t}$ **do**	
$\quad e \leftarrow$ gleichverteilt zufällige Kante in G	
$\quad \mathrm{G} \leftarrow \mathrm{G}/e$	
return Grösse des eindeutigen Schnitts in G	$\triangleright$ in Zeit $\mathrm{O}(z(\mathrm{t}))$

Bootstrapping

$\textsc{Cut1}(\mathrm{G})$	G zusammenhängender Multigraph

while $|V(G)| > t$ **do**
  $e \leftarrow$ gleichverteilt zufällige Kante in G
  $G \leftarrow G/e$
return Grösse des eindeutigen Schnitts in G $\triangleright$ in Zeit $O(z(t))$

Welchen Algorithmus wählen wir im letzten Schritt?

(1) Deterministisch mit Flüssen in Laufzeit $O(t^4 \log t)$ und Erfolgswahrscheinlichkeit $p^*(t) = 1$.

(2) Eine Wiederholung ($\lambda = 1$) von Cut(G) in Laufzeit $O(t^4)$ und Erfolgswahrscheinlichkeit $p^*(t) = 1 - e^{-1}$.

Bootstrapping

$\textsc{Cut1}(G)$	G zusammenhängender Multigraph
while $\lvert V(G)\rvert > t$ **do**	
$\quad e \leftarrow$ gleichverteilt zufällige Kante in G	
$\quad G \leftarrow G/e$	
return Grösse des eindeutigen Schnitts in G	$\triangleright$ in Zeit $O(z(t))$

(2) Eine Wiederholung ($\lambda = 1$) von Cut(G) in Laufzeit $O(t^4)$ und Erfolgswahrscheinlichkeit $p^*(t) = 1 - e^{-1}$.

Die Laufzeit von Cut1(G) ist $O(\underbrace{n \cdot (n - t)}_{\text{Nimm wieder an, dass Kantenkontraktion in } O(n)\text{.}} + t^4)$, die Erfolgswahrscheinlichkeit ergibt sich durch

$$\hat{p}(n) \geq \underbrace{\frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdots \frac{t+1}{t+3} \cdot \frac{t}{t+2} \cdot \frac{t-1}{t+1}}_{\text{Führe Cut(G) aus bis wir einen Multigraphen mit } t \text{ Knoten haben.}} \cdot \underbrace{p^*(t)}_{\text{Führe (2) aus.}} = \frac{t(t-1)}{n(n-1)} \cdot p^*(t)\,.$$

Bootstrapping

```
Cut1(G)                                       G zusammenhängender Multigraph
while |V(G)| > t do
    e ← gleichverteilt zufällige Kante in G
    G ← G/e
return Grösse des eindeutigen Schnitts in G           ▷ in Zeit O(z(t))
```

(2) Eine Wiederholung ($\lambda = 1$) von Cut(G) in Laufzeit $O(t^4)$ und Erfolgswahrscheinlichkeit $p^*(t) = 1 - e^{-1}$.

Cut1(G) mit Laufzeit $O(n \cdot (n-t) + t^4)$ und Erfolgswskt. $\hat{p}(n) \geq \dfrac{t(t-1)}{n(n-1)} \cdot p^*(t)$.

Bei $\lambda \dfrac{n(n-1)}{t(t-1)} \dfrac{e}{e-1}$ Wiederholungen von Cut1(G), erreichen wir eine Erfolgswskt. von $1 - e^{-\lambda}$ und eine Laufzeit von $O\left(\lambda\left(\dfrac{n^4}{t^2} + n^2t^2\right)\right)$.

Bootstrapping

Cut1(G)	G zusammenhängender Multigraph
1: **while** $\|V(G)\| > t$ **do**	
2: $\quad e \leftarrow$ gleichverteilt zufällige Kante in G	
3: $\quad G \leftarrow G/e$	
4: **return** Grösse des eindeutigen Schnitts in G	$\triangleright$ in Zeit $O(z(t))$

(2) Eine Wiederholung ($\lambda = 1$) von Cut(G) in Laufzeit $O(t^4)$ und Erfolgswahrscheinlichkeit $p^*(t) = 1 - e^{-1}$.

Cut1(G) bei $\lambda \dfrac{n(n-1)}{t(t-1)} \dfrac{e}{e-1}$ Wiederholungen liefert Erfolgswskt. von $1 - e^{-\lambda}$

und eine Laufzeit von $O\left(\lambda\left(\dfrac{n^4}{t^2} + n^2t^2\right)\right)$.

Für $t = n^{1/2}$ erhalten wir so $O(\lambda n^3)$.

Bootstrapping

```
Cut1(G)                                        G zusammenhängender Multigraph
1: while |V(G)| > t do
2:     e ← gleichverteilt zufällige Kante in G
3:     G ← G/e
4: return Grösse des eindeutigen Schnitts in G        ▷ in Zeit O(z(t))
```

Wir haben damit die Laufzeit von $O(\lambda n^4)$ auf $O(\lambda n^3)$ verbessern können.

<u>ABER:</u> Nun haben wir für den letzten Schritt in Zeile 4 eine neue Möglichkeit hinzugewonnen:

(3) Eine Wiederholung ($\lambda = 1$) von Cut1(G) in Laufzeit $O(t^3)$ und Erfolgswahrscheinlichkeit $p^*(t) = 1 - e^{-1}$.

Dieses in sich selbst einsetzen eines Algorithmus, um diesen immer schneller zu machen, nennt man **Bootstrapping**.