

Vorlesung

# Effiziente Algorithmen und Komplexitätstheorie

Sommersemester 2008

Ingo Wegener

# Schnittprobleme

## Erinnerung

- $Q$ - $S$ -Schnitte in Netzwerken
- Max Flow = Min Cut

## Eingabe

gewichteter, ungerichteter Graph  $G = (V, E)$

Kantengewichte  $w(e) \in \mathbb{N}$

(ungewichteter Graph  $\hat{=}$  alle Kantengewichte = 1)

## zulässige Lösung

Schnitt  $V_1, V_2$

mit  $V_1 \cup V_2 = V$

## Bewertung

$$w(V_1, V_2) = \sum_{e \in \{\{u,v\} | u \in V_1, v \in V_2\}} w(e)$$

## Min Cut

**Ziel** finde Schnitt mit minimalem Wert

mit  $V_1 \neq \emptyset$  und  $V_2 \neq \emptyset$

## Max Cut

**Ziel** findet Schnitt mit maximalem Wert

# Über Max Cut

klar    Max Cut  $\in \mathcal{NPO}$

bekannt

- zugehöriges Entscheidungsproblem NP-vollständig (Karp 1972)
- Max Cut hat eine 1,14-Approximation (Goemans und Williamson 1994)
- Max Cut hat keine 1,06-Approximation, wenn  $P \neq NP$  (Håstad 1996)

hier und heute    sehr einfacher randomisierter Algorithmus  
schnell, sehr leicht zu implementieren  
erwartete Approximationsgüte  $\leq 2$

# Ein randomisierter Max-Cut-Algorithmus

## Algorithmus 11.1

1.  $V_1 := V_2 := \emptyset$
2. Für  $v \in V$
3. Mit W'keit  $1/2$  setze  $V_1 := V_1 \cup \{v\}$   
sonst setze  $V_2 := V_2 \cup \{v\}$
4. Ausgabe  $V_1, V_2$

## Theorem 11.2

Algorithmus 11.1 berechnet für einen ungewichteten Graphen  $G = (V, E)$  in Zeit  $O(|V|)$  einen Schnitt, der im Erwartungswert  $|E|/2$  Kanten schneidet und erwartete Güte  $\leq 2$  hat.

## Zum Beweis von Theorem 11.2

Beobachtung Laufzeit  $O(|V|)$  ✓

Beobachtung erwartete Güte  $\leq 2$  ✓  
 folgt aus  $E(\# \text{geschnittene Kanten}) = |E|/2$   
 weil  $\text{OPT} \leq |E|$

noch offen  $E(\# \text{geschnittene Kanten}) = E(w(V_1, V_2)) = |E|/2$

Definiere Indikatorvariablen  $X_e$  für  $e \in E$

$$\text{mit } X_e := \begin{cases} 1 & \text{falls } |e \cap V_1| = |e \cap V_2| = 1 \\ 0 & \text{sonst} \end{cases}$$

Beobachtung  $w(v_1, V_2) = \sum_{e \in E} X_e$

# Analyse von Algorithmus 11.1

Wir haben Indikatorvariablen  $X_e$  für  $e \in E$

$$\text{mit } X_e := \begin{cases} 1 & \text{falls } |e \cap V_1| = |e \cap V_2| = 1 \\ 0 & \text{sonst} \end{cases}$$

$$\text{und damit } w(V_1, V_2) = \sum_{e \in E} X_e$$

$$\begin{aligned} \mathbb{E}(w(V_1, V_2)) &= \mathbb{E}\left(\sum_{e \in E} X_e\right) \\ &= \sum_{e \in E} \mathbb{E}(X_e) \\ &= \sum_{e \in E} \text{Prob}(X_e = 1) \\ &= |E| \cdot \text{Prob}(X_e = 1) \\ &= |E| \cdot \text{Prob}(|e \cap V_1| = |e \cap V_2| = 1) \end{aligned}$$

# Erwartete Anzahl geschnittener Kanten

Wir haben

$$\begin{aligned}
 & E(w(V_1, V_2)) \\
 = & |E| \cdot \text{Prob}(|e \cap V_1| = |e \cap V_2| = 1) \\
 = & |E| \cdot \text{Prob}(((u \in V_1) \wedge (v \in V_2)) \vee ((u \in V_2) \wedge (v \in V_1))) \\
 = & |E| \cdot (\text{Prob}((u \in V_1) \wedge (v \in V_2)) + \text{Prob}((u \in V_2) \wedge (v \in V_1))) \\
 = & |E| \cdot (\text{Prob}(u \in V_1) \cdot \text{Prob}(v \in V_2) + \text{Prob}(u \in V_2) \cdot \text{Prob}(v \in V_1)) \\
 = & |E| \cdot \left( \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} \right) = |E| \cdot \left( \frac{1}{4} + \frac{1}{4} \right) = \frac{|E|}{2}
 \end{aligned}$$

□

# Min Cut

bei Min Cut im Unterschied zu Max Cut  
Minimierung von  $w(V_1, V_2)$   
 $V_1 \neq \emptyset$  und  $V_2 \neq \emptyset$

Voraussetzung  $G$  zusammenhängend  
sonst Lösung mit  $w(V_1, V_2) = 0$  optimal  
und in Linearzeit deterministisch berechenbar

Erinnerung Berechnung minimaler  $Q$ - $S$ -Schnitt  $\in P$

klar  $\binom{|V|}{2}$  mögliche Wahlen für  $Q, S$

also Min Cut  $\in \mathcal{PO}$

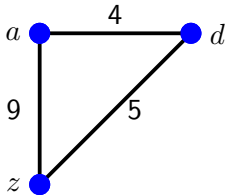
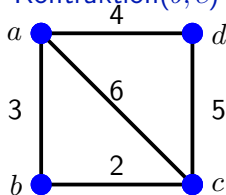
hier und jetzt einfache randomisierte Algorithmen  
mit  $W$ 'keit sehr dicht an 1 minimaler Schnitt  
sehr schnell

# Zentrale Methode Kontraktion( $u, v$ ) mit $u \neq v \in V$

Kontraktion( $u, v$ )

1.  $V := (V \setminus \{u, v\}) \cup \{z\}$ ;  $E := E \setminus \{\{u, v\}\}$
2. Für alle  $x \in V$  mit  $\{\{u, x\}, \{v, x\}\} \cap E \neq \emptyset$
3. If  $|\{\{u, x\}, \{v, x\}\} \cap E| = 2$  Then  
 $E := (E \setminus \{\{u, x\}, \{v, x\}\}) \cup \{\{x, z\}\}$   
 $w(\{x, z\}) := w(\{u, x\}) + w(\{v, x\})$
4. If  $\{u, x\} \in E$  Then  
 $E := (E \setminus \{\{u, x\}\}) \cup \{\{x, z\}\}$ ;  $w(\{x, z\}) := w(\{u, x\})$
5. If  $\{v, x\} \in E$  Then  
 $E := (E \setminus \{\{v, x\}\}) \cup \{\{x, z\}\}$ ;  $w(\{x, z\}) := w(\{v, x\})$

Kontraktion( $b, c$ )



# Min Cut durch Kontraktion

## Idee

1. Repeat
2. Wähle  $u \neq v \in V$  **geeignet**.
3. Kontraktion( $u, v$ )
4. Until  $|V| = 2$
5. Gib den durch die beiden Knoten definierten Schnitt aus.

**klar** Implementierung von „**geeignet**“ **entscheidend**

**Ziel** keine Kante zwischen min. Schnitt  $V_1, V_2$  kontrahieren

**Heuristik** Kanten mit großem Gewicht kontrahieren

**klar** „groß“ ist kontextabhängig

# Ein einfacher Min-Cut-Algorithmus

## Algorithmus 11.3 (Randomisierte Kontraktion)

1. Repeat
2. Wähle  $e = \{u, v\}$  mit W'keit  $w(e) / \sum_{e' \in E} w(e')$ .
3. Kontraktion( $u, v$ )
4. Until  $|V| = 2$
5. Gib den durch die beiden Knoten definierten Schnitt aus.

## Lemma 11.4

Algorithmus 11.3 hat Laufzeit  $O(|V|^2)$  auf einem Graphen  $G = (V, E)$ .

# Laufzeit des Algorithmus Randomisierte Kontraktion

## Beobachtung

- initial zusammenhängend
- also Wahl von  $e = \{u, v\}$  möglich
- Zusammenhang bleibt bei Kontraktion erhalten
- also Wahl von  $e = \{u, v\}$  immer möglich
- also korrekt ✓

## Beobachtung

- initial Summe der Kantengewichte in Zeit  $O(|E|) = O(|V|^2)$
- $|V| - 2$  Aufrufe von Kontraktion
- je Aufruf Zeit  $O(|V|)$ , da nur Nachbarn zu durchlaufen sind
- also Gesamtzeit  $O(|V|^2)$



# Min Cut: Struktureinsichten

**Notation**  $w(E') = \sum_{e \in E'} w(e)$   
 $\text{minCut}(G) = \min\{w(V_1, V_2) \mid V_1, V_2 \text{ Schnitt für } G\}$

## Lemma 11.5

Für alle zusammenhängenden, gewichteten, ungerichteten Graphen  $G = (V, E)$  gelten die folgenden Aussagen.

- 1 Ein Schnitt  $V_1, V_2$  ist genau dann Ausgabe von Algorithmus 11.3, wenn nie eine Kante zwischen  $V_1$  und  $V_2$  kontrahiert wurde.
- 2  $w(E) \geq \text{minCut}(G) \cdot |V| / 2$
- 3  $\forall e = \{u, v\} \in E: \text{minCut}(G) \leq \text{minCut}(\text{Kontraktion}(u, v))$

## Beweis von Lemma 11.5 – Teil 1

zu zeigen    Schnitt  $V_1, V_2$  ist Ausgabe von Algorithmus 11.3  
 $\Leftrightarrow$  keine Kante zwischen  $V_1$  und  $V_2$  wird kontrahiert

Betrachte     $V_1, V_2$  mit  $V_1 \cup V_2 = V$

Beobachtung    Kante zwischen  $V_1$  und  $V_2$  kontrahiert  
 $\Rightarrow u \in V_1$  und  $v \in V_2$  werden verschmolzen  
 $\Rightarrow V_1, V_2$  **nicht** Ausgabe von Algorithmus 11.3

Beobachtung    keine Kante zwischen  $V_1$  und  $V_2$  kontrahiert  
 $\Rightarrow V_1$  und  $V_2$  am Ende getrennt  
 $\Rightarrow$  Ausgabe  $V_1, V_2$



## Beweis von Lemma 11.5 – Teil 2

zu zeigen  $w(E) \geq \text{minCut}(G) \cdot |V| / 2$

Definiere für  $d_w(v) := \sum_{e=\{v,\cdot\} \in E} w(e)$  für  $v \in V$

Beobachtung  $\min\{d_w(v) \mid v \in V\} \geq \text{minCut}(G)$

denn falls  $\exists v \in V : d_w(v) < \text{minCut}(G)$

dann  $w(V_1, V_2) < \text{minCut}(G)$  mit  $V_1 := \{v\}$ ,  $V_2 := V \setminus V_1$

Beobachtung  $\sum_{v \in V} d_w(v) = 2w(E)$

also  $w(E) = \frac{1}{2} \sum_{v \in V} d_w(v) \geq \frac{1}{2} \cdot |V| \cdot \text{minCut}(G)$



## Beweis von Lemma 11.5 – Teil 3

zu zeigen  $\forall e = \{u, v\} \in E: \text{minCut}(G) \leq \text{minCut}(\text{Kontraktion}(u, v))$

**Beobachtung** durch Kontraktion können Schnitt verloren gehen  
aber **keine** hinzukommen

also Optimum kann höchstens schlechter werden

entspricht  $\text{minCut}(G) \leq \text{minCut}(\text{Kontraktion}(u, v))$



# Über „Randomisierte Kontraktion“

## Theorem 11.6

Für einen zusammenhängenden, gewichteten ungerichteten Graphen  $G = (V, E)$  sei  $V_1, V_2$  ein minimaler Schnitt.

Algorithmus 11.3 berechnet in Zeit  $O(|V|^2)$  mit

Wahrscheinlichkeit mindestens  $2/|V|^2$  den Schnitt  $V_1, V_2$ .

Beweis.

schon gesehen Laufzeit (Lemma 11.4) ✓

Definiere  $G_{i+1} = (V_{i+1}, E_{i+1})$  Ergebnis von  $i$ -ter Kontraktion

Betrachte Folge  $G = G_1, G_2, G_3, \dots, G_{|V|-1}$   
mit  $n_i := |V_i| = |V| - i + 1$  ( $n := |V| = n_1$ )

# Über die Kontraktion passender Kanten

**Notation**  $E' = \{\{u, v\} \mid u \in V_1, v \in V_2\}$

**Definiere** Ereignis  $A_i$ : in den ersten  $i - 1$  Kontraktionen keine Kante aus  $E'$  betroffen

damit

Prob (Algorithmus 11.3 berechnen  $V_1, V_2$ )

$$= \prod_{i=1}^{n-2} \text{Prob} (i\text{-te Kontraktion betrifft keine Kante aus } E' \mid A_i)$$

**Beobachtung**  $A_i \Rightarrow \text{minCut}(G) = \text{minCut}(G_i)$

**Beobachtung**  $\text{minCut}(G) = \text{minCut}(G_i)$   
 $\Rightarrow w(E_i) \geq \text{minCut}(G) \cdot |V_i| / 2$   
(Lemma 11.5)

## Kontraktion von Kanten aus $E'$

Wir brauchen

Prob ( $i$ -te Kontraktion betrifft keine Kante aus  $E' \mid A_i$ )

**Beobachtung** Gegenwahrscheinlichkeit leichter zu berechnen

$$\begin{aligned}
 & \text{Prob} (i\text{-Kontraktion betrifft Kante aus } E' \mid A_i) \\
 = & \sum_{e \in E'} \text{Prob} (i\text{-te Kontraktion betrifft } e \mid A_i) \\
 = & \sum_{e \in E'} \frac{w(e)}{w(E_i)} = \frac{w(E')}{w(E_i)} = \frac{\text{minCut}(G)}{w(E_i)} \\
 \leq & \frac{\text{minCut}(G)}{\text{minCut}(G) \cdot |V_i| / 2} = \frac{2}{|V| - i + 1} = \frac{2}{n - i + 1}
 \end{aligned}$$

## Beweis von Theorem 11.6

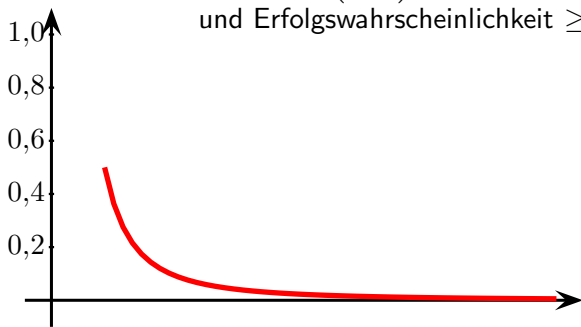
Wir haben  $\text{Prob}(\text{Algorithmus 11.3 berechnen } V_1, V_2)$   
 $= \prod_{i=1}^{n-2} \text{Prob}(i\text{-te Kontr. betrifft nicht } E' \mid A_i)$   
und  $\text{Prob}(i\text{-Kontraktion betrifft } E' \mid A_i) = \frac{2}{n-i+1}$

also

$$\begin{aligned} & \text{Prob}(\text{Algorithmus 11.3 berechnen } V_1, V_2) \\ &= \prod_{i=1}^{n-2} \text{Prob}(i\text{-te Kontraktion betrifft keine Kante aus } E' \mid A_i) \\ &= \prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right) = \prod_{i=1}^{n-2} \frac{n-i-1}{n-i+1} \\ &= \prod_{i=3}^n \frac{i-2}{i} = \frac{(n-2)!}{(n!)/2} = \frac{2}{n(n-1)} \geq \frac{2}{n^2} \end{aligned}$$

# Algorithmus „Randomisierte Kontraktion“ im Rückblick

Wir haben Laufzeit  $O(|V|^2)$  (prima)  
und Erfolgswahrscheinlichkeit  $\geq 2/n^2$  (Mist!)



Also Randomisierte Kontraktion nutzlos?

Erinnerung Probability Amplification

# Probability Amplification

## Lemma 11.7

Gibt man für einen Graphen  $G = (V, E)$  einen Schnitt mit kleinstem Wert unter  $r$  Ausgaben von unabhängigen Wiederholungen von Algorithmus 11.3 aus, so ist der ausgegebene Schnitt mit Wahrscheinlichkeit mindestens  $1 - \left(1 - 2/|V|^2\right)^r \geq 1 - e^{-(2r)/|V|^2}$  minimal.

**Beweis.**

**Theorem 11.6** je Durchlauf  $\text{Prob}(\text{min. Schnitt}) \geq 2/n^2$   
( $n = |V|$ )

**also** je Durchlauf  $\text{Prob}(\text{kein min. Schnitt}) \leq 1 - 2/n^2$

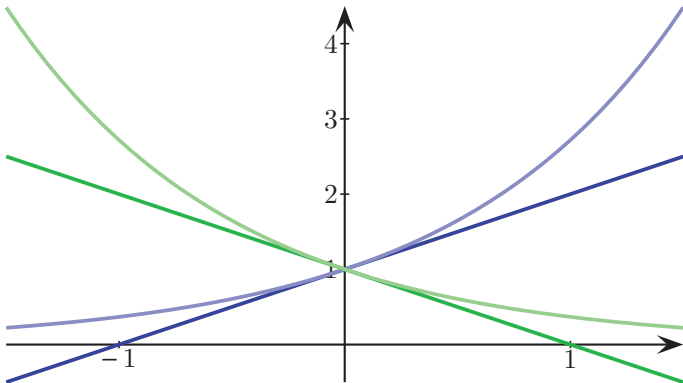
**also**  $\text{Prob}(\text{in } r \text{ Durchläufen nie min. Schnitt}) \leq \left(1 - 2/n^2\right)^r$

**also**  $\text{Prob}(\text{min. Schnitt in } r \text{ Durchläufen}) \geq 1 - \left(1 - 2/n^2\right)^r$

# Beweis von Lemma 11.7

Wir haben

$$\text{Prob}(\text{min. Schnitt in } r \text{ Durchläufen}) \geq 1 - (1 - 2/n^2)^r$$



$$\left(1 - 2/n^2 \leq e^{-2/n^2}\right) \Leftrightarrow \left((1 - 2/n^2)^r \leq e^{-2r/n^2}\right)$$



# Nützliche Randomisierte Kontraktion

## Korollar 11.8

Durch unabhängige Wiederholungen von Algorithmus 11.3 findet man für einen Graphen  $G = (V, E)$  mit  $|V| = n$  und jede Konstante  $c > 0$  in Zeit  $O(n^4 \log n)$  einen minimalen Schnitt mit Wahrscheinlichkeit mindestens  $1 - 1/n^{2c}$ .

**Beweis.**

**klar** mit Lemma 11.7

**Wähle**  $r := \lceil cn^2 \ln n \rceil$

**damit** Erfolgsw'keit  $\geq 1 - e^{-(2cn^2 \ln n)/n^2} = 1 - 1/n^{2c}$

**außerdem** Gesamtlaufzeit  $O(n^2 \cdot cn^2 \ln n) = O(n^4 \log n)$  □

**Beobachtung** schon mit  $c = 1/2$  Erfolgsw'keit **völlig ausreichend**

**Beobachtung** Laufzeit  $O(n^4 \log n)$  **nicht überzeugend**

# Laufzeit $n^4 \log n$ im Rückblick

Wieso brauchen wir so lange?

Zwei Faktoren

- 1 Laufzeit  $\Theta(n^2)$  für Randomisierte Kontraktion
- 2  $\Theta(n^2 \log n)$  Wiederholungen

Beobachtung  $\Theta(n^2)$  für Schnitt im Graphen **prima**

Wie können wir die Anzahl der Wiederholungen senken?

Wie können wir die Erfolgsw'keit je Durchgang verbessern?

## Lemma 11.9

Bricht man Algorithmus 11.3 auf einem Graphen  $G = (V, E)$  mit  $|V| = n$  ab, wenn  $|V| = t$  gilt (mit  $t \in \{2, 3, \dots, n\}$ ), so ist mit Wahrscheinlichkeit mindestens  $t \cdot (t - 1) / (n \cdot (n - 1))$  noch keine Kante eines festen minimalen Schnitts  $V_1, V_2$  kontrahiert.

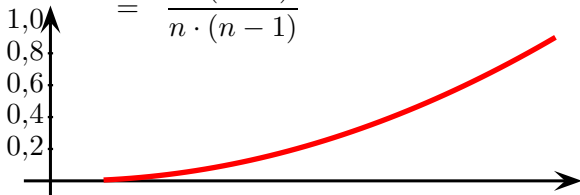
# Beweis von Lemma 11.9

Beweis.

analog zum Beweis von Theorem 11.6

Prob(Schnitt  $V_1, V_2$  noch möglich)

$$\begin{aligned}
 &\geq \prod_{i=1}^{n-t} \left( 1 - \frac{2}{n-i+1} \right) = \prod_{i=1}^{n-t} \frac{n-i-1}{n-i+1} \\
 &= \frac{(n-2) \cdot (n-3) \cdots (t+1) \cdot t \cdot (t-1)}{n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdots (t+2) \cdot (t+1)} \\
 &= \frac{t \cdot (t-1)}{n \cdot (n-1)}
 \end{aligned}$$



□

# Struktureinsicht und Verbesserungsidee

## Beobachtung

- anfangs Wahl **einfach**, Prob (wähle günstig) **groß**
- am Ende Wahl **schwierig**, Prob (wähle günstig) **klein**

## Ideen

- reduziere Knotenanzahl **rekursiv** um konstanten Faktor
- berechne auf jeder Rekursionsebene zwei Schnitte
- $\rightsquigarrow$  Anzahl Versuche wächst exponentiell mit Rekursionstiefe
- löse sehr kleine Schnittprobleme optimal

Dauern exponentiell viele Versuche nicht zu lange?

Vergrößert das die Erfolgswahrscheinlichkeit signifikant?

# Algorithmus Fast Cut

## Algorithmus 11.10

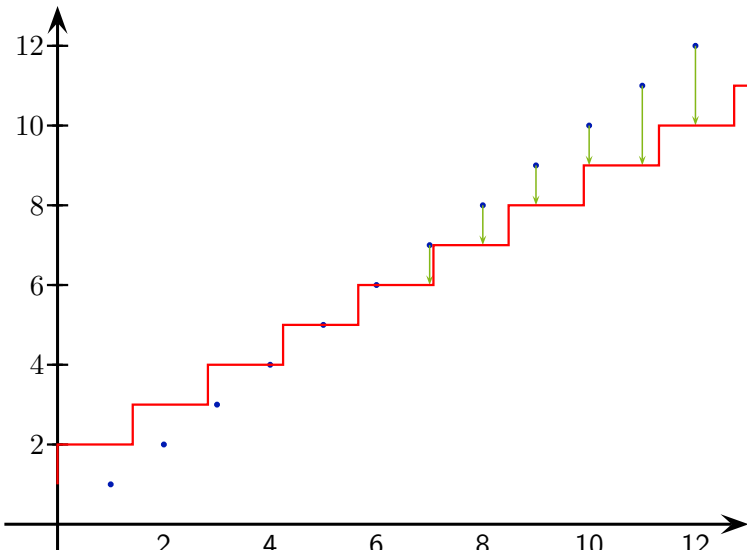
**Eingabe** zusammenhängender gewichteter Graph  $G = (V, E)$

**Ausgabe** Schnitt  $V_1, V_2$

1. If  $|V| \leq 6$  Then  
     Berechne minimalen Schnitt  $V_1, V_2$ .  
     Ausgabe  $V_1, V_2$
2. Else
3.    $t := \left\lceil 1 + \frac{|V|}{\sqrt{2}} \right\rceil$
4.    $H := \text{RandomisierteKontraktion}(G)$   
     bis  $t$  Knoten übrig sind.
5.    $H' := \text{RandomisierteKontraktion}(G)$   
     bis  $t$  Knoten übrig sind.
6.    $(V_1, V_2) := \text{FastCut}(H)$
7.    $(V'_1, V'_2) := \text{FastCut}(H')$
8.   Vergleiche  $(V_1, V_2)$  und  $(V'_1, V'_2)$ , gib kleineren Schnitt aus.

# Über die Wahl des Rekursionsendes

Wieso Ende der Rekursion gerade bei  $|V| \leq 6$ ?



# Laufzeit von Fast Cut

## Lemma 11.11

Algorithmus 11.10 hat auf einem Graphen  $G = (V, E)$  mit  $|V| = n$  Laufzeit  $O(n^2 \log n)$ .

**Beweis.**

**Definiere**  $T(n) =$  Laufzeit von Algo. 11.10 auf  $G$  mit  $n$  Knoten

**Erinnerung** Laufzeit von Algo 11.3 (Zeilen 4, 5)  $O(n^2)$   
**Notationsvereinfachung**  $\leq n^2/2$

**also**  $T(n) = 2 \cdot T\left(\left\lceil 1 + \frac{n}{\sqrt{2}} \right\rceil\right) + n^2$  für  $n > 6$   
 $T(n) = O(1)$  für  $n \leq 6$

**Notationsvereinfachung**  $T(n) = 1$  für  $n \leq 6$

# Analyse der Rekursion

**Betrachte** auftretende Knotenanzahlen

$$n_0 := n, n_1 := \left\lceil 1 + \frac{n_0}{\sqrt{2}} \right\rceil, n_2 := \left\lceil 1 + \frac{n_1}{\sqrt{2}} \right\rceil, \dots,$$

$$n_r := \left\lceil 1 + \frac{n_{r-1}}{\sqrt{2}} \right\rceil \text{ mit } n_r \leq 6$$

**damit**

$$\begin{aligned} T(n) &= T(n_0) = 2T(n_1) + n_0^2 \\ &= 2(2T(n_2) + n_1^2) + n_0^2 = 4T(n_2) + 2n_1^2 + n_0^2 \\ &= 8T(n_3) + 4n_2^2 + 2n_1^2 + n_0^2 \\ &= 2^r T(n_r) + \sum_{i=0}^{r-1} 2^i n_i^2 = 2^r + \sum_{i=0}^{r-1} 2^i n_i^2 \end{aligned}$$

**Wie entwickelt sich  $2^i n_i^2$ ?** klar  $2^{i+1}/2^i = 2$

**Beobachtung**

$$\begin{aligned} \frac{n_{i+1}^2}{n_i^2} &= \frac{\left\lceil 1 + \frac{n_i}{\sqrt{2}} \right\rceil^2}{n_i^2} \geq \frac{\left(1 + \frac{n_i}{\sqrt{2}}\right)^2}{n_i^2} = \frac{(n_i^2/2) + \sqrt{2}n_i + 1}{n_i^2} \\ &= \frac{1}{2} + \frac{\sqrt{2}}{n_i} + \frac{1}{n_i^2} > \frac{1}{2} \end{aligned}$$

**also**  $(2^{i+1} n_{i+1}^2) / (2^i n_i^2) > 2 \cdot (1/2) = 1$

# Laufzeitanalyse von Fast Cut

Wir haben  $T(n) = 2^r + \sum_{i=0}^{r-1} 2^i n_i^2$

$$n_0 := n, n_{i+1} := \left\lceil 1 + \frac{n_i}{\sqrt{2}} \right\rceil$$

$$\frac{2^{i+1} n_i^2}{2^i n_i^2} > 1$$

also  $2^i n_i^2$  maximal für  $i = r$

also  $T(n) = 2^r + \sum_{i=0}^{r-1} 2^i n_i^2$   
 $\leq 2^r + r \cdot 2^r n_r^2 \leq 2^r + r \cdot 2^r \cdot 36 = O(r \cdot 2^r)$

noch offen Rekursionstiefe  $r$

Plan Rekursionstiefe anhand von  $n_i$  bestimmen

## Eine Abschätzung für $n_i$

Wir haben  $n_0 := n, n_{i+1} := \left\lceil 1 + \frac{n_i}{\sqrt{2}} \right\rceil$

Notation  $q := 1/\sqrt{2}$

**Behauptung**  $n \cdot q^i + 2/(1 - q) \geq n_i$

Beweis mit vollständiger Induktion über  $i$

Induktionsanfang  $i = 0$

Beobachtung  $n \cdot q^0 + \frac{2}{1-q} = n + \frac{2}{1-q} > n = n_0$  ✓

Induktionsschritt

$$\begin{aligned} n \cdot q^{i+1} + \frac{2}{1-q} &= q \cdot \left( n \cdot q^i + \frac{2}{1-q} - \frac{2}{1-q} + \frac{2}{q - q^2} \right) \\ &= q \cdot \left( n \cdot q^i + \frac{2}{1-q} + \frac{2}{q} \right) \geq q \cdot \left( n_i + \frac{2}{q} \right) \\ &= 2 + n_i \cdot q \geq \lceil 1 + n_i \cdot q \rceil = n_{i+1} \quad \checkmark \end{aligned}$$

## Bestimmung der Rekursionstiefe

**Wir haben**  $n_i \leq n \cdot q^i + \frac{2}{1-q}$   
mit  $q = 1/\sqrt{2}$

**Wir suchen** kleinstes  $i$  mit  $n_i < 7$

$$\begin{aligned} n \cdot q^i + \frac{2}{1-q} &< 7 \\ \Leftrightarrow n \cdot q^i &< 7 - \frac{2}{1-q} \\ \Leftrightarrow \log(n) + i \log(q) &< \log\left(7 - \frac{2}{1-q}\right) \\ \Leftrightarrow -\frac{i}{2} &< \log\left(7 - \frac{2}{1-q}\right) - \log n \\ \Leftrightarrow i &> 2 \log(n) + 2 \log\left(7 - \frac{2}{1-q}\right) \end{aligned}$$

**also**  $r \leq 2 \log(n) + 6$

**Erinnerung**  $T(n) = O(r \cdot 2^r)$

**also**  $T(n) = O(n^2 \log n)$



# Erfolgswahrscheinlichkeit von Fast Cut

## Lemma 11.12

Algorithmus 11.10 berechnet auf einem Graphen  $G = (V, E)$  mit  $|V| = n$  einen minimalen Schnitt mit Wahrscheinlichkeit  $\Omega(1/\log n)$ .

**Beweis.**

klar für  $n \leq 6$  Erfolgsw'keit 1

**Definiere Ereignisse**

- $A(s)$  in  $H$  keine Kante zwischen  $V_1, V_2$  kontrahiert ( $s$  ist Knotenanzahl in  $G$ )
- $A'(s)$  in  $H'$  keine Kante zwischen  $V_1, V_2$  kontrahiert ( $s$  ist Knotenanzahl in  $G$ )
- $B(s)$  Fast Cut( $H$ ) liefert  $V_1, V_2$  unter der Annahme, dass das möglich ist ( $s$  ist Knotenanzahl in  $H$ )
- $B'(s)$  Fast Cut( $H'$ ) liefert  $V_1, V_2$  unter der Annahme, dass das möglich ist ( $s$  ist Knotenanzahl in  $H'$ )

## Beweis von Lemma 11.12

**Wir suchen** untere Schranke für  
 $\text{Prob}(B(n)) = \text{Prob}((A(n) \wedge B(t)) \vee (A'(n) \wedge B'(t)))$   
mit  $t = \left\lceil 1 + \frac{n}{\sqrt{2}} \right\rceil$

### Beobachtungen

- $A(n)$  und  $A'(n)$  unabhängig
- $B(t)$  und  $B'(t)$  unabhängig
- $A(n)$  und  $A'(n)$  gleichartig  $\rightsquigarrow \text{Prob}(A(n)) = \text{Prob}(A'(n))$
- $B(n)$  und  $B'(n)$  gleichartig  $\rightsquigarrow \text{Prob}(B(n)) = \text{Prob}(B'(n))$
- $(A(n), B(n))$  und  $(A'(n), B'(n))$  unabhängig

**damit**  $\text{Prob}(B(n)) = \text{Prob}((A(n) \wedge B(t)) \vee (A'(n) \wedge B'(t)))$   
etwas vereinfachen

# Analyse der Erfolgswahrscheinlichkeit von Fast Cut

Mit unseren Beobachtungen folgt

$$\begin{aligned}\text{Prob}(B(n)) &= \text{Prob}((A(n) \wedge B(t)) \vee (A'(n) \wedge B'(t))) \\ &= 1 - \text{Prob}(\overline{A(n) \wedge B(t)} \wedge \overline{A'(n) \wedge B'(t)}) \\ &= 1 - \text{Prob}(\overline{A(n) \wedge B(t)})^2 \\ &= 1 - (1 - \text{Prob}(A(n) \wedge B(t)))^2 \\ &= 1 - (1 - \text{Prob}(A(n)) \cdot \text{Prob}(B(t)))^2\end{aligned}$$

jetzt  $\text{Prob}(A(n))$  und  $\text{Prob}(B(t))$  getrennt abschätzen

# Abschätzen der beiden Algorithmenteile

Wir haben  $\text{Prob}(B(n)) = 1 - (1 - \text{Prob}(A(n)) \cdot \text{Prob}(B(t)))^2$

Erinnerung  $\text{Prob}(A(n)) \geq \frac{\lceil 1 + \frac{n}{\sqrt{2}} \rceil \cdot (\lceil 1 + \frac{n}{\sqrt{2}} \rceil - 1)}{n \cdot (n-1)}$   
aus Lemma 11.9

damit  $\text{Prob}(A(n)) \geq \frac{1}{2}$  für alle  $n$

also  $\text{Prob}(B(n)) \geq 1 - \left(1 - \frac{\text{Prob}(B(t))}{2}\right)^2$

Erinnerung Knotenanzahlen  $n_0, n_1, \dots, n_r$

## Noch eine Analyse einer Rekursion

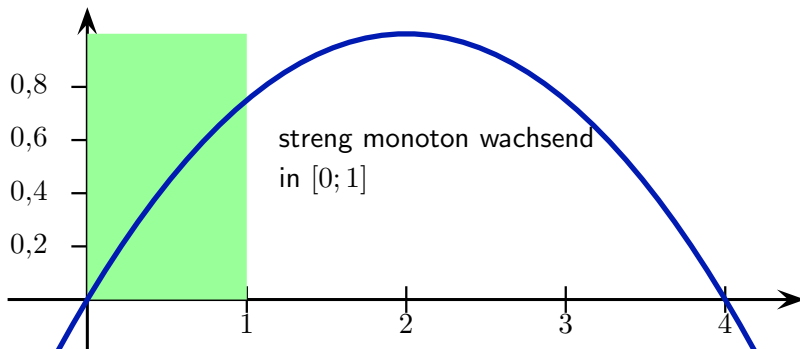
Wir haben  $\text{Prob}(B(n)) \geq 1 - \left(1 - \frac{\text{Prob}(B(t))}{2}\right)^2$

Definiere  $p(i) := \text{Prob}(B(n_{r-i}))$

Beobachtung  $p(0) = \text{Prob}(B(n_r)) = 1$

damit  $p(i+1) \geq 1 - \left(1 - \frac{p(i)}{2}\right)^2 = p(i) - \frac{p(i)^2}{4}$

Betrachte  $f(x) = x - \frac{x^2}{4}$



# Analyse von $p(i)$

Wir haben  $p(i+1) \geq p(i) - \frac{p(i)^2}{4}$

$$p(0) = 1$$

$f(x) = x - x^2/4$  streng monoton wachsend in  $[0; 1]$

**Behauptung**  $p(i) \geq \frac{1}{d} \Rightarrow p(i+1) \geq \frac{1}{d+1}$

**Beobachtung**  $p(i+1) \geq f(p(i))$

**Erinnerng**  $f$  streng monoton wachsend in  $[0; 1]$

also  $p(i) \geq \frac{1}{d} \Rightarrow p(i+1) \geq f(1/d)$

also **genügt zu zeigen**

$$p(i) \geq \frac{1}{d} \Rightarrow f(1/d) \geq \frac{1}{d+1}$$

## Nachrechnen der Behauptung

Wir behaupten  $p(i) \geq \frac{1}{d} \Rightarrow p(i+1) \geq \frac{1}{d+1}$

Es genügt zu zeigen  $p(i) \geq \frac{1}{d} \Rightarrow f(1/d) \geq \frac{1}{d+1}$

$$\begin{aligned} f\left(\frac{1}{d}\right) &\geq \frac{1}{d+1} \\ \Leftrightarrow \frac{1}{d} - \frac{1}{4d^2} &\geq \frac{1}{d+1} \\ \Leftrightarrow d + 1 - \frac{d+1}{4d} &\geq d \\ \Leftrightarrow 1 &\geq \frac{d+1}{4} \\ \Leftrightarrow d &\geq \frac{1}{3} \\ \Leftrightarrow \frac{1}{d} &\leq 3 \end{aligned}$$

Erinnerung  $p(i) \geq 1/d$  vorausgesetzt

klar  $p(i) \leq 1$  weil  $p(i)$  Wahrscheinlichkeit

also  $\frac{1}{d} \leq 1$ , erst recht  $\frac{1}{d} \leq 3$  ✓

# Abschätzung von $p(i)$

Wir haben  $p(0) = 1$   
 $(p(i) \geq \frac{1}{d}) \Rightarrow (p(i+1) \geq \frac{1}{d+1})$

also  $p(i) \geq \frac{1}{i+1}$  (induktiv)

also  $p(i) = \Omega(1/i)$

durch Einsetzen  $\text{Prob}(B(n_{r-r})) = \text{Prob}(B(n_0)) = \text{Prob}(B(n))$   
 $= p(r) = \Omega\left(\frac{1}{r}\right)$

Erinnerung  $r = O(\log n)$

zusammen Erfolgswahrscheinlichkeit  $\Omega\left(\frac{1}{\log n}\right)$  □

# Mit Probability Amplification

## Theorem 11.13

Unabhängige Wiederholungen von Algorithmus 11.10 (Fast Cut) finden für Graphen  $G = (V, E)$  mit  $|V| = n$  und jede Konstante  $c > 0$  in Zeit  $O(n^2 \log^3 n)$  einen minimalen Schnitt mit W'keit mindestens  $1 - 1/n^c$  und für jede Konstante  $\varepsilon > 0$  in Zeit  $O(n^2 \log^2 n)$  einen minimalen Schnitt mit W'keit mindestens  $1 - \varepsilon$ .

**Bew:** Erinnerung für Fast Cut gilt  $\exists k_1: T \leq k_1 \cdot n^2 \log n$  (Lemma 11.11)  
 $\exists k_2: \text{Prob}(\text{Erfolg}) \geq k_2 / \ln n$  (Lemma 11.12)

**Wähle** Anzahl unabhängiger Wiederholungen  
 $w := \lceil (c/k_2) \ln^2 n \rceil$

**Beobachtung** Gesamtrechenzeit  $O(n^2 \log^3 n)$  ✓

**Erfolgsw'keit**  $\geq 1 - \left(1 - \frac{k}{\ln n}\right)^w \geq 1 - e^{-c \ln n} = 1 - \frac{1}{n^c}$  ✓

**analog** zweite Aussage mit  $w := \lceil (1/k_2) \ln(1/\varepsilon) \ln n \rceil$

