



Chapter 6

Randomization

Algorithm Theory
WS 2012/13

Fabian Kuhn

Last Lecture

Due to technical problems, we have no recordings 😞

We discussed 2 problems

- **Contention resolution:**

We will put the copy of a book chapter on the web
(beginning of Ch. 13 of book Algorithm Design by Kleinberg & Tardos)

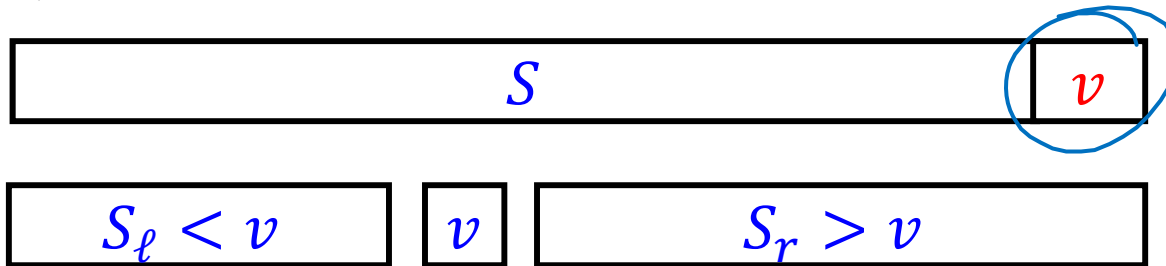
- **Miller-Rabin primality test:**

Use slides, colleagues, wikipedia page

– For the exam, the number-theoretic details are not relevant

Randomized Quicksort

Quicksort:



function Quick (S : sequence): sequence;

{returns the sorted sequence S }

begin

if $\#S \leq 1$ **then return** S

else { choose pivot element v in S ;

partition S into S_ℓ with elements $< v$,

and S_r with elements $> v$

return Quick(S_ℓ) v Quick(S_r)

end;

choose a uniformly random element

Randomized Quicksort Analysis

Randomized Quicksort: pick **uniform random** element as **pivot**

Running Time of sorting n elements:

- Let's just count the **number of comparisons**
- In the partitioning step, all $n - 1$ non-pivot elements have to be compared to the pivot



- **Number of comparisons:**

$$\underline{n - 1} + \underline{\text{\#comparisons in recursive calls}}$$

- If **rank of pivot** is r :
recursive calls with $\underline{r - 1}$ and $\underline{n - r}$ elements

Randomized Quicksort Analysis

Random variables:

$$C = n - 1 + C_\ell + C_r$$

- C : total number of comparisons (for a given array of length n)
- R : rank of first pivot
- C_ℓ, C_r : number of comparisons for the 2 recursive calls



$$\mathbb{E}[C] = n - 1 + \mathbb{E}[C_\ell] + \mathbb{E}[C_r]$$

Law of Total Expectation:

$$\begin{aligned} \mathbb{E}[C] &= \sum_{r=1}^n \mathbb{P}(R = r) \cdot \mathbb{E}[C | R = r] \\ &= \sum_{r=1}^n \mathbb{P}(R = r) \cdot (n - 1 + \mathbb{E}[C_\ell | R = r] + \mathbb{E}[C_r | R = r]) \end{aligned}$$

\uparrow exp. # comp. when sorting $r-1$ values \uparrow ... $n-r$ values

Randomized Quicksort Analysis

We have seen that:

$$\underbrace{\mathbb{E}[C]}_{T(n)} = \sum_{r=1}^n \underbrace{\mathbb{P}(R = r)}_{1/n} \cdot (n - 1 + \underbrace{\mathbb{E}[C_\ell | R = r]}_{T(r-1)} + \underbrace{\mathbb{E}[C_r | R = r]}_{T(n-r)})$$

Define:

- **$T(n)$** : expected number of comparisons when sorting n elements

$$\begin{aligned} \mathbb{E}[C] &= T(n) \\ \mathbb{E}[C_\ell | R = r] &= T(r - 1) \\ \mathbb{E}[C_r | R = r] &= T(n - r) \end{aligned}$$

Recursion:

$$\begin{aligned} T(n) &= \sum_{r=1}^n \frac{1}{n} \cdot (n - 1 + T(r - 1) + T(n - r)) \\ T(0) &= T(1) = 0 \end{aligned}$$

Randomized Quicksort Analysis

Theorem: The expected number of comparisons when sorting n elements using randomized quicksort is $T(n) \leq 2n \ln n$.

Proof:

$$0 = T(1) \leq 2 \cdot 1 \cdot \ln(1) = 0$$

$$T(n) = \sum_{r=1}^n \frac{1}{n} \cdot (n-1 + T(r-1) + T(n-r)), \quad T(0) = 0$$

$$= n-1 + \frac{1}{n} \cdot \sum_{i=0}^{n-1} (T(i) + T(n-1-i))$$

$$= n-1 + \frac{2}{n} \cdot \sum_{i=1}^{n-1} T(i) \leq 2 \cdot i \cdot \ln(i)$$

$$\leq n-1 + \frac{4}{n} \cdot \sum_{i=1}^{n-1} i \ln(i) \leq n-1 + \frac{4}{n} \cdot \int_1^n x \ln(x) dx$$

incr. with i



Randomized Quicksort Analysis

Theorem: The expected number of comparisons when sorting n elements using randomized quicksort is $T(n) \leq \underline{2n \ln n}$.

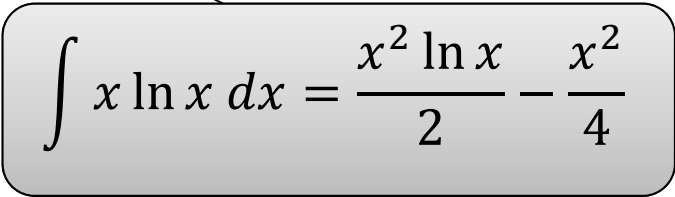
Proof:

$$T(n) \leq n - 1 + \frac{4}{n} \cdot \int_1^n x \ln x \, dx$$

$$T(n) \leq n - 1 + \frac{4}{n} \left(\frac{n^2 \ln n}{2} - \frac{n^2}{4} + \frac{1}{4} \right)$$

$$= n - 1 + 2n \ln n - n + \frac{1}{n}$$

$$= 2n \ln n + \frac{1}{n} - 1 \leq \underline{\underline{2n \ln n}}$$

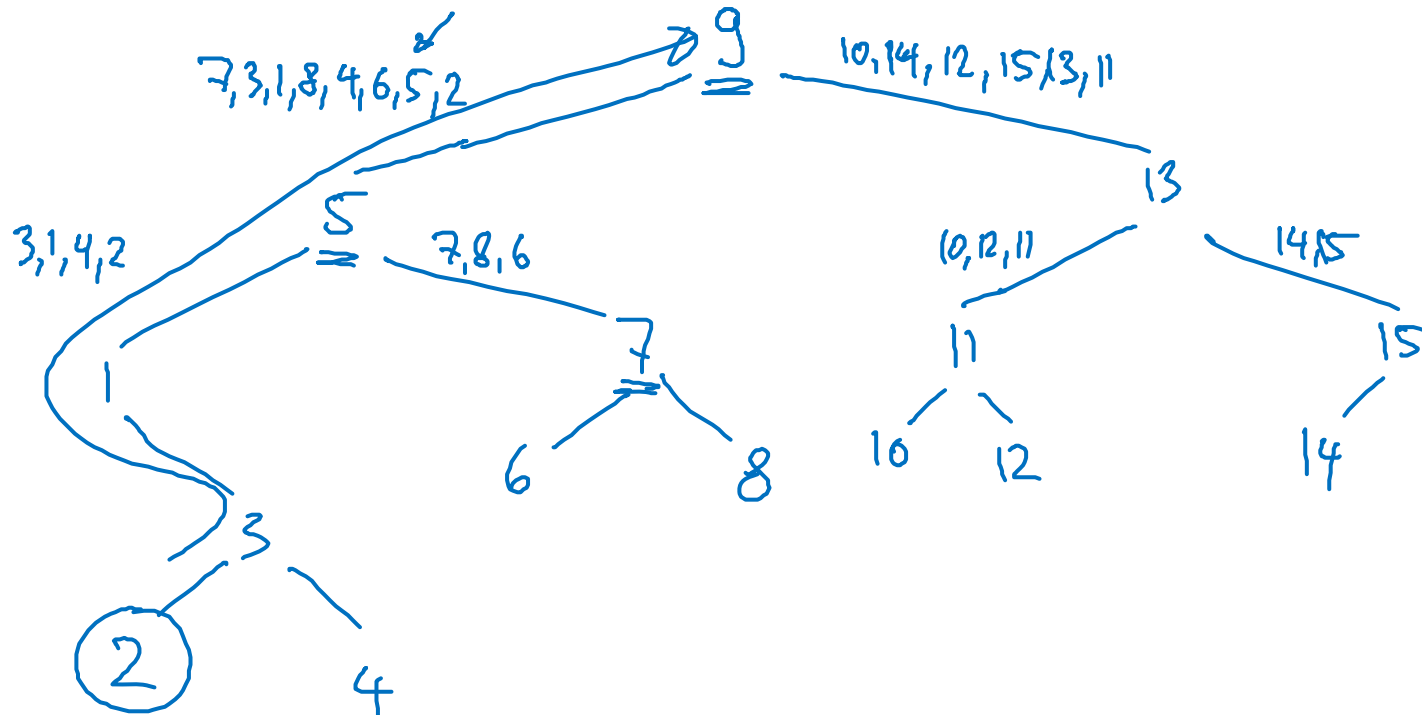


$$\int x \ln x \, dx = \frac{x^2 \ln x}{2} - \frac{x^2}{4}$$

Alternative Analysis

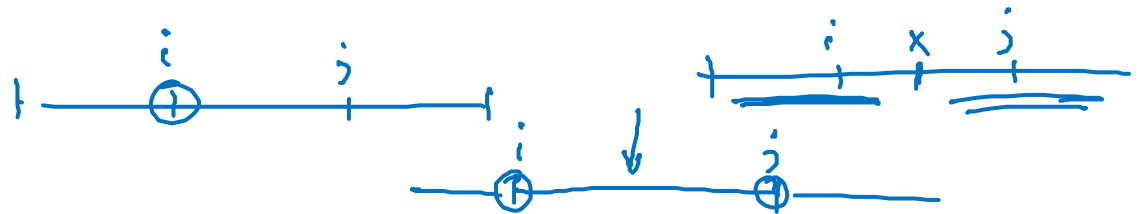
Array to sort: [7 , 3 , 1 , 10 , 14 , 8 , 12 , 9 , 4 , 6 , 5 , 15 , 2 , 13 , 11]

Viewing quicksort run as a **tree**:



Comparisons

- Comparisons are only between pivot and non-pivot elements
- Every element can only be the pivot once:
 → every 2 elements can only be compared once!
- W.l.o.g., assume that the elements to sort are 1, 2, ..., n
- Elements i and j are compared if and only if either i or j is a pivot before any element $h: i < h < j$ is chosen as pivot
 - i.e., iff i is an ancestor of j or j is an ancestor of i



$$\mathbb{P}(\text{comparison betw. } i \text{ and } j) = \frac{2}{j - i + 1}$$

Counting Comparisons

Random variable for every pair of elements (i, j) :

$$\underline{X_{ij}} = \begin{cases} 1, & \text{if there is a comparison between } i \text{ and } j \\ 0, & \text{otherwise} \end{cases}$$

$$\mathbb{P}(X_{ij} = 1) = \frac{j-2}{j-i+1} \quad \mathbb{E}[X_{ij}] = \frac{j-2}{j-i+1}$$

Number of comparisons: **X**

$$X = \sum_{i < j} X_{ij}$$

- What is $\mathbb{E}[X]$?

Randomized Quicksort Analysis

Theorem: The expected number of comparisons when sorting n elements using randomized quicksort is $T(n) \leq 2n \ln n$.

Proof:

- **Linearity of expectation:**

For all random variables X_1, \dots, X_n and all $a_1, \dots, a_n \in \mathbb{R}$,

$$\mathbb{E} \left[\sum_i^n a_i X_i \right] = \sum_i^n a_i \mathbb{E}[X_i].$$

$$\begin{aligned}
 X &= \sum_{i < j} X_{ij} & \mathbb{E}[X] &= \sum_{i < j} \mathbb{E}[X_{ij}] = \sum_{i < j} \frac{2}{j-i+1} \\
 & & &= \sum_{i=1}^{n-1} \sum_{j=i+1}^n \frac{2}{j-i+1}
 \end{aligned}$$

Randomized Quicksort Analysis

Theorem: The expected number of comparisons when sorting n elements using randomized quicksort is $T(n) \leq 2n \ln n$.

Proof:

$$\mathbb{E}[X] = 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^n \frac{1}{j-i+1} = 2 \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{1}{k}$$

$\leq H(n) - 1$

Harmonic series:

$$H(n) = \sum_{i=1}^n \frac{1}{i}$$

$$\underline{H(n) \leq 1 + \ln(n)}$$

$$\leq 2 \sum_{i=1}^{n-1} (H(n) - 1)$$

$$= 2(n-1)(H(n) - 1)$$

$$\leq \underline{\underline{2(n-1) \ln(n)}}$$

□

Types of Randomized Algorithms

Las Vegas Algorithm:

- always a **correct solution**
- **running time** is a **random** variable
- **Example:** randomized quicksort, contention resolution

Monte Carlo Algorithm:

- **probabilistic correctness** guarantee (mostly correct)
- fixed (deterministic) running time
- **Example:** primality test

Minimum Cut

Reminder: Given a graph $G = (V, E)$, a cut is a partition (A, B) of V such that $V = A \cup B$, $A \cap B = \emptyset$, $A, B \neq \emptyset$

Size of the cut (A, B) : # of edges crossing the cut



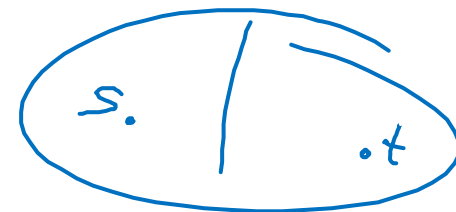
- For weighted graphs, total edge weight crossing the cut

Goal: Find a cut of minimal size (i.e., of size $\lambda(G)$)

edge connectivity

Maximum-flow based algorithm:

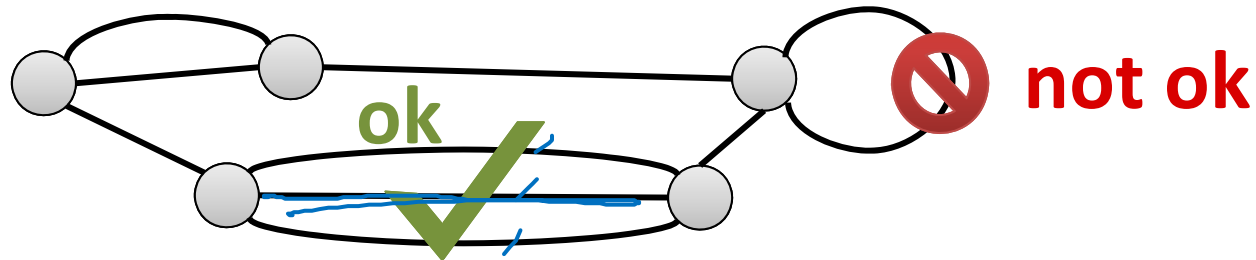
- Fix s , compute min s - t -cut for all $t \neq s$
- $O(m \cdot \lambda(G)) = O(mn)$ per s - t cut
- Gives an $O(mn\lambda(G)) = O(mn^2)$ -algorithm



Best-known deterministic algorithm: $O(mn + n^2 \log n) = O(n^3)$

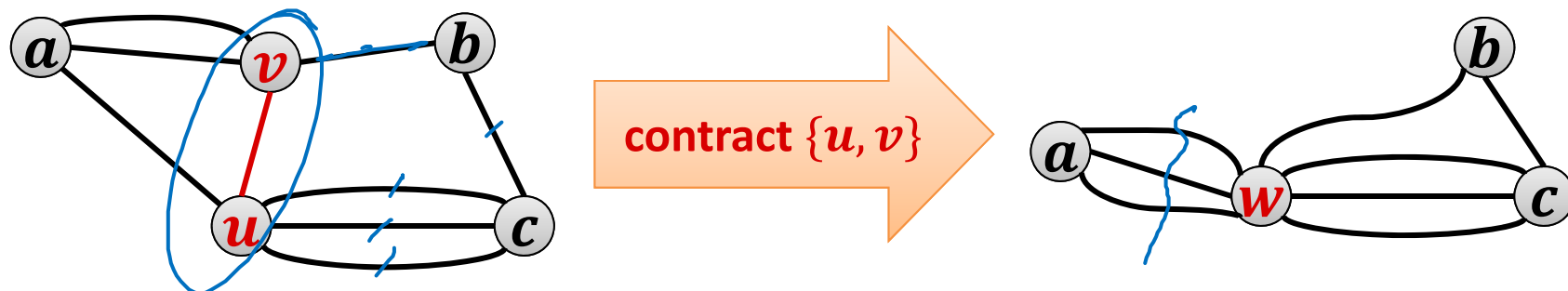
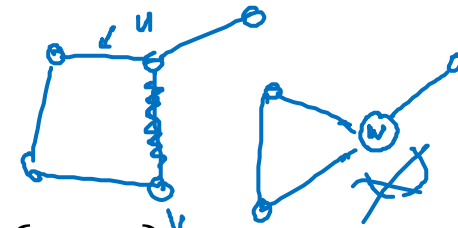
Edge Contractions

- In the following, we consider multi-graphs that can have multiple edges (but no self-loops)



Contracting edge $\{u, v\}$:

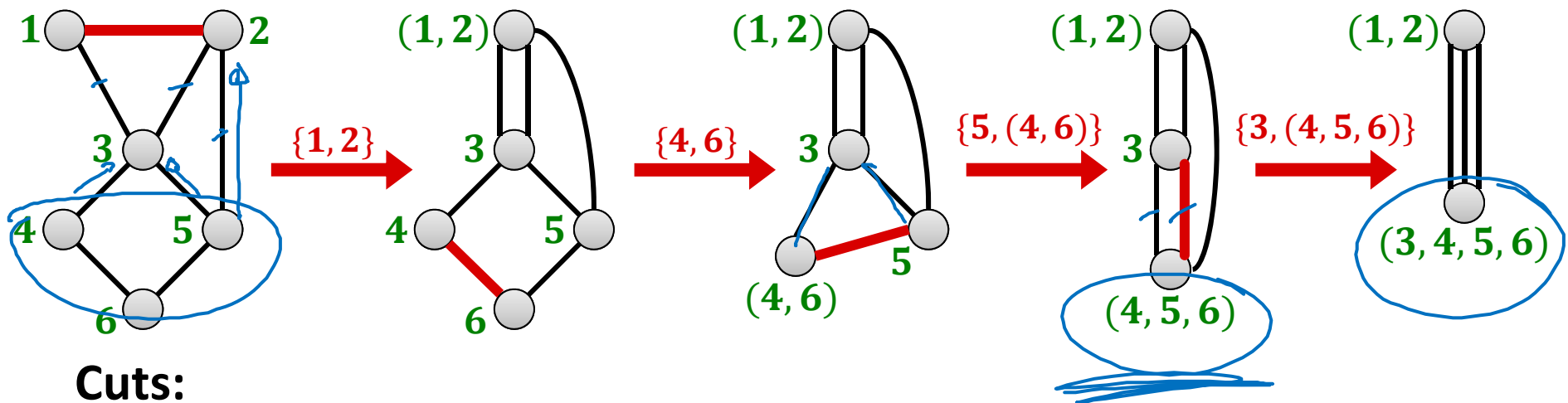
- Replace nodes u, v by new node w
- For all edges $\{u, x\}$ and $\{v, x\}$, add an edge $\{w, x\}$
- Remove self-loops created at node w



Properties of Edge Contractions

Nodes:

- After contracting $\{u, v\}$, the new node represents u and v
- After a series of contractions, each node represents a subset of the original nodes



Cuts:

- Assume in the contracted graph, w represents nodes $S_w \subset V$
- The edges of a node w in a contracted graph are in a one-to-one correspondence with the edges crossing the cut $(S_w, V \setminus S_w)$

Randomized Contraction Algorithm

Algorithm:

while there are > 2 nodes **do**

 contract a uniformly random edge

return cut induced by the last two remaining nodes

(cut defined by the original node sets represented by the last 2 nodes)



Theorem: The random contraction algorithm returns a minimum cut with probability at least $1/O(n^2)$.

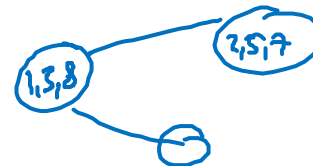
- We will show this next.

Theorem: The random contraction algorithm can be implemented in time $O(n^2)$.

- There are $n - 2$ contractions, each can be done in time $O(n)$.
- You will show this in the exercises.

Contractions and Cuts

Lemma: If two original nodes $u, v \in V$ are merged into the same node of the contracted graph, there is a path connecting u and v in the original graph s.t. all edges on the path are contracted.



Proof:

- Contracting an edge $\{x, y\}$ merges the node sets represented by x and y and does not change any of the other node sets.
- The claim follows by induction on the number of edge contractions.

Contractions and Cuts

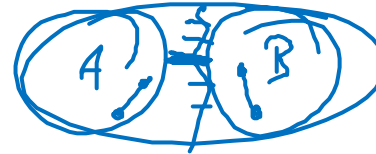
Lemma: During the contraction algorithm, the edge connectivity (i.e., the size of the min. cut) cannot get smaller.

Proof:

- All cuts in a (partially) contracted graph correspond to cuts of the same size in the original graph G as follows:
 - For a node u of the contracted graph, let S_u be the set of original nodes that have been merged into u (the nodes that u represents)
 - Consider a cut (A, B) of the contracted graph
 - (A', B') with

$$A' := \bigcup_{u \in A} S_u, \quad B' := \bigcup_{v \in B} S_v$$
 is a cut of G .
 - The edges crossing cut (A, B) are in one-to-one correspondence with the edges crossing cut (A', B') .

Contraction and Cuts



Lemma: The contraction algorithm outputs a cut (A, B) of the input graph G if and only if it never contracts an edge crossing (A, B) .

Proof:

1. If an **edge crossing (A, B) is contracted**, a pair of nodes $u \in A$, $v \in V$ is merged into the same node and the algorithm **outputs** a cut **different from (A, B)** .
2. If **no edge of (A, B) is contracted**, no two nodes $u \in A$, $v \in B$ end up in the same contracted node because every path connecting u and v in G contains some edge crossing (A, B)

In the end there are only 2 sets \rightarrow output is (A, B)

Getting The Min Cut

Theorem: The probability that the algorithm outputs a minimum cut is at least $\frac{2}{n(n-1)}$.

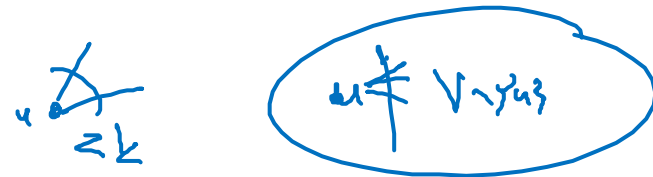
To prove the theorem, we need the following claim:

Claim: If the edge-count minimum cut size of a multigraph G (no self-loops) is k , G has at least $\frac{kn}{2}$ edges.

Proof:

- Min cut has size k \implies all nodes have degree $\geq k$
 - A node v of degree $< k$ gives a cut $(\{v\}, V \setminus \{v\})$ of size $< k$

- Number of edges $m = \frac{1}{2} \cdot \sum_v \deg(v)$ $\sum \deg(v) = 2m$

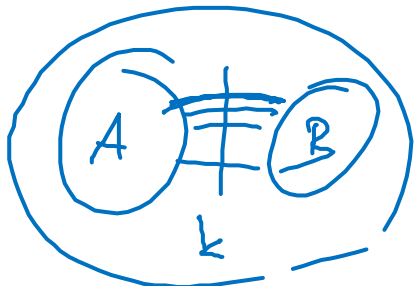


Getting The Min Cut

Theorem: The probability that the algorithm outputs a minimum cut is at least $2/n(n - 1)$.

Proof:

- Consider a fixed min cut (A, B) , assume (A, B) has size k
- The algorithm outputs (A, B) iff none of the k edges crossing (A, B) gets contracted.
- Before contraction i , there are $n + 1 - i$ nodes
 \rightarrow and thus $\geq (n + 1 - i)k/2$ edges
- If no edge crossing (A, B) is contracted before, the probability to contract an edge crossing (A, B) in step i is at most



$$\frac{\overset{k}{\circ}}{(n + 1 - i)k} = \frac{2}{\underline{\underline{n + 1 - i}}}$$

Getting The Min Cut

Theorem: The probability that the algorithm outputs a minimum cut is at least $2/n(n - 1)$.

Proof:

- If no edge crossing (A, B) is contracted before, the probability to contract an edge crossing (A, B) in step i is at most $2/n+1-i$.
- Event \mathcal{E}_i : edge contracted in step i is **not** crossing (A, B)

$$P(\mathcal{E}_i \mid \mathcal{E}_1 \wedge \dots \wedge \mathcal{E}_{i-1}) = 1 - \frac{2}{n+1-i}$$

↑
not
contr.
in first $i-1$ contr., no
edge across (A, B) contracted

Getting The Min Cut

Theorem: The probability that the algorithm outputs a minimum cut is at least $\frac{2}{n(n-1)}$.

Proof:

- $\mathbb{P}(\mathcal{E}_{i+1} | \mathcal{E}_1 \cap \dots \cap \mathcal{E}_i) = 1 - \frac{2}{n-i}$ $\mathbb{P}(\mathcal{E}_1) = \frac{2}{n}$
- No edge crossing (A, B) contracted: event $\mathcal{E} = \bigcap_{i=1}^{n-2} \mathcal{E}_i$

$$\begin{aligned}
 \mathbb{P}(\mathcal{E}) &= \mathbb{P}(\mathcal{E}_1) \cdot \mathbb{P}(\mathcal{E}_2 | \mathcal{E}_1) \cdot \mathbb{P}(\mathcal{E}_3 | \mathcal{E}_1 \cap \mathcal{E}_2) \cdot \dots \cdot \mathbb{P}(\mathcal{E}_{n-2} | \mathcal{E}_1 \cap \dots \cap \mathcal{E}_{n-3}) \\
 &= \frac{n-2}{n} \cdot \left(1 - \frac{2}{n-1}\right) \cdot \left(1 - \frac{2}{n-2}\right) \cdot \dots \cdot \left(1 - \frac{2}{3}\right) \\
 &= \frac{\cancel{n-2}}{n} \cdot \frac{\cancel{n-3}}{n-1} \cdot \frac{\cancel{n-4}}{n-2} \cdot \frac{\cancel{n-5}}{n-3} \cdot \dots \cdot \frac{2}{3} = \frac{2}{n(n-1)} = \frac{1}{\binom{n}{2}}
 \end{aligned}$$

Randomized Min Cut Algorithm

Theorem: If the contraction algorithm is independently repeated $O(n^2 \log n)$ times, one instance returns a minimum cut w.h.p.

Proof:

- Probability to not get a minimum cut in $c \cdot \binom{n}{2} \cdot \ln n$ iterations:

$$\left(1 - \frac{1}{\binom{n}{2}}\right)^{c \cdot \binom{n}{2} \cdot \ln n} < e^{-c \ln n} = \frac{1}{n^c}$$