Min-Cut Problem

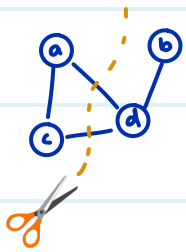
Multigraph: $G = (V, E)$ ohne Schleifen, ungerichtet, ungewichtet, mehrere Kanten zwischen zwei Knoten sind erlaubt

Der Grad ist die Anzahl anliegender Kanten, und nicht die Anzahl Nachbaren

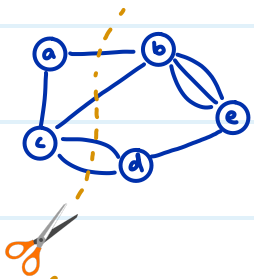
$\deg(a) = 3$, $\deg(b) = 5$, $\deg(c) = 2$

"C" für Cut

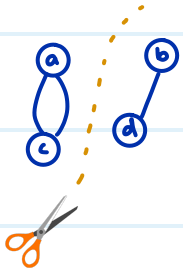
Kantenschnitt (= edge cut) Eine Teilmenge der Kanten $C \subseteq E$, sodass $(V, E \setminus C)$ unzusammenhängend ist



$C = \{\{a,d\},\{c,d\}\}$



$C = \{\{a,b\},\{b,c\},\{c,d\},\{c,d\}\}$

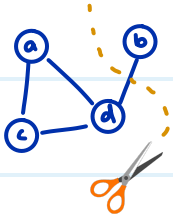


$C = \emptyset$

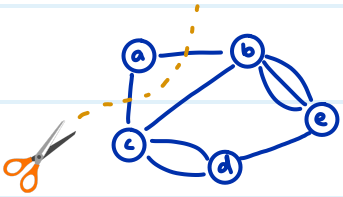
Achtung: dieser Cut ist nicht dasselbe wie der Cut bei Flüssen

"μ" für min

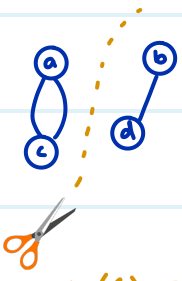
$\mu(G)$: Kardinalität $|C|$ des kleinstmöglichen Schnitts C im Multigraphen G



$\mu(G) = 1$



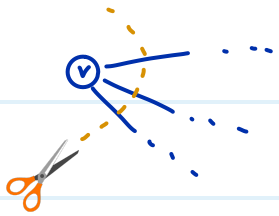
$\mu(G) = 2$



$\mu(G) = 0$

Es gilt immer: $\mu(G) \leq \min_{v \in V} \deg(v)$

kleinster Grad



wir könnten einfach diesen Knoten herausschneiden

Min-Cut Algorithmus via Flüsse:

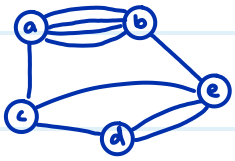
1) Gegeben einen Multigraphen G, konstruieren wir ein Netzwerk $N = (V, A, c, s, t)$

wobei
- N dieselben Knoten hat wie G
- A hat eine Kante (u,v) und (v,u) falls G die Kante $\{u,v\}$ hat
- $c(u,v)$ = Anzahl Kanten zwischen u und v in G
- s ist ein beliebiger, fixer Knoten
- t... (nächster Schritt)

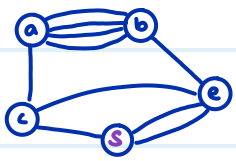
2) Wir iterieren über alle möglichen targets $t \in V \setminus \{s\}$ und berechnen die Kapazität $cap(S,T)$ des minimalen Schnitts (S,T) mit dem maxflow Algorithmus

3) Gib die kleinste Kapazität aus, die gefunden wurde

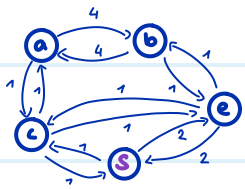
Bsp 1) G:



wir wählen s = d

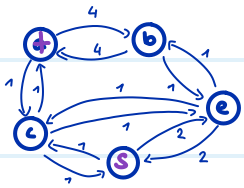


N:

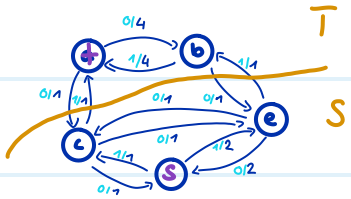


2) Wir iterieren über alle targets $t \in \{a, b, c, e\}$:

$t = a$:

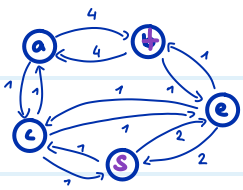


max flow Algorithmus →

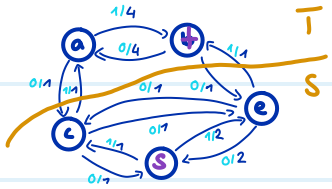


max flow = min cap(S,T)
= 2

$t = b$

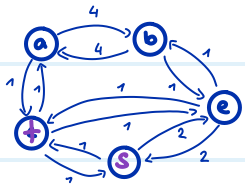


max flow Algorithmus →

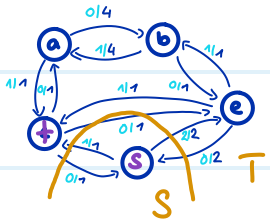


max flow = min cap(S,T)
= 2

$t = c$:

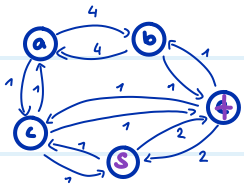


max flow Algorithmus →

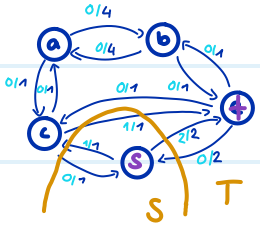


max flow = min cap(S,T)
= 3

$t = e$:



max flow Algorithmus →



max flow = min cap(S,T)
= 3

3.) $\mu(G) = \min\{2, 2, 3, 3\} = \underline{\underline{2}}$

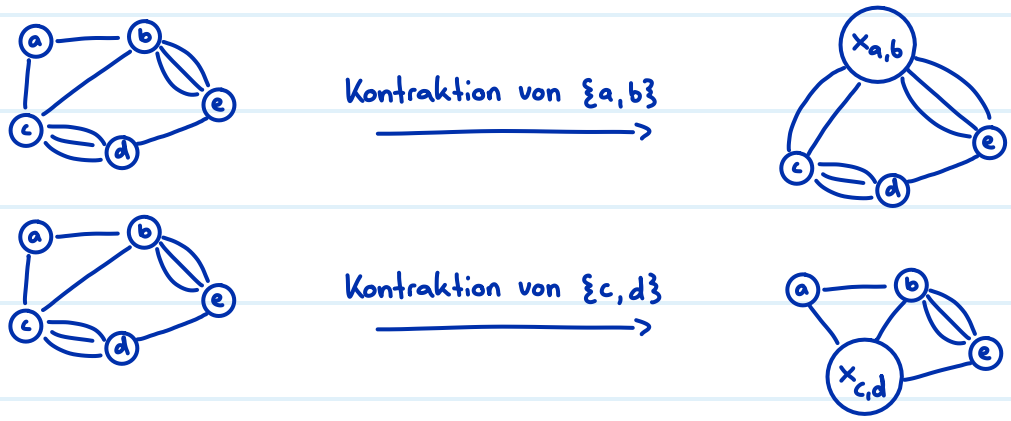
Intuition: Cap(S,T) ist genau die minimale Anzahl an Kanten, die man entfernen muss, damit s und t nicht mehr verbunden sind in G

Laufzeit: $O(\underbrace{n}_{\text{über } t \text{ iterieren}} \underbrace{n^3 \log n}_{\text{max flow in } O(m\, n \log n) \text{ mit } m \leq n^2}) = O(n^4 \cdot \log n)$

Kontraktionen

Kontraktion einer Kante $e = \{u, v\}$:

- u und v werden zu einem Knoten $x_{u,v}$
- Kanten zwischen u und v werden entfernt
- andere Kanten nach u oder v gehen neu zu $x_{u,v}$
- den entstehenden Graphen nennen wir G/e

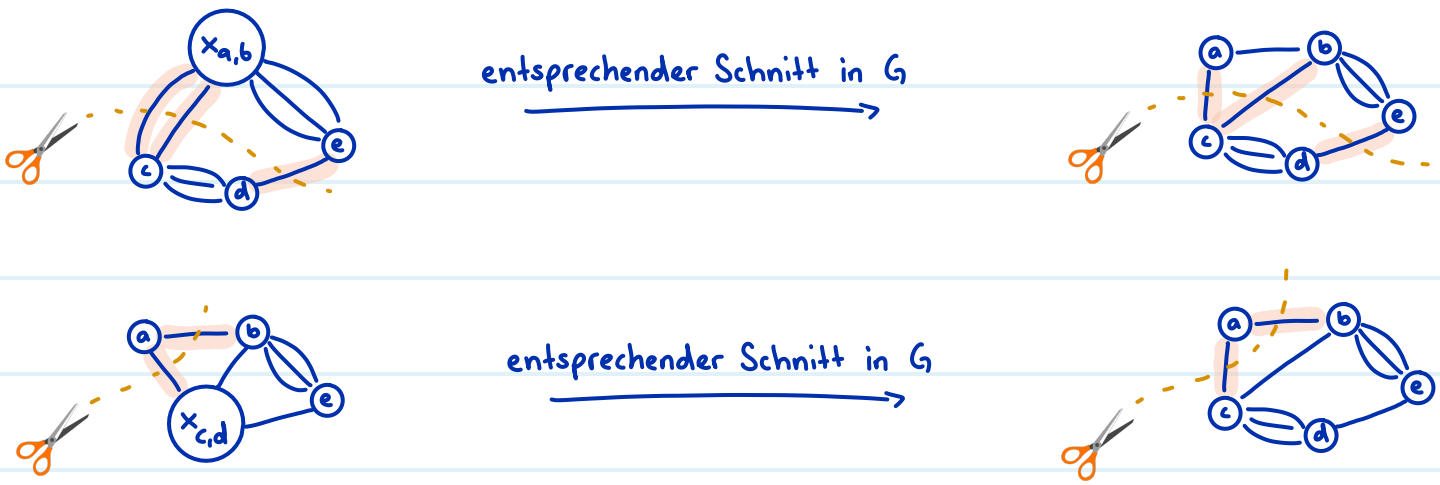


Für alle Kanten e gilt: $\mu(G \setminus e) \geq \mu(G)$

Das heisst, durch Kontraktionen wird der minimale Kantenschnitt grösser oder er bleibt gleich.

Beweis: Sei C ein minimaler Schnitt in $G \setminus e$

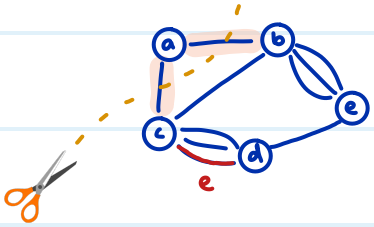
Dann gibt es auch einen Schnitt in G mit "denselben" Kanten



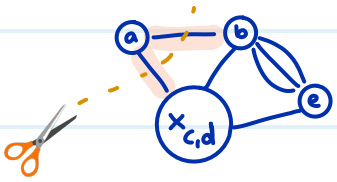
Falls G einen minimalen Kantenschnitt C mit $e \notin C$ hat, dann ist $\mu(G \setminus e) = \mu(G)$

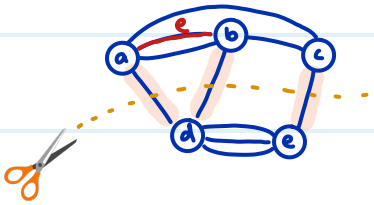
Beweis: Sei C ein minimaler Schnitt in G mit $e \notin C$

Dann gibt es auch einen Schnitt in G/e mit "denselben" Kanten

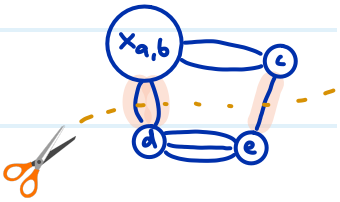


entsprechender Schnitt in G/e





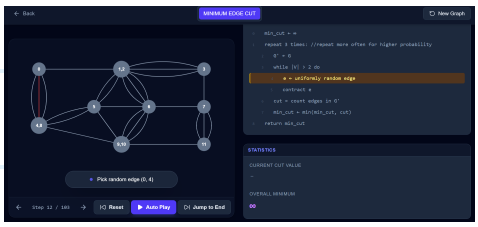
entsprechender Schnitt in G/e



Algorithmus:

CUT(G) in $O(n^2)$ — G zusammenhängender Multigraph

1: $G' \leftarrow G$
2: **while** $|V(G')| > 2$ **do** → $O(n)$ Iterationen
3: $e \leftarrow$ gleichverteilt zufällige Kante in G'
4: $G' \leftarrow G'/e$ → in $O(n)$
5: **return** Grösse des eindeutigen Schnitts in G'



- Monte-Carlo Algorithmus

Falls e zufällig gleichverteilt ausgewählt wird, gilt $\Pr[\mu(G) = \mu(G \setminus e)] \geq 1 - \frac{n}{2}$

Beweis: Sei C ein minimaler Kantenschnitt und $e \in E$ zufällig gleichverteilt

$$|E| = \frac{1}{2} \sum_{v \in V} \deg(v)$$ Handschlag Lemma

$$\Rightarrow \quad |E| \geq \frac{1}{2} \sum_{v \in V} |C|$$ weil für alle $v \in V$ gilt: $\deg(v) \geq \underbrace{\mu(G)}_{= |C|}$

$\Rightarrow \quad |E| \geq \frac{1}{2} \, n \, |C|$ $\qquad |V| = n$

$$\Pr[\mu(G) = \mu(G \setminus e)] \geq \Pr[e \notin C] \qquad \text{schon gezeigt}$$

$$= 1 - \Pr[e \in C] \qquad \text{Gegenwahrscheinlichkeit}$$

$$= 1 - \frac{|C|}{|E|} \qquad \text{zufällig gleichverteilt} \rightarrow \text{Laplace Raum}$$

$$\geq 1 - \frac{|C|}{\frac{1}{2} n |C|} \qquad |E| \geq \frac{1}{2} n |C|$$

$$= 1 - \frac{2}{n} \qquad \text{kürzen}$$

Wir definieren: $\hat{p}(G) := \Pr[\text{Cut}(G) \text{ ist korrekt}]$ (Erfolgswahrscheinlichkeit für G)

und $\hat{p}(n) := \inf_{G=(V,E), |V|=n} \hat{p}(G)$ (Erfolgswahrscheinlichkeit für den schwierigsten Graphen mit n Knoten)

Es gilt $\forall n \geq 3: \; \hat{p}(n) \geq \frac{2}{n(n-1)}$

Beweis Sei G ein Multigraph mit n Knoten

"Cut(G) ist korrekt" tritt genau dann ein, wenn folgende beide Ereignisse eintreten

$E_1 := \mu(G) = \mu(G \setminus e)$ (= erste Kontraktion erhält den min cut)

$E_2 :=$ Cut($G \setminus e$) ist korrekt (= restliche Kontraktionen erhalten den min cut)

Das heisst $\hat{p}(G) = \Pr[E_1 \wedge E_2]$

$$= \Pr[E_1] \; \Pr[E_2 \mid E_1] \qquad \text{(Multiplikationssatz)}$$

$$\geq \left(1 - \frac{2}{n}\right) \; \hat{p}(n-1) \qquad \left(\Pr[E_1] \geq 1 - \frac{2}{n} \text{ wurde oben gezeigt,}\right.$$

$$\left. \text{und } \Pr[E_2 \mid E_1] \geq \hat{p}(n-1) \text{ per Definition}\right)$$

Weil dies für jeden beliebigen Multigraphen mit n Knoten gilt, folgt:

$$\hat{p}(n) \geq \left(1 - \frac{2}{n}\right) \hat{p}(n-1)$$

Durch wiederholtes Einsetzen ergibt sich:

$$\hat{p}(n) \geq \frac{n-2}{n} \cdot \hat{p}(n-1) \qquad \left(1 - \frac{n}{2} = \frac{n-2}{n}\right)$$

$$\geq \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \hat{p}(n-2)$$

$$\vdots$$

$$\geq \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdot \frac{n-5}{n-3} \cdot \ldots \cdot \frac{4}{6} \cdot \frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3} \cdot \underbrace{\hat{p}(2)}_{=1,\ \text{trivialer Fall}}$$

$$= \frac{\cancel{n-2}}{n} \cdot \frac{\cancel{n-3}}{\cancel{n-1}} \cdot \frac{\cancel{n-4}}{\cancel{n-2}} \cdot \frac{\cancel{n-5}}{\cancel{n-3}} \cdot \ldots \cdot \frac{\cancel{4}}{\cancel{6}} \cdot \frac{\cancel{3}}{\cancel{5}} \cdot \frac{\cancel{2}}{\cancel{4}} \cdot \frac{\cancel{1}}{\cancel{3}}$$

$$= \frac{2}{n\,(n-1)}$$

Fehlerreduktion für Cut(G)

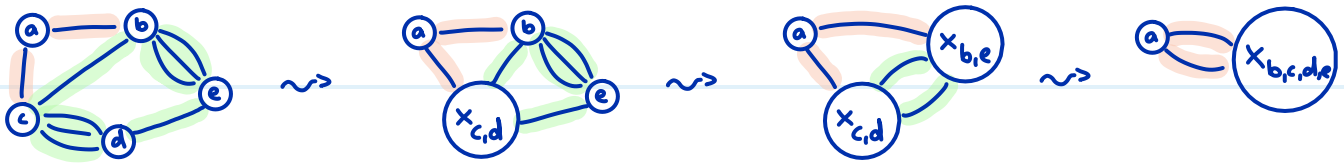
- aktuelle Korrektheit: $\varepsilon \geq \frac{2}{n\,(n-1)}$
- Zielfehler: $\delta = e^{-\lambda}$

$\Rightarrow N = \lceil \varepsilon^{-1} \ln(\delta^{-1}) \rceil = \lceil \frac{n\,(n-1)}{2} \ln(e^{\lambda}) \rceil = \lceil \lambda \frac{n\,(n-1)}{2} \rceil$ Wiederholungen, von denen wir am Schluss den kleinst gefundenen Schnitt ausgeben

Laufzeit in $O(\underbrace{n^2}_{\text{Laufzeit von Cut(G)}} \cdot \underbrace{\lambda n^2}_{\text{Wiederholungen}}) = O(\lambda n^4)$ → schlecht

Bootstrapping

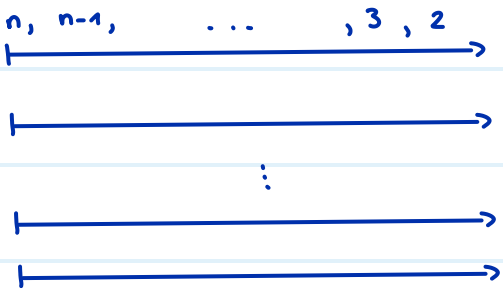
Feststellung: umso kleiner der Graph wird, desto wahrscheinlicher ist es, dass der Algorithmus einen Fehler macht



rote Kante kontrahieren = Fehler, grüne Kante kontrahieren = min cut bleibt erhalten

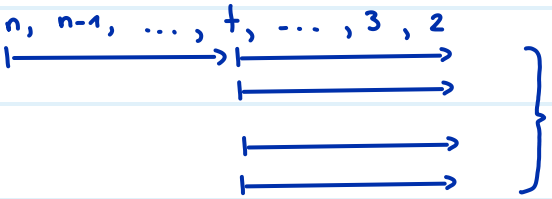
Idee: vorsichtiger sein wo es gefährlich ist

vorher:



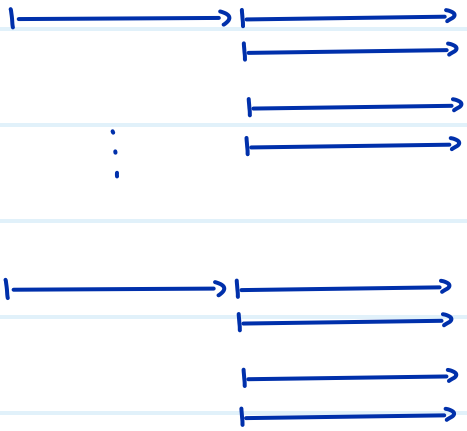
$\lambda \frac{n \cdot (n-1)}{2}$ Wiederholungen von Cut(G)

bootstrapping:



$\frac{t \cdot (t-1)}{2}$ Wiederholungen von Cut(G')

$\lambda \cdot \frac{n\,(n-1)}{t\,(t-1)} \cdot \frac{e}{e-1}$ Wiederholungen von $Cut_t(G)$



$\text{CUT}_t(G)$ — G zusammenhängender Multigraph

1: $G' \leftarrow G$
2: **while** $|V(G')| > t$ **do** → bei t übrigen Knoten abbrechen
3: $e \leftarrow$ gleichverteilt zufällige Kante in G'
4: $G' \leftarrow G'/e$
5: **return** Resultat des $O(t^4)$ Algorithmus auf G' $= \lambda \frac{t(t-1)}{2}$ Wiederholungen von $Cut(G')$

mit $\lambda = 1$ $\Rightarrow$ Erfolgswahrscheinlichkeit $\geq 1 - e^{-\lambda}$
$= 1 - e^{-1}$
$= \frac{e-1}{e}$

Wir definieren: $\hat{p}_t(G) := Pr[Cut_t(G) \text{ ist korrekt}]$ (Erfolgswahrscheinlichkeit für G)

und $\hat{p}_t(n) := \inf_{G=(V,E), |V|=n} \hat{p}_t(G)$ (Erfolgswahrscheinlichkeit für den schwierigsten Graphen mit n Knoten)

Analog erhalten wir

$$\hat{p}_t(n) \geq \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdot \ldots \cdot \frac{t}{t+2} \frac{t-1}{t+1} \underbrace{\hat{p}_t(t)}_{\geq \frac{e-1}{e}}$$

$$\geq \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdot \ldots \cdot \frac{t}{t+2} \frac{t-1}{t+1} \frac{e-1}{e}$$

$$= \frac{t(t-1)}{n(n-1)} \frac{e-1}{e}$$

Fehlerreduktion für $Cut_t(G)$

- aktuelle Korrektheit: $\varepsilon \geq \frac{t(t-1)}{n(n-1)} \frac{e-1}{e}$
- Zielfehler: $\delta = e^{-\lambda}$

$\Rightarrow N = \lceil \varepsilon^{-1} \ln(\delta^{-1}) \rceil = \lceil \lambda \frac{n(n-1)}{t(t-1)} \cdot \frac{e}{e-1} \rceil$ Wiederholungen

Laufzeit in $O\left(\left(n \underbrace{(n-t)}_{\text{Anzahl Kontraktionen bis } t \text{ Knoten übrig bleiben}} + \underbrace{t^4}_{\text{Zeit für eine Kontraktion}}\right) \cdot \lambda \frac{n(n-1)}{t(t-1)} \cdot \frac{e}{e-1}\right) \leq O\left(\lambda \left(\frac{n^4}{t^2} + n^2 \cdot t^2\right)\right)$

$\frac{t(t-1)}{2}$ Wiederholungen von $Cut(G')$ in jeweils $O(t^2)$

Wiederholungen von $Cut_t(G)$

Wie t wählen? $\rightarrow$ t ist optimal falls $\frac{n^4}{t^2} = n^2 \cdot t^2$

$\downarrow \cdot t^2$

$\Leftrightarrow n^4 = n^2 t^4$

$\downarrow \cdot n^2$

$\Leftrightarrow n^2 = t^4$

$\downarrow \sqrt{}$

$\Leftrightarrow t = \sqrt{n}$ ergibt Laufzeit $O\left(\lambda \left(\frac{n^4}{\sqrt{n}^2} + n^2 \sqrt{n}^2\right)\right) = O(\lambda n^3)$

Bootstrapping können wir beliebig oft machen, und erhalten im Limit die Laufzeit

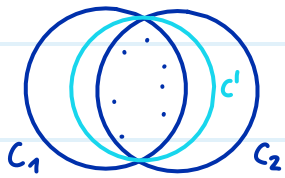
$O(n^2 \cdot \text{poly}(\log n))$

Smallest enclosing disk

Gegeben: eine endliche Punktemenge $P \subseteq \mathbb{R}^2$

Gesucht: der kleinste Kreis, der P umschliesst. (Punkte auf dem Rand sind erlaubt)

Eindeutigkeit: Wir nehmen per Widerspruch an, dass $C_1 \neq C_2$ zwei kleinste umschliessende Kreise sind. Dann können wir einen noch kleineren Kreis C' bestimmen, was ein Widerspruch zur Annahme ist, dass C_1 und C_2 kleinstmöglich sind

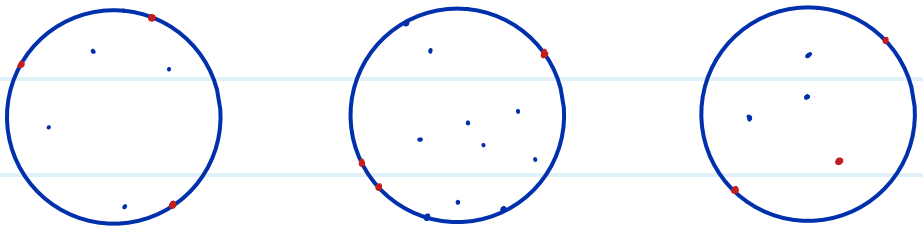


Lemma 3.26: Für jede Punktemenge P mit $|P| \geq 3$ gibt es eine Teilmenge $Q \subseteq P$, sodass $|Q| = 3$ und $\underbrace{C(P) = C(Q)}$

kleinster Kreis von P = kleinster Kreis von Q

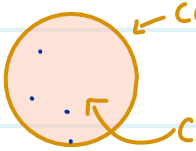
Q ist ein Zertifikat für die Minimalität des kleinsten Kreises

Q = rote Punkte



↳ Q muss nicht eindeutig sein

$C(P)$ vs $C^\bullet(P)$



$C(P)$ = Randlinie

$C^\bullet(P)$ = gesamte Fläche inklusive Randlinie

Brute Force Algorithmus:

CompleteEnumeration(P) in $O(n^4)$

```
1: for all Q ⊆ P mit |Q| = 3 do
2:     bestimme C(Q)
3:     if P ⊆ C•(Q) then
4:         return C(Q)
```

$\binom{n}{3} = \frac{n\,(n-1)\,(n-2)}{3\,2} \leq O(n^3)$ viele Möglichkeiten

in $O(1)$

in $O(n)$, prüfen ob alle n Punkte aus P in der Kreisscheibe $C^\bullet(Q)$ enthalten sind

Las-Vegas Algorithmus:

Randomised_CleverVersion(P) in $O(n \log n)$

```
1: repeat forever
2:     wähle Q ⊆ P mit |Q| = 11 zufällig und gleichverteilt
3:     bestimme C(Q)
4:     if P ⊆ C•(Q) then
5:         return C(Q)
6:     verdoppele alle Punkte von P ausserhalb von C(Q)
```

in $O(n)$ mit Hilfe von Zählern

in $O(1)$

in $O(n)$

Den Beweis für die Laufzeit lassen wir aus Zeitgründen weg

Lemma 3.28: Sei P' eine Multimenge mit $|P'| = N$ Punkten

Sei $r \leq N$ und sei $R \in \binom{P'}{r}$ zufällig gleichverteilt (es gilt $|R| = r$ und $R \subseteq P'$)

Sei $X := |P' \setminus C^\bullet(R)|$ die Anzahl Punkte ausserhalb von $C(R)$

Dann ist $\mathbb{E}[X] \leq 3 \cdot \frac{N}{r+1}$

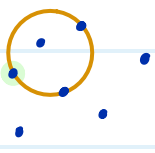
Bsp. $|P'| = 7$

$r = 4$

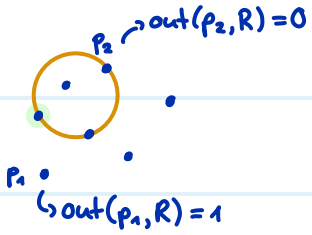
R

C(R)

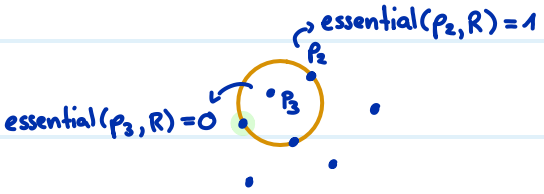
$|X| = 3$



Beweis: wir definieren: $\text{out}(p,R) := \begin{cases} 1 & \text{falls } p \notin C^\bullet(R) \\ 0 & \text{sonst} \end{cases}$



$$\text{essential}(p,Q) := \begin{cases} 1 & \text{falls } C(Q \setminus \{p\}) \neq C(Q) \\ 0 & \text{sonst} \end{cases}$$



Für $p \notin R$ gilt: $\text{out}(p,R) = 1 \iff \text{essential}(p, R \cup \{p\}) = 1$

Sei $\Omega = \binom{P'}{r}$ ein Laplace-Raum

$$\mathbb{E}[X] = \sum_{R \in \binom{P'}{r}} Pr[R] \; \mathbb{E}[X|R]$$

Satz für bedingten Erwartungswert

$$= \sum_{R \in \binom{P'}{r}} \frac{1}{\binom{N}{r}} \sum_{s \in P' \setminus R} \text{out}(s,R)$$

$Pr[R] = \frac{1}{|\Omega|} = \frac{1}{\binom{N}{r}}$ und $\mathbb{E}[X|R] = \sum_{s \in P' \setminus R} \text{out}(s,R)$ zählt alle Punkte, die ausserhalb von R liegen

$$= \frac{1}{\binom{N}{r}} \sum_{R \in \binom{P'}{r}} \sum_{s \in P' \setminus R} \text{out}(s,R)$$

Distributivgesetz

$$= \frac{1}{\binom{N}{r}} \sum_{R \in \binom{P'}{r}} \sum_{s \in P' \setminus R} \text{essential}(s, R \cup \{s\})$$

weil $\text{out}(s,R) = \text{essential}(s, R \cup \{s\})$

$$= \frac{1}{\binom{N}{r}} \sum_{Q \in \binom{P'}{r+1}} \sum_{p \in Q} \text{essential}(p,Q)$$

weil $R \in \binom{P'}{r}$ plus einen weiteren Punkt $s \in P' \setminus R$ gleich $Q \in \binom{P'}{r+1}$ ist. Wir ersetzen $R \cup \{s\}$ mit Q

$$\leq \frac{1}{\binom{N}{r}} \sum_{Q \in \binom{P'}{r+1}} 3$$

es gibt ≤ 3 essential points

$$= \frac{1}{\binom{N}{r}} \binom{N}{r+1} \cdot 3$$

$\left|\binom{P'}{r+1}\right| = \binom{N}{r+1}$ weil $|P'| = N$

$$= \frac{(N-r)! \cdot r!}{N!} \cdot \frac{N!}{(N-(r+1))! \cdot (r+1)!} \cdot 3$$

def. Binomialkoeffizient

$$= 3 \frac{N-r}{r+1} \leq 3 \frac{N}{r+1}$$

kürzen

∎

Flow Aufgaben

Ein Datennetzwerk $N = (V, A, c, s, t)$ besteht aus n Servern und einigen gerichteten Kabelverbindungen dazwischen. Ein Hauptserver s möchte Datenpakete an einen Zielserver t senden. Jedes Kabel kann maximal c Pakete pro Sekunde weiterleiten. Allerdings haben auch die Server selbst eine begrenzte Rechenleistung: Der Server i kann pro Sekunde insgesamt maximal v_i Datenpakete verarbeiten und weiterleiten (egal woher sie kommen und wohin sie gehen).

Modelliere den maximalen Datenfluss von s nach t als ein maxflow Problem.

Wir erstellen ein modifiziertes Netzwerk $N' = (V', A', c', s', t')$ wobei

- $V' =$
- $A' =$
- $c'(x, y) =$
- $s' =$
- $t' =$

Ein Staudamm im Gebirge versorgt über ein Leitungssystem n Dörfer $d_1, \ldots, d_n$ mit Wasser. Es gibt eine separate Leitung ohne Kapazitätsgrenze zu jedem Dorf. Im Winter gibt der Staudamm insgesamt 100 Einheiten Wasser, aber im Sommer nur 10 Einheiten.

Dorf i benötigt im Winter w_i Einheiten Wasser und im Sommer s_i Einheiten Wasser. Dorf i hat auch ein Speicherbecken der Grösse b_i, worin Wasser vom Winter für den Sommer aufbewahrt werden kann.

Einige Dörfer sind miteinander befreundet, d.h. sie können im Sommer gegenseitig Wasserreserven austauschen.

Modelliere dieses Problem als ein maxflow Problem, um festzustellen, ob es eine Wasserzuteilung gibt, sodass alle Dörfer genug Wasser haben.