BBM402-Lecture 12: Randomized Algorithms

Lecturer: Lale Özkahya

Resources for the presentation:
https://courses.engr.illinois.edu/cs473/fa2016/lectures.html

Outline

Randomization is very powerful

How do you play R-P-S?
Calculating insurance.

Outline

Randomization is very powerful

How do you play R-P-S?
Calculating insurance.

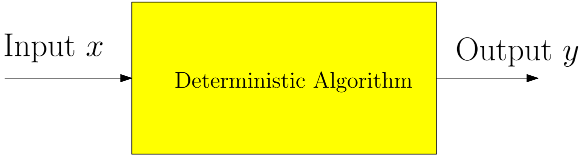
Our goal

- Basics of randomization – probability space, expectation, events, random variables, etc.
- Randomized Algorithms – Two types
  - Las Vegas
  - Monte Carlo
- Randomized Quick Sort

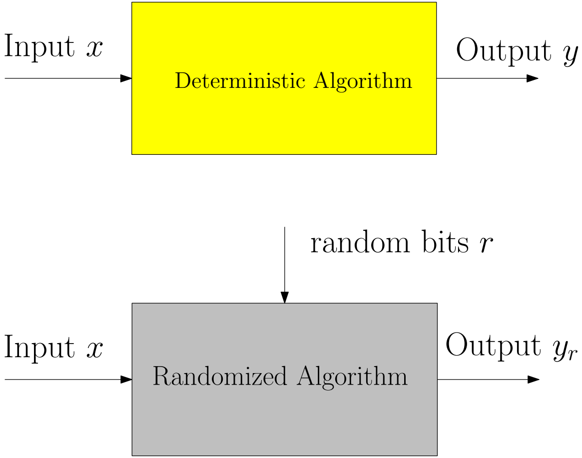
Part I

Introduction to Randomized Algorithms

Randomized Algorithms



Randomized Algorithms



Example: Randomized **QuickSort**

QuickSort ?

1. Pick a pivot element from array
2. Split array into 3 subarrays: those smaller than pivot, those larger than pivot, and the pivot itself.
3. Recursively sort the subarrays, and concatenate them.

Randomized **QuickSort**

1. Pick a pivot element **uniformly at random** from the array
2. Split array into 3 subarrays: those smaller than pivot, those larger than pivot, and the pivot itself.
3. Recursively sort the subarrays, and concatenate them.

Example: Randomized Quicksort

Recall: **QuickSort** can take $\Omega(n^2)$ time to sort array of size n.

Example: Randomized Quicksort

Recall: **QuickSort** can take $\Omega(n^2)$ time to sort array of size n.

Theorem

Randomized **QuickSort** *sorts a given array of length* n *in* $O(n \log n)$ *expected time.*

Example: Randomized Quicksort

Recall: **QuickSort** can take $\boldsymbol{\Omega(n^2)}$ time to sort array of size $\mathbf{n}$.

Theorem

Randomized **QuickSort** *sorts a given array of length* $\mathbf{n}$ *in* $\mathbf{O(n \log n)}$ *expected time.*

Note: On *every* input randomized **QuickSort** takes $\mathbf{O(n \log n)}$ time in expectation. On *every* input it may take $\boldsymbol{\Omega(n^2)}$ time with some small probability.

Example: Verifying Matrix Multiplication

Problem

Given three $\mathbf{n} \times \mathbf{n}$ matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ is $\mathbf{AB} = \mathbf{C}$?

Example: Verifying Matrix Multiplication

Problem

Given three $\mathbf{n} \times \mathbf{n}$ matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ is $\mathbf{AB} = \mathbf{C}$?

Deterministic algorithm:

1. Multiply $\mathbf{A}$ and $\mathbf{B}$ and check if equal to $\mathbf{C}$.
2. Running time?

Example: Verifying Matrix Multiplication

Problem

Given three $n \times n$ matrices A, B, C is $AB = C$?

Deterministic algorithm:

1. Multiply A and B and check if equal to C.
2. Running time? $O(n^3)$ by straight forward approach. $O(n^{2.37})$ with fast matrix multiplication (complicated and impractical).

Example: Verifying Matrix Multiplication

Problem

Given three $\mathbf{n} \times \mathbf{n}$ matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ is $\mathbf{AB} = \mathbf{C}$?

Example: Verifying Matrix Multiplication

Problem

Given three $\mathbf{n} \times \mathbf{n}$ matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ is $\mathbf{AB} = \mathbf{C}$?

Randomized algorithm:

1. Pick a random $\mathbf{n} \times \mathbf{1}$ vector $\mathbf{r}$.
2. Return the answer of the equality $\mathbf{ABr} = \mathbf{Cr}$.
3. Running time?

Example: Verifying Matrix Multiplication

Problem

Given three $\mathbf{n} \times \mathbf{n}$ matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ is $\mathbf{AB} = \mathbf{C}$?

Randomized algorithm:

1. Pick a random $\mathbf{n} \times \mathbf{1}$ vector $\mathbf{r}$.
2. Return the answer of the equality $\mathbf{ABr} = \mathbf{Cr}$.
3. Running time? $\mathbf{O(n^2)}$!

Example: Verifying Matrix Multiplication

Problem

Given three $\mathbf{n} \times \mathbf{n}$ matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ is $\mathbf{AB} = \mathbf{C}$?

Randomized algorithm:

1. Pick a random $\mathbf{n} \times \mathbf{1}$ vector $\mathbf{r}$.
2. Return the answer of the equality $\mathbf{ABr} = \mathbf{Cr}$.
3. Running time? $\mathbf{O(n^2)}$!

Theorem

If $\mathbf{AB} = \mathbf{C}$ then the algorithm will always say YES. If $\mathbf{AB} \neq \mathbf{C}$ then the algorithm will say YES with probability at most $\mathbf{1/2}$. Can repeat the algorithm $\mathbf{100}$ times independently to reduce the probability of a false positive to $\mathbf{1/2^{100}}$.

Why randomized algorithms?

1. Many many applications in algorithms, data structures and computer science!
2. In some cases only known algorithms are randomized or randomness is provably necessary.
3. Often randomized algorithms are (much) simpler and/or more efficient.
4. Several deep connections to mathematics, physics etc.
5. ...
6. Lots of fun!

Average case analysis vs Randomized algorithms

Average case analysis:

1. Fix a deterministic algorithm.
2. Assume inputs comes from a probability distribution.
3. Analyze the algorithm's *average* performance over the distribution over inputs.

Randomized algorithms:

1. Algorithm uses random bits in addition to input.
2. Analyze algorithms *average* performance over the given input where the average is over the random bits that the algorithm uses.
3. On each input behaviour of algorithm is random. Analyze worst-case over all inputs of the (average) performance.

Part II

Basics of Discrete Probability

Discrete Probability

We restrict attention to finite probability spaces.

Definition

A discrete probability space is a pair $(\Omega, \mathbf{Pr})$ consists of finite set Ω of **elementary events** and function $\mathbf{p} : \Omega \to [0, 1]$ which assigns a probability $\mathbf{Pr}[\omega]$ for each $\omega \in \Omega$ such that $\sum_{\omega \in \Omega} \mathbf{Pr}[\omega] = 1$.

Discrete Probability

We restrict attention to finite probability spaces.

Definition

A discrete probability space is a pair $(\Omega, \Pr)$ consists of finite set Ω of **elementary events** and function $p : \Omega \to [0, 1]$ which assigns a probability $\Pr[\omega]$ for each $\omega \in \Omega$ such that $\sum_{\omega \in \Omega} \Pr[\omega] = 1$.

Example

An unbiased coin. $\Omega = \{H, T\}$ and $\Pr[H] = \Pr[T] = 1/2$.

Example

A 6-sided unbiased die. $\Omega = \{1, 2, 3, 4, 5, 6\}$ and $\Pr[i] = 1/6$ for $1 \leq i \leq 6$.

Events

Definition

Given a probability space $(\Omega, \Pr)$ an **event** is a subset of Ω. In other words an event is a collection of elementary events. The probability of an event A, denoted by $\Pr[A]$, is $\sum_{\omega \in A} \Pr[\omega]$.

The **complement event** of an event $A \subseteq \Omega$ is the event $\Omega \setminus A$ frequently denoted by $\bar{A}$.

Events

Definition

Given a probability space $(\Omega, \Pr)$ an **event** is a subset of Ω. In other words an event is a collection of elementary events. The probability of an event A, denoted by $\Pr[A]$, is $\sum_{\omega \in A} \Pr[\omega]$.

The **complement event** of an event $A \subseteq \Omega$ is the event $\Omega \setminus A$ frequently denoted by $\bar{A}$.

Example

A pair of independent dice. $\Omega = \{(i,j) \mid 1 \leq i \leq 6, 1 \leq j \leq 6\}$.
Let A be the event that the sum of the two numbers on the dice is even.
Then $A = \left\{(i,j) \in \Omega \,\middle|\, (i+j) \text{ is even}\right\}$.
$\Pr[A] = |A|/36 = 1/2$.

Independent Events

Definition

Given a probability space $(\Omega, \Pr)$ and two events A, B are **independent** if and only if $\Pr[A \cap B] = \Pr[A] \Pr[B]$. Otherwise they are *dependent*. In other words A, B independent implies one does not affect the other.

Independent Events

Definition

Given a probability space $(\Omega, \Pr)$ and two events A, B are **independent** if and only if $\Pr[A \cap B] = \Pr[A]\Pr[B]$. Otherwise they are *dependent*. In other words A, B independent implies one does not affect the other.

Example

Two coins. $\Omega = \{HH, TT, HT, TH\}$ and $\Pr[HH] = \Pr[TT] = \Pr[HT] = \Pr[TH] = 1/4$.

1. A is the event that the first coin is heads and B is the event that second coin is tails. A, B are independent.

Independent Events

Definition

Given a probability space $(\Omega, \Pr)$ and two events A, B are **independent** if and only if $\Pr[A \cap B] = \Pr[A] \Pr[B]$. Otherwise they are *dependent*. In other words A, B independent implies one does not affect the other.

Example

Two coins. $\Omega = \{HH, TT, HT, TH\}$ and $\Pr[HH] = \Pr[TT] = \Pr[HT] = \Pr[TH] = 1/4$.

1. A is the event that the first coin is heads and B is the event that second coin is tails. A, B are independent.
2. A is the event that both are not tails and B is event that second coin is heads.

Independent Events

Definition

Given a probability space $(\Omega, \Pr)$ and two events A, B are **independent** if and only if $\Pr[A \cap B] = \Pr[A] \Pr[B]$. Otherwise they are *dependent*. In other words A, B independent implies one does not affect the other.

Example

Two coins. $\Omega = \{HH, TT, HT, TH\}$ and $\Pr[HH] = \Pr[TT] = \Pr[HT] = \Pr[TH] = 1/4$.

1. A is the event that the first coin is heads and B is the event that second coin is tails. A, B are independent.
2. A is the event that both are not tails and B is event that second coin is heads. A, B **are dependent**.

Union bound

The probability of the union of two events, is no bigger than the probability of the sum of their probabilities.

Lemma

For any two events $\mathcal{E}$ and $\mathcal{F}$, we have that $\mathbf{Pr}\left[\mathcal{E} \cup \mathcal{F}\right] \leq \mathbf{Pr}\left[\mathcal{E}\right] + \mathbf{Pr}\left[\mathcal{F}\right]$.

Proof.

Consider $\mathcal{E}$ and $\mathcal{F}$ to be a collection of elmentery events (which they are). We have

$$\mathbf{Pr}\left[\mathcal{E} \cup \mathcal{F}\right] = \sum_{x \in \mathcal{E} \cup \mathcal{F}} \mathbf{Pr}[x]$$

$$\leq \sum_{x \in \mathcal{E}} \mathbf{Pr}[x] + \sum_{x \in \mathcal{F}} \mathbf{Pr}[x] = \mathbf{Pr}\left[\mathcal{E}\right] + \mathbf{Pr}\left[\mathcal{F}\right].$$

Random Variables

Definition

Given a probability space $(\Omega, \Pr)$ a (real-valued) random variable $\mathbf{X}$ over Ω is a function that maps each elementary event to a real number. In other words $\mathbf{X} : \Omega \to \mathbb{R}$.

Random Variables

Definition

Given a probability space $(\Omega, \Pr)$ a (real-valued) random variable X over Ω is a function that maps each elementary event to a real number. In other words $X : \Omega \to \mathbb{R}$.

Definition

Expectation For a random variable X over a probability space $(\Omega, \Pr)$ the **expectation** of X is defined as $\sum_{\omega \in \Omega} \Pr[\omega] \, X(\omega)$. In other words, the expectation is the average value of X according to the probabilities given by $\Pr[\cdot]$.

Expectation

Example

A 6-sided unbiased die. $\Omega = \{1,2,3,4,5,6\}$ and $\Pr[i] = 1/6$ for $1 \le i \le 6$.

1. $X : \Omega \rightarrow \mathbb{R}$ where $X(i) = i \mod 2$. Then $E[X] = \sum_{i=1}^{6} \Pr[i] \cdot X(i) = \frac{1}{6}\sum_{i=1}^{6} X(i) = 1/2$.

Expectation

Example

A 6-sided unbiased die. $\Omega = \{1, 2, 3, 4, 5, 6\}$ and $\Pr[i] = 1/6$ for $1 \leq i \leq 6$.

1. $X : \Omega \to \mathbb{R}$ where $X(i) = i \mod 2$. Then $E[X] = \sum_{i=1}^{6} \Pr[i] \cdot X(i) = \frac{1}{6} \sum_{i=1}^{6} X(i) = 1/2$.
2. $Y : \Omega \to \mathbb{R}$ where $Y(i) = i^2$. Then $E[Y] = \sum_{i=1}^{6} \frac{1}{6} \cdot i^2 = 91/6$.

Expected number of vertices?

Let $G = (V, E)$ be a graph with $\mathbf{n}$ vertices and $\mathbf{m}$ edges. Let H be the graph resulting from independently deleting every vertex of G with probability $\mathbf{1/2}$. Compute the expected number of vertices in H.

(A) $n/2$.

(B) $n/4$.

(C) $m/2$.

(D) $m/4$.

(E) none of the above.

Expected number of vertices is:

Probability Space

- $\Omega = \{0,1\}^n$. For $\omega \in \{0,1\}^n$, $\omega_v = 1$ if vertex v is present in H, else is zero.
- For each $\omega \in \Omega$, $\Pr[\omega] = \frac{1}{2^n}$.

Expected number of vertices is:

Probability Space

- $\Omega = \{0,1\}^n$. For $\omega \in \{0,1\}^n$, $\omega_v = 1$ if vertex v is present in H, else is zero.
- For each $\omega \in \Omega$, $\Pr[\omega] = \frac{1}{2^n}$.
- $X(\omega) = \#$ vertices in H as per $\omega = \#$ 1s in ω.

Expected number of vertices is:

Probability Space

- $\Omega = \{0,1\}^n$. For $\omega \in \{0,1\}^n$, $\omega_v = 1$ if vertex v is present in H, else is zero.
- For each $\omega \in \Omega$, $\Pr[\omega] = \frac{1}{2^n}$.
- $X(\omega) = \#$ vertices in H as per $\omega = \#$ 1s in ω.

$$
\begin{aligned}
E[X] &= \sum_{\omega\in\Omega} \Pr[\omega]\, X(\omega) \\
&= \sum_{\omega\in\Omega} {}^1\!/_{2^n} X(\omega) \\
&= {}^1\!/_{2^n} \sum_{k=0}^{n} \binom{n}{k} k \\
&= {}^1\!/_{2^n} (2^n \tfrac{n}{2}) \\
&= n/2
\end{aligned}
$$

Expected number of edges?

Let $G = (V, E)$ be a graph with $\mathbf{n}$ vertices and $\mathbf{m}$ edges. Let H be the graph resulting from independently deleting every vertex of G with probability $\mathbf{1/2}$. The expected number of edges in H is

- **(A)** $n/2$.
- **(B)** $n/4$.
- **(C)** $m/2$.
- **(D)** $m/4$.
- **(E)** none of the above.

Expected number of edges is:

Probability Space

- $\Omega = \{0,1\}^n$. For $\omega \in \{0,1\}^n$, $\omega_v = 1$ if vertex v is present in H, else is zero.
- For each $\omega \in \Omega$, $\Pr[\omega] = \frac{1}{2^n}$.

Expected number of edges is:

Probability Space

- $\Omega = \{0,1\}^n$. For $\omega \in \{0,1\}^n$, $\omega_v = 1$ if vertex v is present in H, else is zero.
- For each $\omega \in \Omega$, $\Pr[\omega] = \frac{1}{2^n}$.
- $X(\omega) = \#$ edges present in H as per ω = **??**

Expected number of edges is:

Probability Space

- $\Omega = \{0,1\}^n$. For $\omega \in \{0,1\}^n$, $\omega_v = 1$ if vertex v is present in H, else is zero.
- For each $\omega \in \Omega$, $\Pr[\omega] = \frac{1}{2^n}$.
- $X(\omega) = \#$ edges present in H as per ω = **??**

How to compute $E[X]$?

Indicator Random Variables

Definition

A **binary random variable** is one that takes on values in $\{0, 1\}$.

Indicator Random Variables

Definition

A **binary random variable** is one that takes on values in $\{0, 1\}$.

Special type of random variables that are quite useful.

Definition

Given a probability space $(\Omega, \Pr)$ and an event $A \subseteq \Omega$ the indicator random variable X_A is a binary random variable where $X_A(\omega) = 1$ if $\omega \in A$ and $X_A(\omega) = 0$ if $\omega \notin A$.

Indicator Random Variables

Definition

A **binary random variable** is one that takes on values in $\{0, 1\}$.

Special type of random variables that are quite useful.

Definition

Given a probability space $(\Omega, \Pr)$ and an event $A \subseteq \Omega$ the indicator random variable X_A is a binary random variable where $X_A(\omega) = 1$ if $\omega \in A$ and $X_A(\omega) = 0$ if $\omega \notin A$.

Example

A 6-sided unbiased die. $\Omega = \{1, 2, 3, 4, 5, 6\}$ and $\Pr[i] = 1/6$ for $1 \leq i \leq 6$. Let A be the even that i is divisible by 3. Then $X_A(i) = 1$ if $i = 3, 6$ and 0 otherwise.

Expectation

Proposition

For an indicator variable $\mathbf{X_A}$, $\mathbf{E[X_A] = Pr[A]}$.

Proof.

$$\begin{aligned}
\mathbf{E[X_A]} &= \sum_{y \in \Omega} \mathbf{X_A(y)\, Pr[y]} \\
&= \sum_{y \in A} 1 \cdot \mathbf{Pr[y]} + \sum_{y \in \Omega \setminus A} 0 \cdot \mathbf{Pr[y]} \\
&= \sum_{y \in A} \mathbf{Pr[y]} \\
&= \mathbf{Pr[A]}\,.
\end{aligned}$$

Linearity of Expectation

Lemma

Let $\mathbf{X}, \mathbf{Y}$ *be two random variables (not necessarily independent) over a probability space* $(\Omega, \Pr)$*. Then* $\mathbf{E}[\mathbf{X} + \mathbf{Y}] = \mathbf{E}[\mathbf{X}] + \mathbf{E}[\mathbf{Y}]$*.*

Proof.

$$\begin{aligned}\mathbf{E}[\mathbf{X} + \mathbf{Y}] &= \sum_{\omega \in \Omega} \Pr[\omega] \left(\mathbf{X}(\omega) + \mathbf{Y}(\omega)\right) \\ &= \sum_{\omega \in \Omega} \Pr[\omega]\, \mathbf{X}(\omega) + \sum_{\omega \in \Omega} \Pr[\omega]\, \mathbf{Y}(\omega) = \mathbf{E}[\mathbf{X}] + \mathbf{E}[\mathbf{Y}]\,.\end{aligned}$$

□

Linearity of Expectation

Lemma

Let $\mathbf{X}, \mathbf{Y}$ *be two random variables (not necessarily independent) over a probability space* $(\Omega, \Pr)$. *Then* $\mathbf{E}[\mathbf{X} + \mathbf{Y}] = \mathbf{E}[\mathbf{X}] + \mathbf{E}[\mathbf{Y}]$.

Proof.

$$\begin{aligned}
\mathbf{E}[\mathbf{X} + \mathbf{Y}] &= \sum_{\omega \in \Omega} \Pr[\omega] \left(\mathbf{X}(\omega) + \mathbf{Y}(\omega)\right) \\
&= \sum_{\omega \in \Omega} \Pr[\omega] \, \mathbf{X}(\omega) + \sum_{\omega \in \Omega} \Pr[\omega] \, \mathbf{Y}(\omega) = \mathbf{E}[\mathbf{X}] + \mathbf{E}[\mathbf{Y}].
\end{aligned}$$

□

Corollary

$\mathbf{E}[a_1 \mathbf{X}_1 + a_2 \mathbf{X}_2 + \ldots + a_n \mathbf{X}_n] = \sum_{i=1}^{n} a_i \, \mathbf{E}[\mathbf{X}_i]$.

Expected number of edges?

Let $G = (V, E)$ be a graph with $\mathbf{n}$ vertices and $\mathbf{m}$ edges. Let H be the graph resulting from independently deleting every vertex of G with probability $\mathbf{1/2}$. The expected number of edges in H is

Expected number of edges?

Let $G = (V, E)$ be a graph with n vertices and m edges. Let H be the graph resulting from independently deleting every vertex of G with probability $1/2$. The expected number of edges in H is

- Event $A_e =$ edge $e \in E$ is present in H.
- $\Pr\left[A_{e=(u,v)}\right] = \Pr[u \text{ and } v \text{ both are present}] = \Pr[u \text{ is present}] \cdot \Pr[v \text{ is present}] = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.

Expected number of edges?

Let $G = (V, E)$ be a graph with n vertices and m edges. Let H be the graph resulting from independently deleting every vertex of G with probability $1/2$. The expected number of edges in H is

- Event $A_e =$ edge $e \in E$ is present in H.
- $\Pr\left[A_{e=(u,v)}\right] = \Pr[u \text{ and } v \text{ both are present}] = \Pr[u \text{ is present}] \cdot \Pr[v \text{ is present}] = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.
- X_{A_e} indicator random variables, then $E[X_{A_e}] = \Pr[A_e]$.

Expected number of edges?

Let $G = (V, E)$ be a graph with $\mathbf{n}$ vertices and $\mathbf{m}$ edges. Let H be the graph resulting from independently deleting every vertex of G with probability $1/2$. The expected number of edges in H is

- Event $\mathbf{A_e}$ = edge $\mathbf{e} \in E$ is present in H.
- $\mathbf{Pr}\left[\mathbf{A_{e=(u,v)}}\right] = \mathbf{Pr}[\mathbf{u}$ and $\mathbf{v}$ both are present$] =$ $\mathbf{Pr}[\mathbf{u}$ is present$] \cdot \mathbf{Pr}[\mathbf{v}$ is present$] = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.
- $\mathbf{X_{A_e}}$ indicator random variables, then $\mathbf{E}[\mathbf{X_{A_e}}] = \mathbf{Pr}[\mathbf{A_e}]$.
- Let $\mathbf{X} = \sum_{e \in E} \mathbf{X_{A_e}}$ (Number of edges in H)

Expected number of edges?

Let $G = (V, E)$ be a graph with $\mathbf{n}$ vertices and $\mathbf{m}$ edges. Let H be the graph resulting from independently deleting every vertex of G with probability $\mathbf{1/2}$. The expected number of edges in H is

- Event $\mathbf{A_e}$ = edge $\mathbf{e} \in E$ is present in H.
- $\mathbf{Pr}\left[\mathbf{A_{e=(u,v)}}\right] = \mathbf{Pr}[\mathbf{u}$ and $\mathbf{v}$ both are present$] = \mathbf{Pr}[\mathbf{u}$ is present$] \cdot \mathbf{Pr}[\mathbf{v}$ is present$] = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.
- $\mathbf{X_{A_e}}$ indicator random variables, then $\mathbf{E}[\mathbf{X_{A_e}}] = \mathbf{Pr}[\mathbf{A_e}]$.
- Let $\mathbf{X} = \sum_{e \in E} \mathbf{X_{A_e}}$ (Number of edges in H)

$$\mathbf{E}[\mathbf{X}] = \mathbf{E}\left[\sum_{e \in E} \mathbf{X_{A_e}}\right] = \sum_{e \in E} \mathbf{E}[\mathbf{X_{A_e}}] = \sum_{e \in E} \mathbf{Pr}[\mathbf{A_e}] = \frac{\mathbf{m}}{\mathbf{4}}$$

Expected number of edges?

Let $G = (V, E)$ be a graph with $\mathbf{n}$ vertices and $\mathbf{m}$ edges. Let H be the graph resulting from independently deleting every vertex of G with probability $1/2$. The expected number of edges in H is

- Event $\mathbf{A_e}$ = edge $\mathbf{e} \in E$ is present in H.
- $\mathbf{Pr}\left[\mathbf{A_{e=(u,v)}}\right] = \mathbf{Pr}[\mathbf{u}$ and $\mathbf{v}$ both are present$] =$ $\mathbf{Pr}[\mathbf{u}$ is present$] \cdot \mathbf{Pr}[\mathbf{v}$ is present$] = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.
- $\mathbf{X_{A_e}}$ indicator random variables, then $\mathbf{E}[\mathbf{X_{A_e}}] = \mathbf{Pr}[\mathbf{A_e}]$.
- Let $\mathbf{X} = \sum_{e \in E} \mathbf{X_{A_e}}$ (Number of edges in H)

$$\mathbf{E}[\mathbf{X}] = \mathbf{E}\left[\sum_{e\in E} \mathbf{X_{A_e}}\right] = \sum_{e \in E} \mathbf{E}[\mathbf{X_{A_e}}] = \sum_{e\in E} \mathbf{Pr}[\mathbf{A_e}] = \frac{\mathbf{m}}{4}$$

It is important to setup random variables carefully.

Expected number of triangles?

Let $G = (V, E)$ be a graph with $\mathbf{n}$ vertices and $\mathbf{m}$ edges. Assume G has $\mathbf{t}$ triangles (i.e., a triangle is a simple cycle with three vertices). Let H be the graph resulting from deleting independently each vertex of G with probability $\mathbf{1/2}$. The expected number of triangles in H is

- **(A)** $t/2$.
- **(B)** $t/4$.
- **(C)** $t/8$.
- **(D)** $t/16$.
- **(E)** none of the above.

Independent Random Variables

Definition

Random variables $\mathbf{X}, \mathbf{Y}$ are said to be independent if

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}, \quad \mathbf{Pr}[\mathbf{X} = \mathbf{x} \wedge \mathbf{Y} = \mathbf{y}] = \mathbf{Pr}[\mathbf{X} = \mathbf{x}]\,\mathbf{Pr}[\mathbf{Y} = \mathbf{y}]$$

.

Independent Random Variables

Definition

Random variables $\mathbf{X}, \mathbf{Y}$ are said to be independent if

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}, \quad \mathbf{Pr}[\mathbf{X} = \mathbf{x} \wedge \mathbf{Y} = \mathbf{y}] = \mathbf{Pr}[\mathbf{X} = \mathbf{x}]\,\mathbf{Pr}[\mathbf{Y} = \mathbf{y}]$$

.

Examples

Two independent un-biased coin flips: $\mathbf{\Omega} = \{\mathbf{HH}, \mathbf{HT}, \mathbf{TH}, \mathbf{TT}\}$.

- $\mathbf{X} = \mathbf{1}$ if first coin is $\mathbf{H}$ else $\mathbf{0}$. $\mathbf{Y} = \mathbf{1}$ if second coin is $\mathbf{H}$ else $\mathbf{0}$.

Independent Random Variables

Definition

Random variables $\mathbf{X}, \mathbf{Y}$ are said to be independent if

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}, \quad \mathbf{Pr}[\mathbf{X} = \mathbf{x} \wedge \mathbf{Y} = \mathbf{y}] = \mathbf{Pr}[\mathbf{X} = \mathbf{x}] \, \mathbf{Pr}[\mathbf{Y} = \mathbf{y}]$$

.

Examples

Two independent un-biased coin flips: $\Omega = \{\mathbf{HH}, \mathbf{HT}, \mathbf{TH}, \mathbf{TT}\}$.

- $\mathbf{X} = \mathbf{1}$ if first coin is $\mathbf{H}$ else $\mathbf{0}$. $\mathbf{Y} = \mathbf{1}$ if second coin is $\mathbf{H}$ else $\mathbf{0}$. Independent.

Independent Random Variables

Definition

Random variables $\mathbf{X}, \mathbf{Y}$ are said to be independent if

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}, \quad \mathbf{Pr}[\mathbf{X} = \mathbf{x} \wedge \mathbf{Y} = \mathbf{y}] = \mathbf{Pr}[\mathbf{X} = \mathbf{x}] \, \mathbf{Pr}[\mathbf{Y} = \mathbf{y}]$$

.

Examples

Two independent un-biased coin flips: $\Omega = \{\mathbf{HH}, \mathbf{HT}, \mathbf{TH}, \mathbf{TT}\}$.

- $\mathbf{X} = \mathbf{1}$ if first coin is $\mathbf{H}$ else $\mathbf{0}$. $\mathbf{Y} = \mathbf{1}$ if second coin is $\mathbf{H}$ else $\mathbf{0}$. Independent.
- $\mathbf{X} = \#\mathbf{H}$, $\mathbf{Y} = \#\mathbf{T}$.

Independent Random Variables

Definition

Random variables $\mathbf{X}, \mathbf{Y}$ are said to be independent if

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}, \quad \mathbf{Pr}[\mathbf{X} = \mathbf{x} \wedge \mathbf{Y} = \mathbf{y}] = \mathbf{Pr}[\mathbf{X} = \mathbf{x}]\,\mathbf{Pr}[\mathbf{Y} = \mathbf{y}]$$

.

Examples

Two independent un-biased coin flips: $\Omega = \{\mathbf{HH}, \mathbf{HT}, \mathbf{TH}, \mathbf{TT}\}$.

- $\mathbf{X} = \mathbf{1}$ if first coin is $\mathbf{H}$ else $\mathbf{0}$. $\mathbf{Y} = \mathbf{1}$ if second coin is $\mathbf{H}$ else $\mathbf{0}$. Independent.
- $\mathbf{X} = \#\mathbf{H}$, $\mathbf{Y} = \#\mathbf{T}$. Dependent. Why?

independent Randomized Variables

Lemma

If $\mathbf{X}$ *and* $\mathbf{Y}$ *are independent then* $\mathbf{E}[\mathbf{X} \cdot \mathbf{Y}] = \mathbf{E}[\mathbf{X}] \cdot \mathbf{E}[\mathbf{Y}]$

Proof.

$$
\begin{aligned}
\mathbf{E}[\mathbf{X} \cdot \mathbf{Y}] &= \sum_{\omega \in \Omega} \Pr[\omega] \left(\mathbf{X}(\omega) \cdot \mathbf{Y}(\omega)\right) \\
&= \sum_{x,y \in \mathbb{R}} \Pr[\mathbf{X} = x \wedge \mathbf{Y} = y] \left(x \cdot y\right) \\
&= \sum_{x,y \in \mathbb{R}} \Pr[\mathbf{X} = x] \cdot \Pr[\mathbf{Y} = y] \cdot x \cdot y \\
&= \Big(\sum_{x \in \mathbb{R}} \Pr[\mathbf{X} = x]\, x\Big)\Big(\sum_{y \in \mathbb{R}} \Pr[\mathbf{Y} = y]\, y\Big) = \mathbf{E}[\mathbf{X}]\, \mathbf{E}[\mathbf{Y}]
\end{aligned}
$$

$\square$

Types of Randomized Algorithms

Typically one encounters the following types:

1. **Las Vegas randomized algorithms:** for a given input $\mathbf{x}$ output of *algorithm is always correct* but the *running time is a random variable*. In this case we are interested in analyzing the *expected* running time.

Types of Randomized Algorithms

Typically one encounters the following types:

1. **Las Vegas randomized algorithms:** for a given input **x** output of *algorithm is always correct* but the *running time is a random variable*. In this case we are interested in analyzing the *expected* running time.
2. **Monte Carlo randomized algorithms:** for a given input **x** the *running time is deterministic* but the *output is random*; correct with some probability. In this case we are interested in analyzing the *probability* of the correct output (and also the running time).
3. Algorithms whose running time and output may both be random.

Analyzing Las Vegas Algorithms

Deterministic algorithm Q for a problem Π:

1. Let $Q(x)$ be the time for Q to run on input x of length $|x|$.
2. Worst-case analysis: run time on worst input for a given size n.

$$T_{wc}(n) = \max_{x:|x|=n} Q(x).$$

Analyzing Las Vegas Algorithms

Deterministic algorithm $\mathbf{Q}$ for a problem $\mathbf{\Pi}$:

1. Let $\mathbf{Q(x)}$ be the time for $\mathbf{Q}$ to run on input $\mathbf{x}$ of length $|\mathbf{x}|$.
2. Worst-case analysis: run time on worst input for a given size $\mathbf{n}$.

$$T_{wc}(n) = \max_{x:|x|=n} Q(x).$$

Randomized algorithm $\mathbf{R}$ for a problem $\mathbf{\Pi}$:

1. Let $\mathbf{R(x)}$ be the time for $\mathbf{Q}$ to run on input $\mathbf{x}$ of length $|\mathbf{x}|$.
2. $\mathbf{R(x)}$ is a random variable: depends on random bits used by $\mathbf{R}$.
3. $\mathbf{E[R(x)]}$ is the expected running time for $\mathbf{R}$ on $\mathbf{x}$
4. Worst-case analysis: expected time on worst input of size $\mathbf{n}$

$$T_{rand-wc}(n) = \max_{x:|x|=n} E[R(x)].$$

Analyzing Monte Carlo Algorithms

Randomized algorithm $\mathbf{M}$ for a problem $\mathbf{\Pi}$:

1. Let $\mathbf{M(x)}$ be the time for $\mathbf{M}$ to run on input $\mathbf{x}$ of length $|\mathbf{x}|$. For Monte Carlo, assumption is that run time is deterministic.
2. Let $\mathbf{Pr[x]}$ be the probability that $\mathbf{M}$ is correct on $\mathbf{x}$.
3. $\mathbf{Pr[x]}$ is a random variable: depends on random bits used by $\mathbf{M}$.
4. Worst-case analysis: success probability on worst input

$$\mathbf{P_{rand-wc}(n) = \min_{x:|x|=n} Pr[x]}.$$

Part III

Why does randomization help?

Ping and find.

Consider a deterministic algorithm $\mathbf{A}$ that is trying to find an element in an array $\mathbf{X}$ of size $\mathbf{n}$. At every step it is allowed to ask the value of one cell in the array, and the adversary is allowed after each such ping, to shuffle elements around in the array in any way it seems fit. For the best possible deterministic algorithm the number of rounds it has to play this game till it finds the required element is

(A) $\mathbf{O(1)}$

(B) $\mathbf{O(n)}$

(C) $\mathbf{O(n \log n)}$

(D) $\mathbf{O(n^2)}$

(E) ∞.

Ping and find randomized.

Consider an algorithm **randFind** that is trying to find an element in an array **X** of size **n**. At every step it asks the value of one random cell in the array, and the adversary is allowed after each such ping, to shuffle elements around in the array in any way it seems fit. This algorithm would stop in expectation after

- **(A)** $\mathbf{O(1)}$
- **(B)** $\mathbf{O(\log n)}$
- **(C)** $\mathbf{O(n)}$
- **(D)** $\mathbf{O(n^2)}$
- **(E)** ∞.

steps.

Abundance of witnesses

Consider the problem of finding an "approximate median" of an unsorted array $\mathbf{A[1..n]}$: an element of $\mathbf{A}$ with rank between $\mathbf{n/4}$ and $\mathbf{3n/4}$.

- Finding an approximate median is not any easier than a proper median.
- $\mathbf{n/2}$ elements of $\mathbf{A}$ qualify as approximate medians and hence a random element is good with probability $\mathbf{1/2}$!

Part IV

Randomized Quick Sort

Randomized **QuickSort**

Randomized **QuickSort**

1. Pick a pivot element *uniformly at random* from the array.
2. Split array into 3 subarrays: those smaller than pivot, those larger than pivot, and the pivot itself.
3. Recursively sort the subarrays, and concatenate them.

Analysis

What events to count?

- Number of Comparisions.

Analysis

What events to count?

- Number of Comparisions.

What is the probability space?

- All the coin tosses at all levels and parts of recursion.

Analysis

What events to count?

- Number of Comparisions.

What is the probability space?

- All the coin tosses at all levels and parts of recursion.

Too Big!!

Analysis

What events to count?

- Number of Comparisions.

What is the probability space?

- All the coin tosses at all levels and parts of recursion.

Too Big!!

What random variables to define?
What are the events of the algorithm?

Analysis via Recurrence

1. Given array $\mathbf{A}$ of size $\mathbf{n}$, let $\mathbf{Q(A)}$ be number of comparisons of randomized **QuickSort** on $\mathbf{A}$.
2. Note that $\mathbf{Q(A)}$ is a random variable.
3. Let $\mathbf{A}^{\mathbf{i}}_{\text{left}}$ and $\mathbf{A}^{\mathbf{i}}_{\text{right}}$ be the left and right arrays obtained if:

Let $\mathbf{X_i}$ be indicator random variable, which is set to $\mathbf{1}$ if pivot is of rank $\mathbf{i}$ in $\mathbf{A}$, else zero.

$$\mathbf{Q(A)} = \mathbf{n} + \sum_{\mathbf{i}=1}^{\mathbf{n}} \mathbf{X_i} \cdot \left(\mathbf{Q}(\mathbf{A}^{\mathbf{i}}_{\text{left}}) + \mathbf{Q}(\mathbf{A}^{\mathbf{i}}_{\text{right}}) \right).$$

Analysis via Recurrence

1. Given array $\mathbf{A}$ of size $\mathbf{n}$, let $\mathbf{Q(A)}$ be number of comparisons of randomized **QuickSort** on $\mathbf{A}$.
2. Note that $\mathbf{Q(A)}$ is a random variable.
3. Let $\mathbf{A}^{\mathbf{i}}_{\text{left}}$ and $\mathbf{A}^{\mathbf{i}}_{\text{right}}$ be the left and right arrays obtained if:

Let $\mathbf{X_i}$ be indicator random variable, which is set to $\mathbf{1}$ if pivot is of rank $\mathbf{i}$ in $\mathbf{A}$, else zero.

$$\mathbf{Q(A)} = \mathbf{n} + \sum_{\mathbf{i}=1}^{\mathbf{n}} \mathbf{X_i} \cdot \left(\mathbf{Q}(\mathbf{A}^{\mathbf{i}}_{\text{left}}) + \mathbf{Q}(\mathbf{A}^{\mathbf{i}}_{\text{right}}) \right).$$

Since each element of $\mathbf{A}$ has probability exactly of $\mathbf{1/n}$ of being chosen:

$$\mathbf{E}[\mathbf{X_i}] = \mathbf{Pr}[\text{pivot has rank } \mathbf{i}] = \mathbf{1/n}.$$

Independence of Random Variables

Lemma

Random variables $\mathbf{X_i}$ *is independent of random variables* $\mathbf{Q(A^i_{\mathit{left}})}$ *as well as* $\mathbf{Q(A^i_{\mathit{right}})}$, i.e.

$$\mathbf{E}\left[\mathbf{X_i} \cdot \mathbf{Q(A^i_{\mathit{left}})}\right] = \mathbf{E}[\mathbf{X_i}]\,\mathbf{E}\left[\mathbf{Q(A^i_{\mathit{left}})}\right]$$
$$\mathbf{E}\left[\mathbf{X_i} \cdot \mathbf{Q(A^i_{\mathit{right}})}\right] = \mathbf{E}[\mathbf{X_i}]\,\mathbf{E}\left[\mathbf{Q(A^i_{\mathit{right}})}\right]$$

Proof.

This is because the algorithm, while recursing on $\mathbf{Q(A^i_{left})}$ and $\mathbf{Q(A^i_{right})}$ uses new random coin tosses that are independent of the coin tosses used to decide the first pivot. Only the latter decides value of $\mathbf{X_i}$. □

Analysis via Recurrence

Let $\mathbf{T(n) = max_{A:|A|=n} E[Q(A)]}$ be the worst-case expected running time of randomized **QuickSort** on arrays of size $\mathbf{n}$.

Analysis via Recurrence

Let $\mathbf{T(n) = \max_{A:|A|=n} E[Q(A)]}$ be the worst-case expected running time of randomized **QuickSort** on arrays of size $\mathbf{n}$.

We have, for any $\mathbf{A}$:

$$\mathbf{Q(A) = n + \sum_{i=1}^{n} X_i \left(Q(A^i_{\text{left}}) + Q(A^i_{\text{right}}) \right)}$$

Analysis via Recurrence

Let $\mathbf{T(n) = \max_{A:|A|=n} E[Q(A)]}$ be the worst-case expected running time of randomized **QuickSort** on arrays of size $\mathbf{n}$.

We have, for any $\mathbf{A}$:

$$\mathbf{Q(A) = n + \sum_{i=1}^{n} X_i \left(Q(A^i_{\text{left}}) + Q(A^i_{\text{right}}) \right)}$$

By linearity of expectation, and independence random variables:

$$\mathbf{E\Big[Q(A)\Big] = n + \sum_{i=1}^{n} E[X_i] \Big(E\Big[Q(A^i_{\text{left}})\Big] + E\Big[Q(A^i_{\text{right}})\Big] \Big)}.$$

Analysis via Recurrence

Let $\mathbf{T(n) = \max_{A:|A|=n} E[Q(A)]}$ be the worst-case expected running time of randomized **QuickSort** on arrays of size $\mathbf{n}$.

We have, for any $\mathbf{A}$:

$$\mathbf{Q(A) = n + \sum_{i=1}^{n} X_i \left(Q(A^i_{\text{left}}) + Q(A^i_{\text{right}}) \right)}$$

By linearity of expectation, and independence random variables:

$$\mathbf{E\Big[Q(A)\Big] = n + \sum_{i=1}^{n} E[X_i]\Big(E\Big[Q(A^i_{\text{left}})\Big] + E\Big[Q(A^i_{\text{right}})\Big] \Big)}.$$

$$\Rightarrow \quad \mathbf{E\Big[Q(A)\Big] \leq n + \sum_{i=1}^{n} \frac{1}{n} \left(T(i-1) + T(n-i) \right)}.$$

Analysis via Recurrence

Let $\mathbf{T(n) = \max_{A:|A|=n} E[Q(A)]}$ be the worst-case expected running time of randomized **QuickSort** on arrays of size $\mathbf{n}$.

Analysis via Recurrence

Let $\mathbf{T(n) = \max_{A:|A|=n} E[Q(A)]}$ be the worst-case expected running time of randomized **QuickSort** on arrays of size $\mathbf{n}$.
We derived:

$$\mathbf{E}\Big[\mathbf{Q(A)}\Big] \leq \mathbf{n} + \sum_{\mathbf{i=1}}^{\mathbf{n}} \frac{\mathbf{1}}{\mathbf{n}} \left(\mathbf{T(i-1) + T(n-i)}\right).$$

Note that above holds for any $\mathbf{A}$ of size $\mathbf{n}$. Therefore

$$\max_{\mathbf{A:|A|=n}} \mathbf{E[Q(A)] = T(n)} \leq \mathbf{n} + \sum_{\mathbf{i=1}}^{\mathbf{n}} \frac{\mathbf{1}}{\mathbf{n}} \left(\mathbf{T(i-1) + T(n-i)}\right).$$

Solving the Recurrence

$$\mathbf{T(n) \leq n + \sum_{i=1}^{n} \frac{1}{n}\left(T(i-1) + T(n-i)\right)}$$

with base case $\mathbf{T(1) = 0}$.

Solving the Recurrence

$$T(n) \leq n + \sum_{i=1}^{n} \frac{1}{n}\left(T(i-1) + T(n-i)\right)$$

with base case $T(1) = 0$.

Lemma

$T(n) = O(n \log n)$.

Solving the Recurrence

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

with base case $\mathbf{T(1) = 0}$.

Lemma

$\mathbf{T(n) = O(n \log n)}$.

Proof.

(Guess and) Verify by induction. □

Part V

Slick analysis of QuickSort

A Slick Analysis of **QuickSort**

Let $\mathbf{Q(A)}$ be number of comparisons done on input array $\mathbf{A}$:

1. For $1 \leq \mathbf{i} < \mathbf{j} < \mathbf{n}$ let $\mathbf{R_{ij}}$ be the event that rank $\mathbf{i}$ element is compared with rank $\mathbf{j}$ element.
2. $\mathbf{X_{ij}}$ is the indicator random variable for $\mathbf{R_{ij}}$. That is, $\mathbf{X_{ij}} = 1$ if rank $\mathbf{i}$ is compared with rank $\mathbf{j}$ element, otherwise $\mathbf{0}$.

A Slick Analysis of **QuickSort**

Let $\mathbf{Q(A)}$ be number of comparisons done on input array $\mathbf{A}$:

1. For $1 \leq i < j < n$ let R_{ij} be the event that rank i element is compared with rank j element.
2. X_{ij} is the indicator random variable for R_{ij}. That is, $X_{ij} = 1$ if rank i is compared with rank j element, otherwise 0.

$$Q(A) = \sum_{1 \leq i < j \leq n} X_{ij}$$

and hence by linearity of expectation,

$$E\left[Q(A)\right] = \sum_{1 \leq i < j \leq n} E\left[X_{ij}\right] = \sum_{1 \leq i < j \leq n} \Pr\left[R_{ij}\right].$$

A Slick Analysis of **QuickSort**

$\boxed{\mathbf{R_{ij}} = \text{rank } \mathbf{i} \text{ element is compared with rank } \mathbf{j} \text{ element.}}$

Question: What is $\mathbf{Pr[R_{ij}]}$?

A Slick Analysis of **QuickSort**

$\boxed{\mathbf{R_{ij}} = \text{rank } \mathbf{i} \text{ element is compared with rank } \mathbf{j} \text{ element.}}$

Question: What is $\mathbf{Pr[R_{ij}]}$?

	7	5	9	1	3	4	8	6
With ranks:	6	4	8	1	2	3	7	5

A Slick Analysis of **QuickSort**

$\boxed{\mathbf{R_{ij}} = \text{rank } \mathbf{i} \text{ element is compared with rank } \mathbf{j} \text{ element.}}$

Question: What is $\mathbf{Pr[R_{ij}]}$?

	7	5	9	1	3	4	8	6
With ranks:	6	4	8	1	2	3	7	5

As such, probability of comparing **5** to **8** is $\mathbf{Pr[R_{4,7}]}$.

A Slick Analysis of **QuickSort**

$\boxed{\mathbf{R_{ij}} = \text{rank } \mathbf{i} \text{ element is compared with rank } \mathbf{j} \text{ element.}}$

Question: What is $\mathbf{Pr[R_{ij}]}$?

7	5	9	1	3	4	8	6
6	4	8	1	2	3	7	5

With ranks: 6 4 8 1 2 3 7 5

1. If pivot too small (say **3** [rank 2]). Partition and call recursively:

   $\boxed{7\ 5\ 9\ 1\ 3\ 4\ 8\ 6} \implies \boxed{1\quad 3\quad 7\ 5\ 9\ 4\ 8\ 6}$

   Decision if to compare **5** to **8** is moved to subproblem.

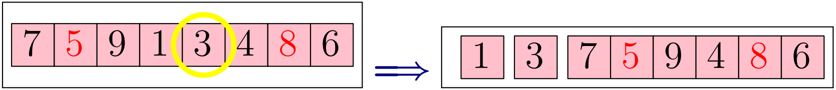
A Slick Analysis of **QuickSort**

$\mathbf{R_{ij}}$ = rank **i** element is compared with rank **j** element.

Question: What is $\mathbf{Pr[R_{ij}]}$?

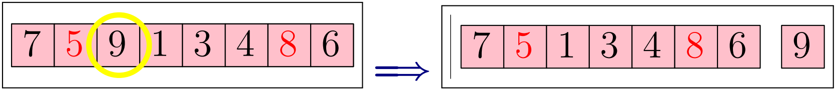
	7	5	9	1	3	4	8	6
With ranks:	6	4	8	1	2	3	7	5

1. If pivot too small (say **3** [rank 2]). Partition and call recursively:



Decision if to compare **5** to **8** is moved to subproblem.

2. If pivot too large (say **9** [rank 8]):



Decision if to compare **5** to **8** moved to subproblem.

A Slick Analysis of **QuickSort**

Question: What is $\mathbf{Pr[R_{i,j}]}$?

7	5	9	1	3	4	8	6
6	4	8	1	2	3	7	5

As such, probability of comparing **5** to **8** is $\mathbf{Pr[R_{4,7}]}$.

1. If pivot is **5** (rank 4). Bingo!

7	5	9	1	3	4	8	6

$\Longrightarrow$

1	3	4	5	7	9	8	6

A Slick Analysis of **QuickSort**

Question: What is $\Pr[R_{i,j}]$?

7	5	9	1	3	4	8	6
6	4	8	1	2	3	7	5

As such, probability of comparing **5** to **8** is $\Pr[R_{4,7}]$.

1. If pivot is **5** (rank 4). Bingo!

7	5	9	1	3	4	8	6

$\Longrightarrow$

1	3	4		5		7	9	8	6

2. If pivot is **8** (rank 7). Bingo!

7	5	9	1	3	4	8	6

$\Longrightarrow$

7	5	1	3	4	6		8		9

A Slick Analysis of **QuickSort**

Question: What is $\mathbf{Pr}[R_{i,j}]$?

7	5	9	1	3	4	8	6
6	4	8	1	2	3	7	5

As such, probability of comparing **5** to **8** is $\mathbf{Pr}[R_{4,7}]$.

1. If pivot is **5** (rank 4). Bingo!

7	5	9	1	3	4	8	6

$\Longrightarrow$

1	3	4	5	7	9	8	6

2. If pivot is **8** (rank 7). Bingo!

7	5	9	1	3	4	8	6

$\Longrightarrow$

7	5	1	3	4	6	8	9

3. If pivot in between the two numbers (say **6** [rank 5]):

7	5	9	1	3	4	8	6

$\Longrightarrow$

5	1	3	4	6	7	8	9

5 and **8** will never be compared to each other.

A Slick Analysis of **QuickSort**

Question: What is $\mathbf{Pr}[\mathbf{R_{i,j}}]$?

Conclusion:

$\mathbf{R_{i,j}}$ happens if and only if:

> $\mathbf{i}$th or $\mathbf{j}$th ranked element is the first pivot out of $\mathbf{i}$th to $\mathbf{j}$th ranked elements.

How to analyze this?

Thinking acrobatics!

1. Assign every element in the array a random priority (say in $[0, 1]$).
2. Choose pivot to be the element with lowest priority in subproblem.
3. Equivalent to picking pivot uniformly at random (as **QuickSort** do).

A Slick Analysis of **QuickSort**

Question: What is $\mathbf{Pr}[\mathbf{R}_{i,j}]$?

How to analyze this?

Thinking acrobatics!

1. Assign every element in the array a random priority (say in $[0, 1]$).
2. Choose pivot to be the element with lowest priority in subproblem.

$\implies$ $\mathbf{R}_{i,j}$ happens if either $\mathbf{i}$ or $\mathbf{j}$ have lowest priority out of elements rank $\mathbf{i}$ to $\mathbf{j}$,

A Slick Analysis of **QuickSort**

Question: What is $\Pr[R_{i,j}]$?

How to analyze this?

Thinking acrobatics!

1. Assign every element in the array a random priority (say in $[0, 1]$).
2. Choose pivot to be the element with lowest priority in subproblem.

$\implies$ $R_{i,j}$ happens if either i or j have lowest priority out of elements rank i to j,
There are $k = j - i + 1$ relevant elements.

$$\Pr\left[R_{i,j}\right] = \frac{2}{k} = \frac{2}{j - i + 1}.$$

A Slick Analysis of **QuickSort**

Question: What is $\mathbf{Pr}[R_{ij}]$?

A Slick Analysis of **QuickSort**

Question: What is $\mathbf{Pr}[\mathbf{R_{ij}}]$?

Lemma

$\mathbf{Pr}\left[\mathbf{R_{ij}}\right] = \frac{2}{j-i+1}$.

A Slick Analysis of **QuickSort**

Question: What is $\mathbf{Pr[R_{ij}]}$?

Lemma

$\mathbf{Pr}\left[\mathbf{R_{ij}}\right] = \frac{2}{j-i+1}$.

Proof.

Let $a_1, \ldots, a_i, \ldots, a_j, \ldots, a_n$ be elements of $\mathbf{A}$ in sorted order. Let $S = \{a_i, a_{i+1}, \ldots, a_j\}$

Observation: If pivot is chosen outside $\mathbf{S}$ then all of $\mathbf{S}$ either in left array or right array.

Observation: a_i and a_j separated when a pivot is chosen from $\mathbf{S}$ for the first time. Once separated no comparison.

Observation: a_i is compared with a_j if and only if either a_i or a_j is chosen as a pivot from $\mathbf{S}$ at separation... $\square$

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\Pr\left[R_{ij}\right] = \frac{2}{j-i+1}$.

Proof.

Let $a_1, \ldots, a_i, \ldots, a_j, \ldots, a_n$ be sort of A. Let $S = \{a_i, a_{i+1}, \ldots, a_j\}$

Observation: a_i is compared with a_j if and only if either a_i or a_j is chosen as a pivot from S at separation.

Observation: Given that pivot is chosen from S the probability that it is a_i or a_j is exactly $2/|S| = 2/(j-i+1)$ since the pivot is chosen uniformly at random from the array. □

How much is this?

$\mathbf{H_n = \sum_{i=1}^{n} \frac{1}{i}}$ is the $\mathbf{n}$'th harmonic number

(A) $\mathbf{H_n = \Theta(1)}$.

(B) $\mathbf{H_n = \Theta(\log\log n)}$.

(C) $\mathbf{H_n = \Theta(\sqrt{\log n})}$.

(D) $\mathbf{H_n = \Theta(\log n)}$.

(E) $\mathbf{H_n = \Theta(\log^2 n)}$.

And how much is this?

$$\mathbf{T_n = \sum_{i=1}^{n-1} \sum_{j=1}^{n-i} \frac{1}{j}}$$

is equal to

(A) $\mathbf{T_n = \Theta(n)}$.

(B) $\mathbf{T_n = \Theta(n \log n)}$.

(C) $\mathbf{T_n = \Theta(n \log^2 n)}$.

(D) $\mathbf{T_n = \Theta(n^2)}$.

(E) $\mathbf{T_n = \Theta(n^3)}$.

A Slick Analysis of **QuickSort**

Continued...

$$\mathbf{E}\Big[\mathbf{Q}(\mathbf{A})\Big] = \sum_{1\leq i<j\leq n} \mathbf{E}[\mathbf{X}_{ij}] = \sum_{1\leq i<j\leq n} \mathbf{Pr}[\mathbf{R}_{ij}]\,.$$

Lemma

$\mathbf{Pr}[\mathbf{R}_{ij}] = \frac{2}{j-i+1}$.

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\mathbf{Pr[R_{ij}] = \frac{2}{j-i+1}}$.

$$\mathbf{E}\left[Q(A)\right] = \sum_{1 \le i < j \le n} \mathbf{Pr}\left[R_{ij}\right] = \sum_{1 \le i < j \le n} \frac{2}{j - i + 1}$$

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$E\left[Q(A)\right] = \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1}$$

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$\mathbf{E}\Big[Q(A)\Big] = \sum_{1 \le i < j \le n} \frac{2}{j-i+1}$$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$$

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$E\left[Q(A)\right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$$

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$E\left[Q(A)\right] = 2\sum_{i=1}^{n-1}\sum_{i<j}^{n}\frac{1}{j-i+1}$$

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$E\left[Q(A)\right] = 2\sum_{i=1}^{n-1}\sum_{i<j}^{n}\frac{1}{j-i+1}$$

A Slick Analysis of **QuickSort**

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$E\left[Q(A)\right] = 2\sum_{i=1}^{n-1}\sum_{i<j}^{n}\frac{1}{j-i+1} \leq 2\sum_{i=1}^{n-1}\quad\sum_{\Delta=2}^{n-i+1}\frac{1}{\Delta}$$

A Slick Analysis of QuickSort

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$E\left[Q(A)\right] = 2\sum_{i=1}^{n-1}\sum_{i<j}^{n}\frac{1}{j-i+1} \leq 2\sum_{i=1}^{n-1}\sum_{\Delta=2}^{n-i+1}\frac{1}{\Delta}$$

$$\leq 2\sum_{i=1}^{n-1}(H_{n-i+1}-1) \leq 2\sum_{1\leq i<n} H_n$$

A Slick Analysis of QuickSort

Continued...

Lemma

$\Pr[R_{ij}] = \frac{2}{j-i+1}$.

$$
\begin{aligned}
E\left[Q(A)\right] &= 2\sum_{i=1}^{n-1}\sum_{i<j}^{n}\frac{1}{j-i+1} \le 2\sum_{i=1}^{n-1}\sum_{\Delta=2}^{n-i+1}\frac{1}{\Delta} \\
&\le 2\sum_{i=1}^{n-1}(H_{n-i+1}-1) \le 2\sum_{1\le i<n} H_n \\
&\le 2nH_n = O(n\log n)
\end{aligned}
$$

Where do I get random bits?

Question: Are true random bits available in practice?

1. Buy them!
2. CPUs use physical phenomena to generate random bits.
3. Can use pseudo-random bits or semi-random bits from nature. Several fundamental unresolved questions in complexity theory on this topic. Beyond the scope of this course.
4. In practice pseudo-random generators work quite well in many applications.
5. The model is interesting to think in the abstract and is very useful even as a theoretical construct. One can *derandomize* randomized algorithms to obtain deterministic algorithms.