Randomized Algorithms

- Lecturers
  - Peter Sanders, Building 46, R.308, sanders@mpi-sb.mpg.de
  - Ernst Althaus, Building 46, R.304,althaus@mpi-sb.mpg.de
- `http://www.mpi-sb.mpg.de/~sanders/courses/randalg/index.html`
- Literature
  - Motwani and Raghavan (Cambridge University Press, 1995)
  - Selected papers

Topics - Applications

- Hashing
- Linear Programming and Related Problems
- Min-Cut
- Minimum Spanning Trees
- Maximal Independent Sets
- Primality Testing

Topics - Tools and Techniques

- Today: Two Motivating Examples
- Basic Probability Theory
- Lower Bounding: Game Tree Evaluation
- Moments and Deviations
- Tail Inequalities and Randomized Rounding
- Expanding Graphs
- Markov Chains and Random Walks: Random Walks on Graphs and Cover Times
- Algebraic Techniques: Verifying Polynomial Identities and Equality of Strings.

Organization

Exercises :

- Every second week
- Participation is a prerequisite
- Time/Place $\longrightarrow$ next week

Credit:

- Exercises
- written or oral examination
  (depends on the number of participants)

Place: Move to 024 in building 46

Quick-sort

Input: a set S of n different numbers

Output: the numbers of S in increasing order

Algorithm: randQS(S)

1. Choose y form S uniformly at random
2. $S_1 := \{x \in S \mid x < y\}; S_2 := \{x \in S \mid x > y\}$
3. If $S_1 \neq \emptyset$ then randQS(S_1)
4. output y
5. If $S_2 \neq \emptyset$ then randQS(S_2)

Very Basic Probability Theory

Probability Space: A (finite) sample space Ω and
a probability measure Pr

Sample Space Ω: A set of elementary events

Probability measure Pr: A function $Pr : \mathcal{P}(\Omega) \to \mathbb{R}$ with

- For each $\mathcal{E} \subseteq \Omega, 0 \leq Pr[\mathcal{E}] \leq 1$
- $Pr[\Omega] = 1$
- For each $\mathcal{E} \subseteq \Omega$, $Pr[\mathcal{E}] = \sum_{\omega \in \mathcal{E}} Pr[\{\omega\}]$

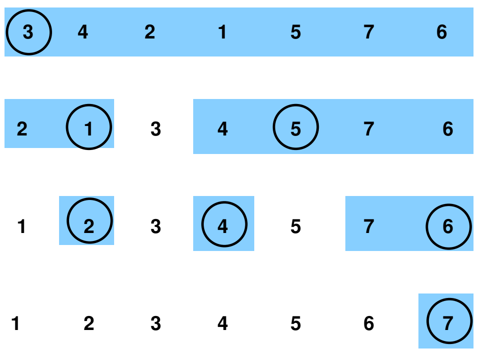
Random Variable X: a function $X : \Omega \to \mathbb{R}$.

Expectation of a random variable: $E[X] = \sum_{x \in \mathbb{R}} x \cdot Pr[X = x]$

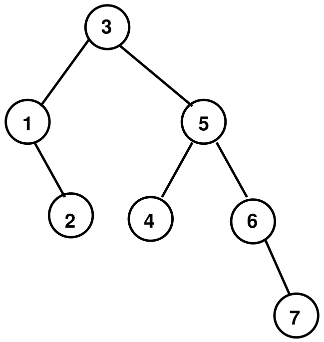
Fact: $E[X + Y] = E[X] + E[Y]$.

Illustration

execution



recursion tree T



Analyzing Quick-sort

- We want to count the number of comparisons we perform.
- Let s_i be the ith smallest element of S.
- We define random variables X_{ij} as

$$X_{ij} = \begin{cases} 1 & \text{if we compare } s_i \text{ and } s_j \\ 0 & \text{otherwise} \end{cases}$$

- The expected running time is

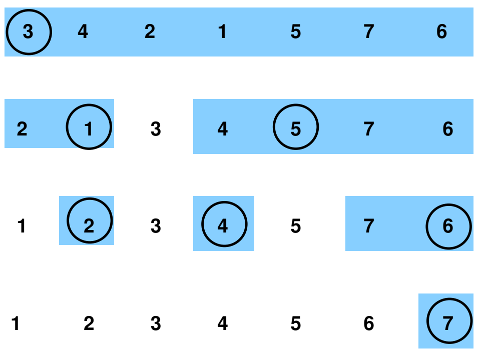
$$E[\sum_{i=1}^{n} \sum_{j=i+1}^{n} X_{ij}] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} E[X_{ij}]$$

- Thus we have to compute $E[X_{ij}]$.

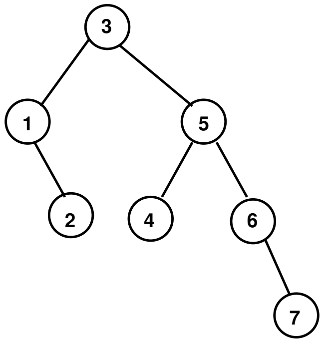
Recursion Tree

Lemma: There is a comparison between s_i and s_j $(i < j)$, iff s_i or s_j occurs earlier in π than any other element in $S_{ij} = \{s_i, s_{i+1}, \ldots s_j\}$.

execution



recursion tree T



Minimal Cuts

Definitions:

Cut in a connected (multi) graph G is a set of edges whose removal breaks G into two or more components

min-cut is a cut of minimum cardinality.

Contraction of (u, v) in a graph $G = (V, E)$ corresponds to

- replacing u and v by a new node w
- removing all edges between u and v
- replacing all other edges (u, y) or (v, y) by (w, y).

Finalizing the Analysis

Lemma: $Pr[X_{ij} = 1] = 2/(j - i + 1)$

$$\Rightarrow E[X_{ij}] = 0 \cdot Pr[X_{ij} = 0] + 1 \cdot Pr[X_{ij} = 1] = 2/(j - i + 1)$$

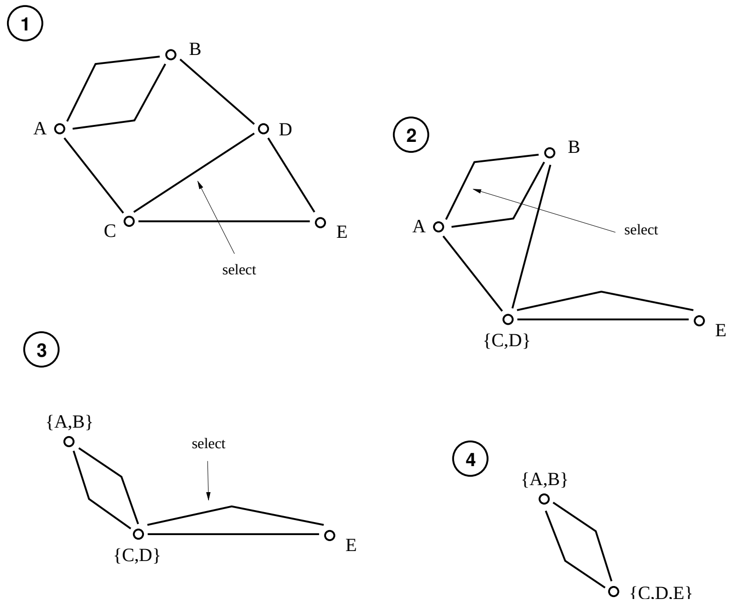
Lemma:

$$\Rightarrow \sum_{i=1}^{n} \sum_{j=i+1}^{n} E[X_{ij}] \quad \leq \quad 2nH_n \approx 2n \log n$$

A min-cut Algorithm

1. $H := G$
2. while H has more than two nodes do
3. choose an edge (u,v) uniformly at random from the edges in H.
4. contract (u, v) in H
5. let C be the edges of H
6. output C

Illustration



Analysis - I

What is the probability that we choose an edge of the minimal cut?

- Let K be a minimal cut and k its size.
- $\Rightarrow$ Every node has degree at least k.
- $\Rightarrow$ There are at least $kn/2$ edges.
- Let $\mathcal{E}_i$ be the event that we do *not* choose an edge of K in iter. i.
- Thus $Pr[\overline{\mathcal{E}_1}] \leq k/(kn/2) = 2/n$
- Assuming $\mathcal{E}_1$ happens $\Rightarrow$ in the second iteration
  the graph has $n-1$ nodes and K has the same size.
  $\Rightarrow \quad Pr[\overline{\mathcal{E}_2} \mid \mathcal{E}_1] \leq 2/(n-1)$
- The same argument gives us $Pr[\overline{\mathcal{E}}_i \mid \cap_{j=1}^{i-1} \mathcal{E}_j] \leq 2/(n-i+1)$

A bit more basic probability theory

Conditional Probability of $\mathcal{E}_1$ given $\mathcal{E}_2$:

$$Pr[\mathcal{E}_1 \mid \mathcal{E}_2] := \frac{Pr[\mathcal{E}_1 \cap \mathcal{E}_2]}{Pr[\mathcal{E}_2]}$$

Independence of $\{\mathcal{E}_i : i \in I\}$: if for all $S \subseteq I$

$$Pr[\cap_{i \in S} \mathcal{E}_1] = \prod_{i \in S} Pr[\mathcal{E}_i]$$

Fact: For every collection of events $\{\mathcal{E}_i : 1 \leq i \leq k\}$

$$Pr[\cap_{i=1}^{k} \mathcal{E}_i] = \prod_{i=1}^{k} Pr[\mathcal{E}_i \mid \cap_{j=1}^{i-1} \mathcal{E}_j].$$

Notation: $\bar{\mathcal{E}} := \Omega \setminus \mathcal{E}$

Analysis - II

- **Lemma**

$$Pr[C \text{ is a min-cut}] \quad \geq \quad \frac{2}{n(n-1)}$$

- Repeat the algorithm $n^2/2$ times. Since these runs are independent:

$$Pr[\text{Alg. fails in all runs}] \leq (1 - 2/n^2)^{n^2/2} < 1/e$$

- This algorithm can be implemented in $O(n^4)$
- Later in the course we will see a method to improve the algorithm: Runtime $O(n^2 \log^2 n)$ with constant failure probability.

Las Vegas - Monte Carlo

Las Vegas Algorithm:

- always gives the correct solution.
- running time is a random variable.
- Example: Quick-sort.

Monte Carlo Algorithm:

- may produce incorrect solutions.
- has a fixed deterministic running time.
- Example: Min-Cut Algorithm.
- For decision problems there are two kinds:

  one-sided errors: either *yes* or *no* answers always correct.

  two-sided errors: both have non-zero probability to be wrong.