LECTURE 8 (September 19^{th})

More Probability & Randomized Algorithms

RECAP Equality Testing

Given two binary vectors $u, v \in \{0,1\}^n$
Decide if they are equal or not

Only operation that is allowed : DOTPRODUCT (a, b) $\rightarrow$ Time $B(n)$

take dot product (mod 2) of any two binary vectors $a, b \in \{0,1\}^n$
i.e. output is

$$\underbrace{\langle a,b\rangle}_{=\sum_{i=1}^{n} a_i b_i} \bmod 2 = \sum_{i=1}^{n} a_i b_i \pmod 2 = \begin{cases} 1 & \text{if } \langle a,b\rangle \text{ is odd} \\ 0 & \text{o/w} \end{cases}$$

Deterministically

Let $e_i = [0 0 \cdots \overset{i^{th}\text{ coordinate}}{1} 0 \cdots 0]$ be the i^{th}-standard basis vector
Invoking DOTPRODUCT (u, e_i) for $i = 1$ to n, tells us what u or v is
Time $= O(n \cdot B(n))$

Algorithm

- Pick a random vector $r \in \{0,1\}^n$
- If $\langle u, r\rangle = \langle v, r\rangle \bmod 2$, then output EQUAL
- Else output NOT EQUAL

$\boxed{\text{Theorem}}$ $\mathbb{P}[\text{Algorithm errs}] \le \frac{1}{2}$ and is running time is $\underbrace{O(n + B(n))}_{\text{obvious}}$

Proof Algorithm only errs if $u \neq v$

Suppose u and v differ on the last bit : $u_n \neq v_n$

Then, $$\langle u, r\rangle = \underbrace{\sum_{i=1}^{n-1} u_i r_i}_{\alpha} + u_n r_n$$

$$\langle v, r\rangle = \underbrace{\sum_{i=1}^{n-1} v_i r_i}_{\beta} + v_n r_n$$

Now, there are two cases

$\boxed{1}$ $\alpha \neq \beta \bmod 2$ w.p. $\frac{1}{2}$ $r_n = 0$, so $\langle u, r\rangle \neq \langle v, r\rangle$

$\boxed{2}$ $\alpha = \beta \bmod 2$ w.p. $\frac{1}{2}$ $r_n = 1$, so $\langle u, r\rangle \neq \langle v, r\rangle$

Thus, $\mathbb{P}[\text{Algorithm errs}] \le \frac{1}{2}$ ← This is not very small
Can we make it $\le \delta$?

Repetition/Amplification Trick Run the algorithm $t = \lceil \log \frac{1}{\delta} \rceil$ times independently

If any execution says NOT EQUAL $\Rightarrow$ output NOT EQUAL

o/w $\Rightarrow$ output EQUAL

Again, algorithm only errs if $u \neq v$,

$$\mathbb{P}[\text{Algorithm errs}] = \mathbb{P}[\text{all } i \text{ iteration return EQUAL}]$$
$$= \prod_{i=1}^{t} \frac{1}{2} = 2^{-t} = 2^{-\lceil \log 1/\delta \rceil} \leq \delta$$

Runtime is now $O(n + B(n) \cdot \log \frac{1}{\delta})$

Testing Matrix Product Given Boolean matrices $B, C, D \in \{0,1\}^{n \times n}$ decide if $BC = D \pmod 2$

Matrix Multiplication takes $O(n^{2.3\ldots})$ time.

Randomness allows us to do it in roughly $O(n^2)$ time.

Algorithm Take a random Boolean vector $r \in \{0,1\}^n$

- Compute (mod 2) $Dr = y$
- Compute (mod 2) $BCr = B(Cr) = x$

Matrix-vector multiplication Takes $O(n^2)$ time

- If $x \neq y$, return NOT EQUAL o/w return EQUAL

Error Analysis If $BC = D \pmod 2$ $\Rightarrow$ algorithm is always correct

If $BC \neq D \pmod 2$ $\Rightarrow$ algorithm may fail

What is the probability of failure ?

Assume i^{th} row of BC and D are not equal

Let $u = i^{th}$ row of BC, $v = i^{th}$ row of D. Then, $u \neq v$ by assumption

By previous lemma, $\mathbb{P}[\langle u,r\rangle \bmod 2 = \langle v,r\rangle \bmod 2] = \frac{1}{2}$

So, $\mathbb{P}[\text{fail}] \leq \frac{1}{2}$

We can make the error at most δ, by repeating $\log 1/\delta$ times

Random Variable

A random variable is a function $X : \Omega \longrightarrow V$
↳ value set

E.g. if $V = \mathbb{Z}$, X is a random integer
$V = \{0,1\}$, X is a random bit
$V =$ graph, X is a random graph

We write $\mathbb{P}[X = x]$ or $\mathbb{P}[X \le x]$ or $\mathbb{P}[X = Y]$ to denote events about random variables

Expectation For real / complex / vector valued random variable X

$$\mathbb{E}[X] = \sum_x x \, \mathbb{P}[X = x]$$

E.g. X = value of random dice $\mathbb{E}[X] = \frac{7}{2}$

Note Random variables over infinite sample spaces (e.g. integers) may not have finite expectations

Conditional Expectation Given an event A, the conditional expectation of X given A is

$$\mathbb{E}[X \mid A] = \sum_x x \cdot \mathbb{P}[X = x \mid A]$$

$$\mathbb{E}[X] = \mathbb{E}[X \mid A] \cdot \mathbb{P}[A] + \mathbb{E}[X \mid \neg A] \cdot \mathbb{P}[\neg A]$$

$$\mathbb{E}[X] = \sum_y \mathbb{E}[X \mid Y = y] \cdot \mathbb{P}[Y = y] = \mathbb{E}[\mathbb{E}[X \mid Y]]$$

Independence Two random variables X and Y are independent if for all x, y: the events $X = x$ and $Y = y$ are independent

If X and Y are independent, then $\mathbb{E}[XY] = \mathbb{E}[X] \cdot \mathbb{E}[Y]$

Similarly, if $X_1 \dots X_n$ are fully independent, then

$$\mathbb{E}\left[\prod_{i=1}^{n} X_i\right] = \prod_{i=1}^{n} \mathbb{E}[X_i]$$

Linearity For any random variables $X_1 \dots X_n$ (may be dependent) & reals $\alpha_1 \dots \alpha_n$

$$\mathbb{E}\left[\sum_{i=1}^{n} (\alpha_i X_i)\right] = \sum_{i=1}^{n} \alpha_i \cdot \mathbb{E}[X_i]$$

Example Toss independent coins where each coin comes up heads w.p. $p \in [0,1]$
Count $\mathbb{E}[\#\text{ heads}]$

$$X_i = \begin{cases} 0 & \text{if coin is tails} \\ 1 & \text{if coin is heads} \end{cases} \quad \text{and} \quad \mathbb{E}[X_i] = p$$

Let $X = \sum_{i=1}^{n} X_i$

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}[X_i] = np$$

Example Toss independent coins where each coin comes up heads w.p. $p \in [0,1]$
How many flips until first head?

$$\mathbb{E}[\#\text{ flips}] = \underbrace{\mathbb{E}[\#\text{ flips} \mid \text{first flip is heads}]}_{=1} \cdot \underbrace{\mathbb{P}[\text{first flip is heads}]}_{=p}$$

$$+ \underbrace{\mathbb{E}[\#\text{ flips} \mid \text{first flip is tails}]}_{=1+\mathbb{E}[\#\text{flips}]} \cdot \underbrace{\mathbb{P}[\text{first flip is tails}]}_{=1-p}$$

$$= p + (1-p)(1 + \mathbb{E}[\#\text{flips}])$$

$$\Longrightarrow \quad \mathbb{E}[\#\text{ flips}] = \frac{1}{p}$$

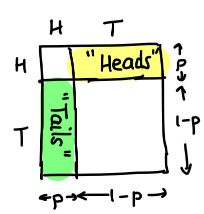
Sampling a Fair Coin from a Biased Coin

Suppose you have a biased coin that comes up heads with some unknown probability p
How can you use it to get a fair coin toss?

Von Neumann in 1951 came up with a strategy

- Flip the biased coin twice
- If results of the two flips are different, return the first one
  HT $\rightarrow$ return "Heads", TH $\rightarrow$ return "Tails"
- Otherwise repeat until success

Why does this return a fair coin toss?



$$\mathbb{P}[HT] = \mathbb{P}[TH] = p(1-p)$$

So, $\mathbb{P}[HT \mid \text{flips diff}]$

$$= \frac{p(1-p)}{2p(1-p)} = \frac{1}{2}$$

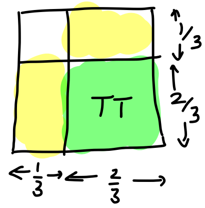
How many flips do we need ?

$\mathbb{P}$[each iteration succeeds] $= 2p(1-p) = q$ → How many times do we need to flip a biased coin until it comes up H ?

$\mathbb{E}$[# times until success] $= \frac{1}{q} = \frac{1}{2p(1-p)}$

Note There are better algorithms if we know the value of p

E.g. if $p = \frac{1}{3}$,



output "Heads" if we see TH or HT
"Tails" if we see TT

Collecting Pokemons – Gotta Catch 'Em All

How many Pokemon cards you need to buy to collect all N pokemons ?

Assume that each time we buy a card, we get a uniformly random Pokemon

Let Y = # cards to get all N pokemons

Let Y_i = # cards after we have $(i-1)$ pokemons to get i pokemons

$$Y = Y_1 + Y_2 + \cdots + Y_N$$

What is $\mathbb{E}[Y]$? $Y_1 = 1$

Y_N = # times we need to flip a $\frac{1}{N}$-biased coin to see heads

$\mathbb{E}[Y_N] = N$

Similarly, Y_i = # times we need to flip a $\frac{N-i+1}{N}$-biased coin to see heads

$\mathbb{E}[Y_i] = \frac{N}{N-i+1}$

Thus, by linearity of expectation

$$\mathbb{E}[Y] = \sum_{i=1}^{n} \mathbb{E}[Y_i] = \sum_{i=1}^{N} \frac{N}{N-i+1} = N \sum_{i=1}^{N} \frac{1}{N-i+1} = N \sum_{j=1}^{N} \frac{1}{j} \qquad (j = N-i+1)$$

$$= N \cdot H_N$$

↳ N^{th} Harmonic Number

$$\approx N \cdot \ln N$$