Cpts 515. 10/28/2020.

Probabilistic Algs.

(1). Monte-Carlo Algs

(2). Las Vegas Algs.

(1). Monte-Carlo Algs.

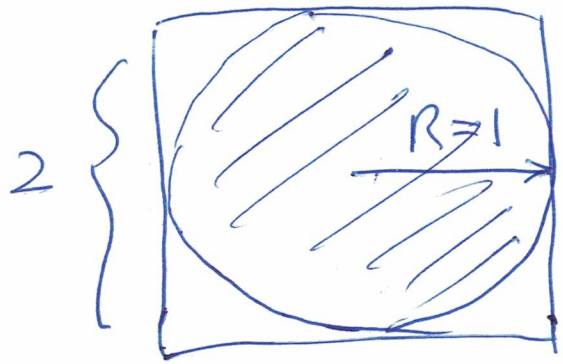
① they are pwbabilistic

② they will not guarantee correctness

③ running time is controlled.

Classic Examples.

Compute π.



throw darts to the square (for $N=1000$ times). Count the # of times the shaded area is hit. $=M$

$$\pi \approx \frac{4M}{N}.$$

Compute AB, and compare with C.

// I want to check $AB == C$?

// where A, B, C are $n \times n$ matrices.

Time: $O(n^3)$.

Monte Carlo Alg: // Random testing.

① create a random vector $d \in \{0,1\}^n$.

② ret $A.(B \cdot d) == C \cdot d$?

($O(n^2)$ steps)

Why does it work?

1. if the alg returns false; i.e.,

$A(B \cdot d) \neq C \cdot d$ then, indeed we have $AB \neq C$. Therefore, there is no false negatives:

2. if the alg returns true; i.e.,

$A(B \cdot d) == C \cdot d$, then we could still have $AB \neq C$. We do have false positives. The probability of false positives:

$A \cdot B \neq C \Rightarrow$ The probability that

$$A(B \cdot d) == C \cdot d \text{ holds}$$

is $\leq \frac{1}{2}$. // the prob is taken on // the d

Why? From $A(B \cdot d) == C \cdot d$, we can treat the $d = (d_1, \ldots, d_n) \in \{0,1\}^n$ as unknowns.

From $AB \neq C$, we take $E = AB - C \neq 0$.

That is, there is at least one element in the matrix E that is not zero, say it is l_{11}.

$$\begin{bmatrix} e_{11} & \cdots & \cdots \\ \vdots & & \vdots \\ \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} d_1 \\ \vdots \\ d_m \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

$$\Rightarrow \quad \underbrace{e_{11} d_1 + \cdots + e_{1n} d_n}_{\#} = 0 \qquad (*)$$

0

For each fixed $d_2, \ldots, d_n$, <u>at most</u> one of $d_1 = 0$, $d_1 = 1$ will make $(*)$ true and hence $A(B \cdot d) = C \cdot d$ true. Hence, the prob $\leq \frac{1}{2}$.

Improved Version:

On input A, B, C

we randomly select vectors in $\{0,1\}^n$:

$\vec{d}_1, \ldots, \vec{d}_k$

and test $A(B \cdot \vec{d}_i) == C \cdot \vec{d}_i$ for all i.

if one of these tests return false,
then, we have $AB \neq C$;

if all of these tests return true,
then we have $AB \neq C$ with probability $\leq \frac{1}{2^k}$.

Las Vegas Algs.

① They are probabilistic Algs

② they are correct (output is deterministic).

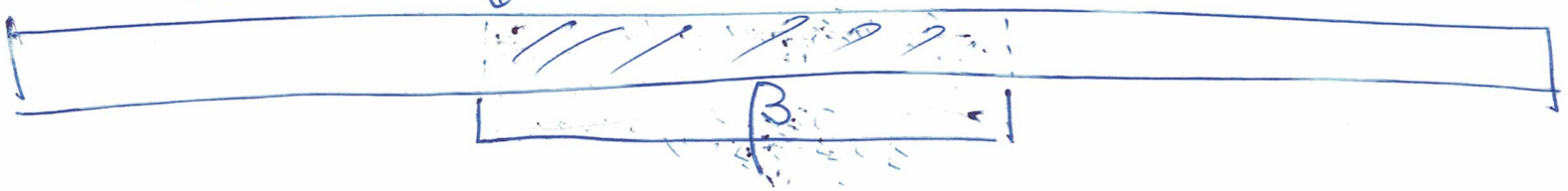
Classic Example: randomized quicksort:

1. input an array
2. we create a random shuffle of the input array
3. Run quicksort on the shuffled array.

One more Example of Las-Vegas Algs.

Karp-Rabin Alg. for pattern matching.

α is a long string;



β is a shorter string, pattern.

Want to find an appearance of β in α.

Karp-Rabin:

For each position i (from 1 to n),

Compute hash(pattern) = hash(β).

and compute hash($\alpha[i, \ldots, i+m-1]$)

if the two hash vals are the same,

compare the two strings;

($\beta \stackrel{?}{=} \alpha[i, \ldots, i+m-1]$)

if the two hash vals are not the same, continue (with $i++$);

① . fast way to update fnn

hash $(\alpha[i], \ldots, \alpha[i+m-1])$ to

hash $(\alpha[i+1], \ldots, \alpha[i+m])$

How? Let q be a random prime number. Then

$$\text{hash}(0,1,1,1) = 0\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 \mod q.$$

Update:

$$\text{hash}(1,1,1,0) = 1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 \mod q$$

$$= \left[(\text{hash}(0,1,1,1) - 0\cdot 2^3)\cdot 2 + \right.$$

$0 \cdot 2^0 \Big] \mod q.$

Analysis of error probability:

error $\equiv$ hash($a[0..m-1]$) = hash($b[0..m-1]$)

(m bits)

but $a[0..m-1] \neq b[0..m-1]$.

Assume that this probability is $\varepsilon > 0$. Then since we have at most n postions in α. (n is the length of input α). So, total error probability $\leq n \cdot \varepsilon$.

We want $n \cdot \varepsilon \leq 1/2$. (later we can amplify it to any small probability).

Trick: pick the random prime number q from a large range, say $N = \{2, \ldots, N\}$.

How big is the N? For each i, the error happens when $|a-b| \bmod q = 0$ (*)

(here a, b are unsigned numbers for $a[0..m-1]$ and $b[0..m-1]$)

Notice that q is random, but a, b are NOT.
// q is randomly chosen from all primes in
// the range of N. I want find N s.t.
$\varepsilon \leq \frac{1}{2n}$ can be reached.

$|a-b|$ takes max. 2^m values; Then
$|a-b|$ has at most m distinct prime
divisors. Therefore, the # of q's that
make (*) true is at most m many.

We need only to choose N large enough so that between $2..N$, we have more than $2nm$ primes so that the error $\varepsilon \leq \frac{m}{2nm} = \frac{1}{2n}$.