4. Randomized Algorithms

Lec 8, 20.12.12

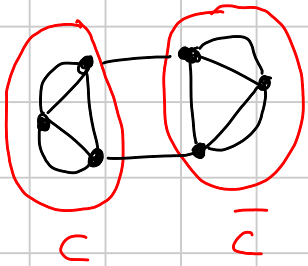
4.1 Def.

Multigraph $G = (V, \overset{\text{multi-set}}{E})$

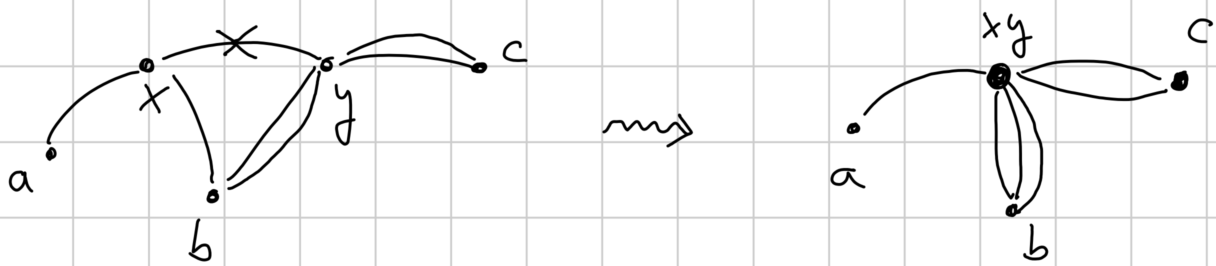
cut $:= \quad V = C \,\dot\cup\, \overline{C}$ with $C, \overline{C} \neq \emptyset$

$E(C, \overline{C}) := \{ \{x,y\} \in E : \ x \in C, \ y \in \overline{C} \}$

Min-Cut Problem: given multigraph G,
find a cut $(C, \overline{C})$ which minimizes $|E(C,\overline{C})|$.



If $\{x,y\} \in E$, then $\boxed{G/\{x,y\}}$ is defined as the graph obtained by $\boxed{\text{contracting}}$ the edge $\{x,y\}$:



4.2 Remark

The fastest known deterministic algorithms for the Min-Cut Problem have running time $O(mn \log(n^2/m))$.

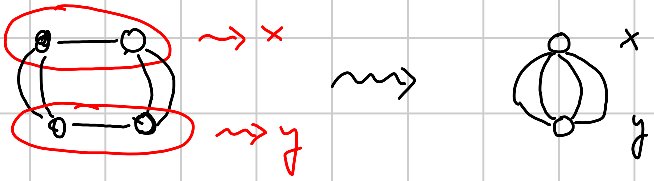
4.3 Algo CONTRACT

Input: multigraph $G=(V,E)$ output: Cut $(C,\overline{C})$.

CONTRACT (G)

1) while $|V| > 2$
2) choose an edge $e=\{x,y\}$ from G uniformly at random
3) $G := G/\{x,y\}$
4) denote by x and y the two remaining vertices of G
5) $C :=$ all vertices that were contracted into x
6) $\overline{C} :=$ .... y

Example:



Prop 4.4

a) Algo 4.3 performs exactly $n-2$ while-loops.
b) Algo 4.3 finds a cut in the input graph.
c) During the execution of Algo 4.3 the min cut of G does never decrease.
d) Let $(C,\overline{C})$ be a cut. Then
CONTRACT finds $(C,\overline{C})$ $\Leftrightarrow$ CONTRACT never chooses an edge from $E(C,\overline{C})$ in line 2)

Proof:

a), b), d): obvious

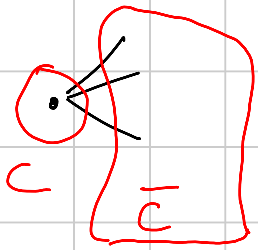
c): a cut in the contracted graph is always a cut in the original graph.

Thm 4.5

Algo 4.3 finds a min-cut with probability $\geq \frac{1}{\binom{n}{2}}$.

Proof: Let $(C, \bar{C})$ be a min-cut in G and set $k := |E(C, \bar{C})|$.

Then the minimum degree of G must satisfy $\delta(G) \geq k$, otherwise there would be a smaller cut:



$$\Rightarrow \quad |E| = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{1}{2} \cdot n \cdot k. \qquad (1)$$

Denote by $\mathcal{E}_i$ the event that „no edge of $E(C, \bar{C})$ is contracted in the i-th round" for $1 \leq i \leq n-2$.

Recall $\quad Pr[A \mid B] = Pr[A \cap B] / Pr[B]$

$\Rightarrow \quad Pr[A \cap B] = Pr[A \mid B] \cdot Pr[B]$

Similarly $\quad Pr[A \cap B \cap C] = Pr[A \mid B \cap C] \cdot Pr[B \mid C] \cdot Pr[C]$

and so on, and in general

$$Pr\left[\bigwedge_{i=1}^{n-2} \mathcal{E}_i\right] = \prod_{i=1}^{n-2} Pr\left[\mathcal{E}_i \,\middle|\, \bigwedge_{j=1}^{i-1} \mathcal{E}_j\right]$$

According to 4.4 d) it suffices to show

$$\frac{1}{\binom{n}{2}} \overset{!}{\leq} \prod_{i=1}^{n-2} Pr\left[\mathcal{E}_i \,\middle|\, \bigwedge_{j=1}^{i-1} \mathcal{E}_j\right] = Pr\left[\bigwedge_{i=1}^{n-2} \mathcal{E}_i\right]$$

For $i=1$: $\Pr[\bar{\mathcal{E}}_1] = \frac{k}{|E|} \overset{(1)}{\leq} \frac{k}{\frac{nk}{2}} = \frac{2}{n}$,

so $\Pr[\mathcal{E}_1] \geq 1 - \frac{2}{n}$.

In general: before the i-th contraction we have exactly $n_i := n-i+1$ vertices. If we condition on the event that so far no edge of $E(C,\bar{C})$ has been contracted, then due to 4.4 c) the min-cut still contains exactly k edges, so we get again

$$\Pr\left[\bar{\mathcal{E}}_i \,\middle|\, \bigwedge_{j=1}^{i-1} \mathcal{E}_j\right] \leq \frac{k}{\frac{n_i \cdot k}{2}} = \frac{2}{n_i}, \quad \text{thus}$$

$$\Pr\left[\mathcal{E}_i \,\middle|\, \bigwedge_{j=1}^{i-1} \mathcal{E}_j\right] \geq 1 - \frac{2}{n_i} = \frac{n_i - 2}{n_i} = \frac{n-i-1}{n-i+1}.$$

So in total

$$\prod_{i=1}^{n-2} \Pr\left[\mathcal{E}_i \,\middle|\, \bigwedge_{j=1}^{i-1} \mathcal{E}_j\right] \geq \prod_{i=1}^{n-2} \frac{n-i-1}{n-i+1} = \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdot \ldots \cdot \frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3}$$

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

qed

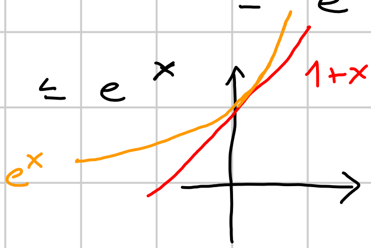
Note: obviously the error probability $\Pr[\bar{\mathcal{E}}_i \mid \bigwedge \mathcal{E}_j] \leq \frac{2}{n_i}$ gets larger for $i \to \infty$ and $n_i \searrow 2$.

Remark 4.6

a) Run Algo 4.3 τn^2 times for one given input, then the error probability decreases from $1 - \frac{1}{n^2}$ to

$$\left(1 - \frac{1}{n^2}\right)^{\tau n^2} \leq e^{-\frac{1}{n^2}\tau n^2} = e^{-\tau}.$$

$$1 + x \leq e^x$$



b) One contraction in Algo 4.3 can be implemented in $O(n)$, including the random choice of the edge. Hence Algo 4.3 has running time $O(n^2)$, and τn^2 repetitions will cost $O(\tau n^4)$ time in total.

Crucial idea:

The smaller the graph gets, the more repetitions we do.

Lemma 4.7

If we stop Algo 4.3 when the graph has exactly t vertices left, then the probability that an arbitrary but fixed min-cut survives until then is at least

$$\frac{\binom{t}{2}}{\binom{n}{2}}.$$

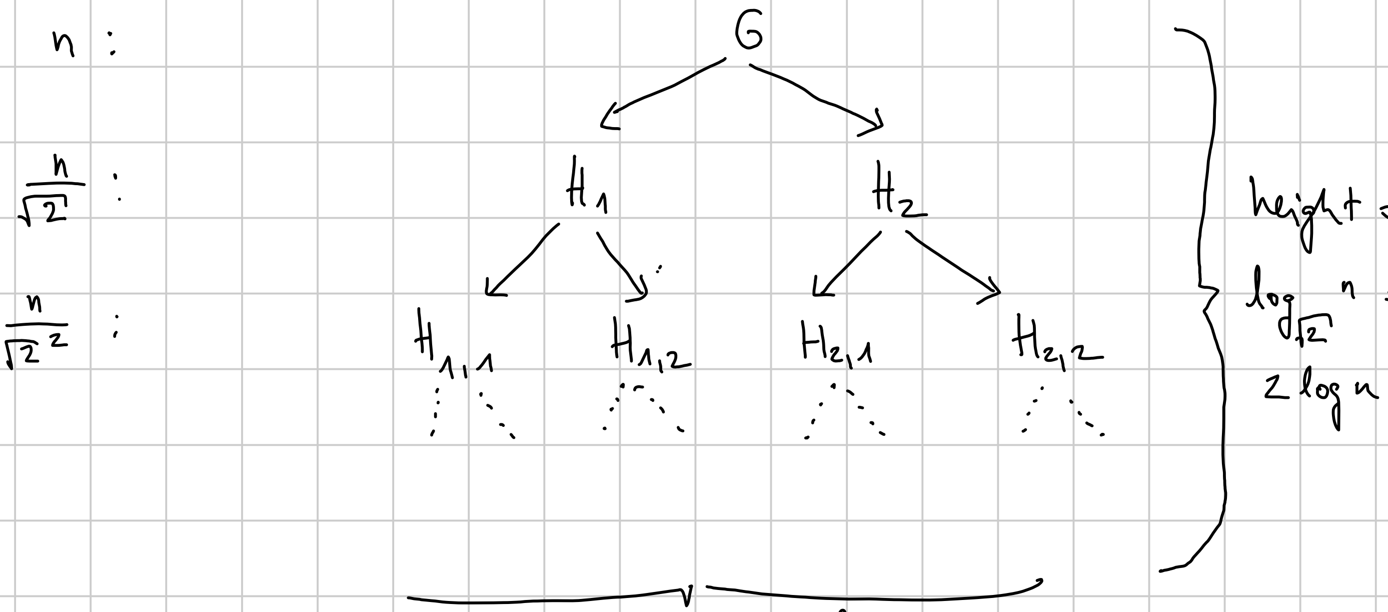
Proof: analogously to proof of Thm 4.5.

Algo 4.8 : FAST CUT

input: multigraph $G=(V,E)$, output: cut $(C,\overline{C})$.

FASTCUT(G)

1) if $n \leq 6$
2) then find min-cut by brute force
3) else
4) $t := \lceil 1 + \frac{n}{\sqrt{2}} \rceil$
5) start two independent runs of CONTRACT(G) which are stopped when two graphs H_1 and H_2 with exactly t vertices each are reached
6) $(C_1, \overline{C_1}) :=$ FASTCUT (H_1)
7) $(C_2, \overline{C_2}) :=$ FASTCUT (H_2)
8) return smaller of the two cuts



$\#$ leaves $= 2^{2\log n} = n^2$

Lemma 4.9 The running time of FASTCUT is $O(n^2 \log n)$.

Proof sketch / rest is homework:

$T(n) :=$ running time of FASTCUT for an n-vertex graph

$dn^2 \geq$ running time of CONTRACT "

$\Rightarrow \quad T(n) \leq 2 \cdot dn^2 + 2\,T\left(\left\lceil 1 + \frac{n}{\sqrt{2}} \right\rceil\right)$

now use this recursion to show that there exists a constant A such that $T(n) \leq A \cdot n^2 \log n$ by induction on n. ... qed

So the running time for FASTCUT is just a bit higher than that of CONTRACT ($n^2 \log n$ versus n^2), but hopefully the success probability will be much higher than $1/n^2$.

Thm 4.10 FASTCUT has a success probability of at least $\Omega(1/\log n)$.

Proof: homework.

Remark 4.11

Suppose the success probability for FASTCUT is $\geq \frac{c}{\log n}$.
Then do $\log^2 n$ independent runs for FASTCUT and obtain:

running time $O(n^2 \log^3 n)$

error probability $\leq \left(1 - \frac{c}{\log n}\right)^{(\log n)^2}$

$$\leq \left(e^{-\frac{c}{\log n}}\right)^{(\log n)^2} = e^{-c \log n} = n^{-c} = \frac{1}{n^c}$$