§Randomized Algorithms

In a randomized algorithm (probabilistic algorithm), we make some random choices.

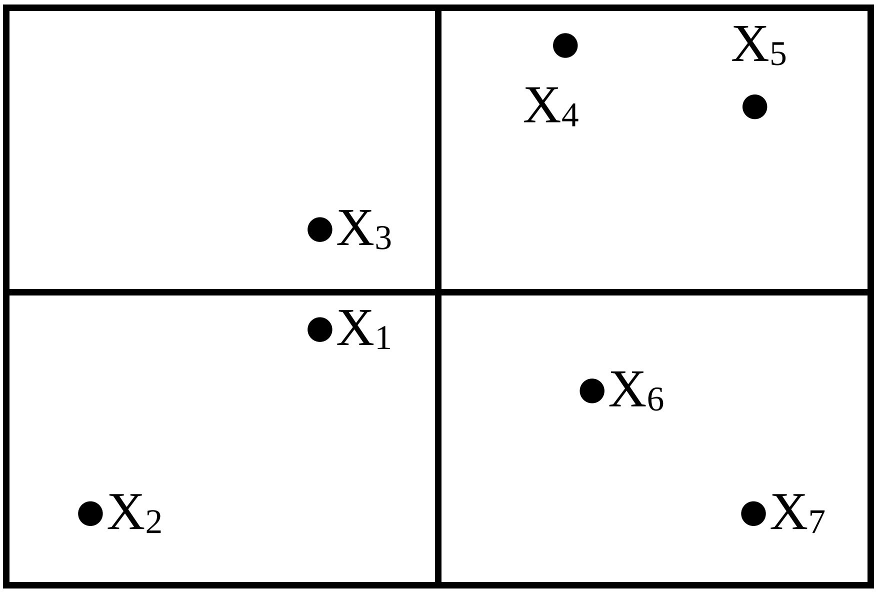
2 types of randomized algorithms:

(1) for optimization problems, a randomized algorithm gives an optimal solution. The average case time-complexity is more important than the worst case time-complexity.

(2) For decision problems, a randomized algorithm may make mistakes. The probability of producing wrong solutions is very small.

- A randomized algorithm to solve the closest pair problem
- This problem can be solved by the divide-and-conquer approach in O(nlogn) time.
- The randomized algorithm:

partition the points into several clusters:



We only calculate distances among points within the same cluster.
Similar to the divide-and-conquer strategy.
There is a dividing process, but no merging process.

Algorithm 11.1 A Randomized Algorithm for Finding a Closest Pair

Input: A set S consisting of n elements $x_1, x_2, \ldots, x_n$, where $S \subseteq R^2$.

Output: The closest pair in S.

Step 1. Randomly choose a set $S_1=\{x_{i_1}, x_{i_2}, \ldots, x_{i_m}\}$ where $m=n^{2/3}$. Find the closest pair of S_1 and let the distance between this pair of points be denoted as δ.

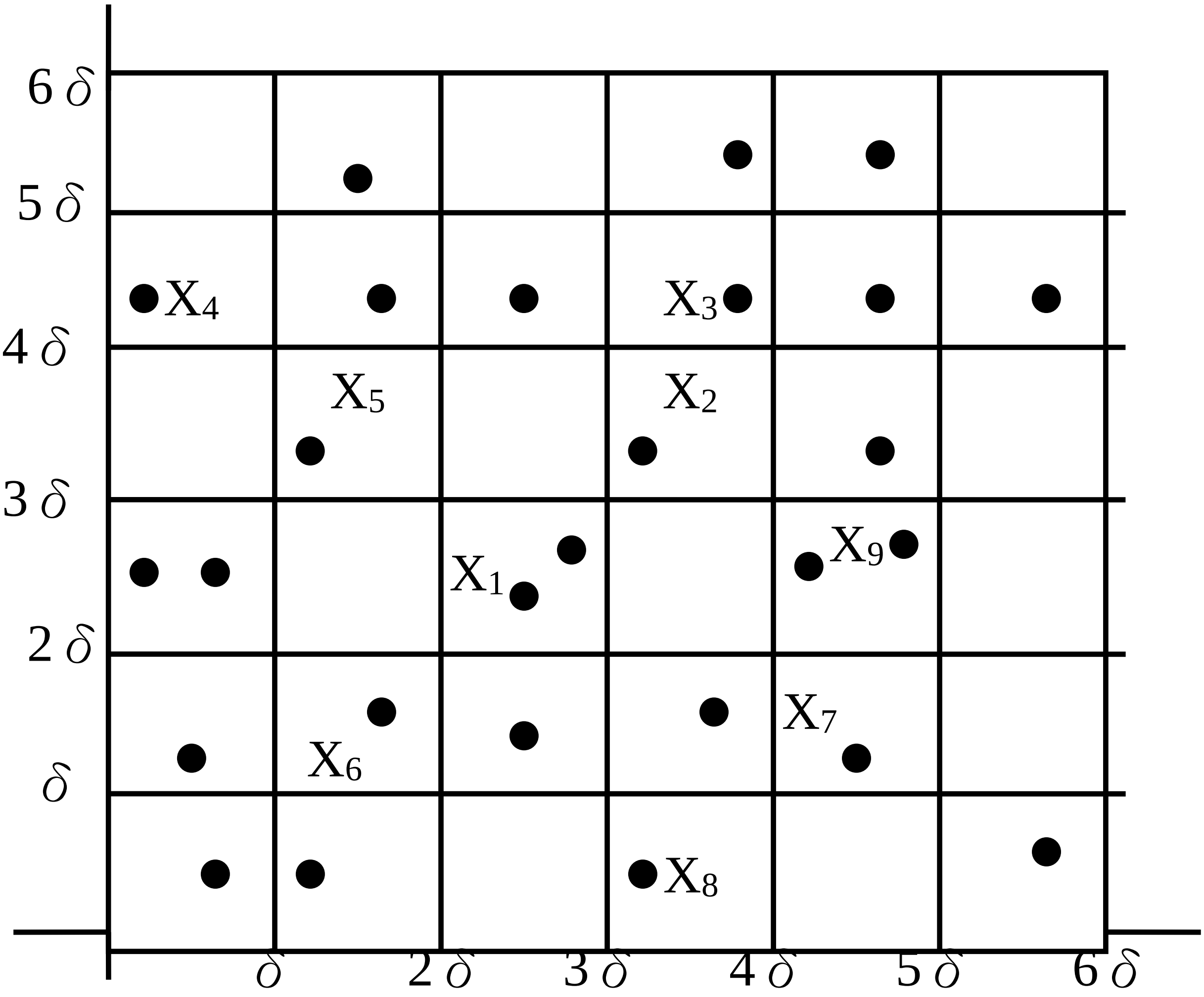
Step 2. Construct a set of squares T with mesh-size δ.

Step 3. Construct four sets of squares T_1, T_2, T_3 and T_4 derived from T by doubling the mesh-size to 2δ.

Step 4. For each T_i, find the induced decomposition $S=S_1^{(i)} \cup S_2^{(i)} \cup \ldots \cup S_{k_i}^{(i)}$, $1 \le i \le 4$, where $S_j^{(i)}$ is a non-empty intersection of S with a square of T_i.

Step 5. For each $x_p, x_q \in S_j^{(i)}$, compute $d(x_p, x_q)$. Let x_a and x_b be the pair of points with the shortest distance among these pairs. Return x_a and x_b as the closest pair.

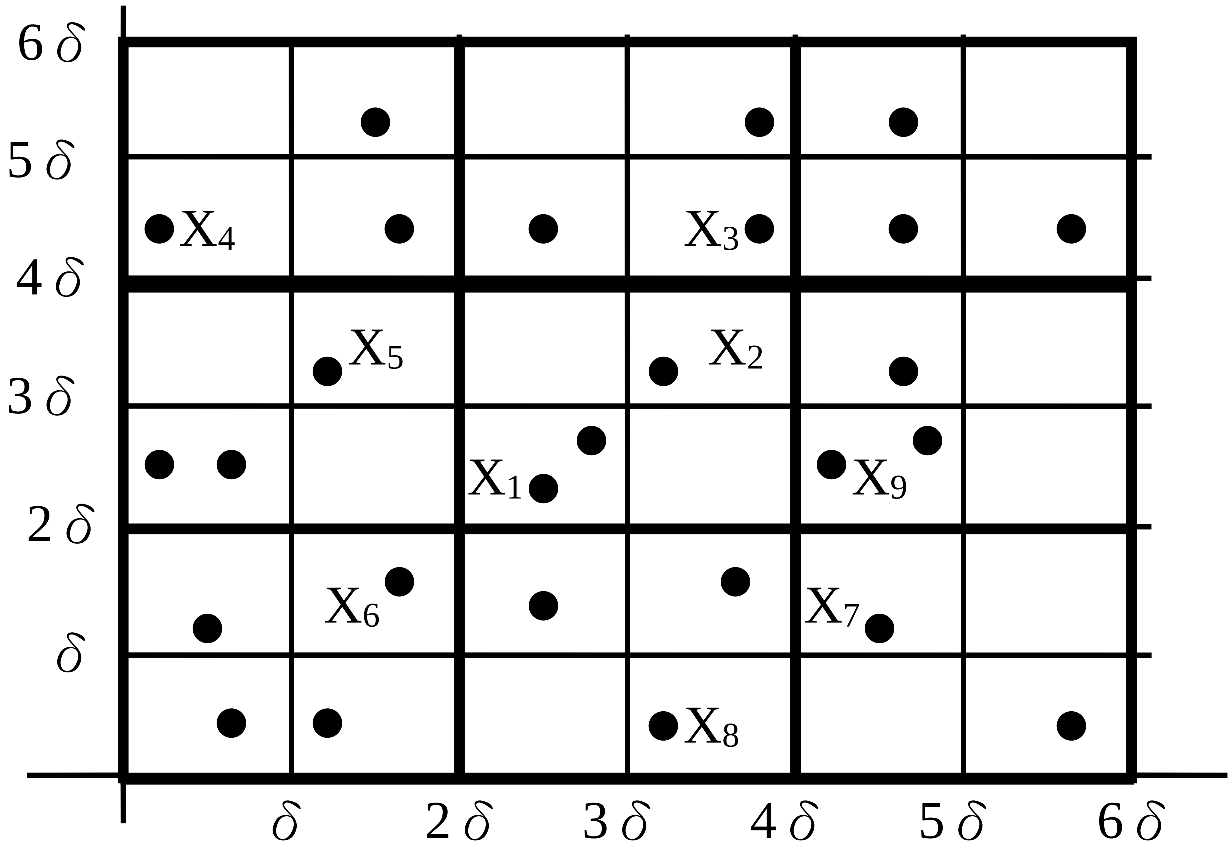
e.g.



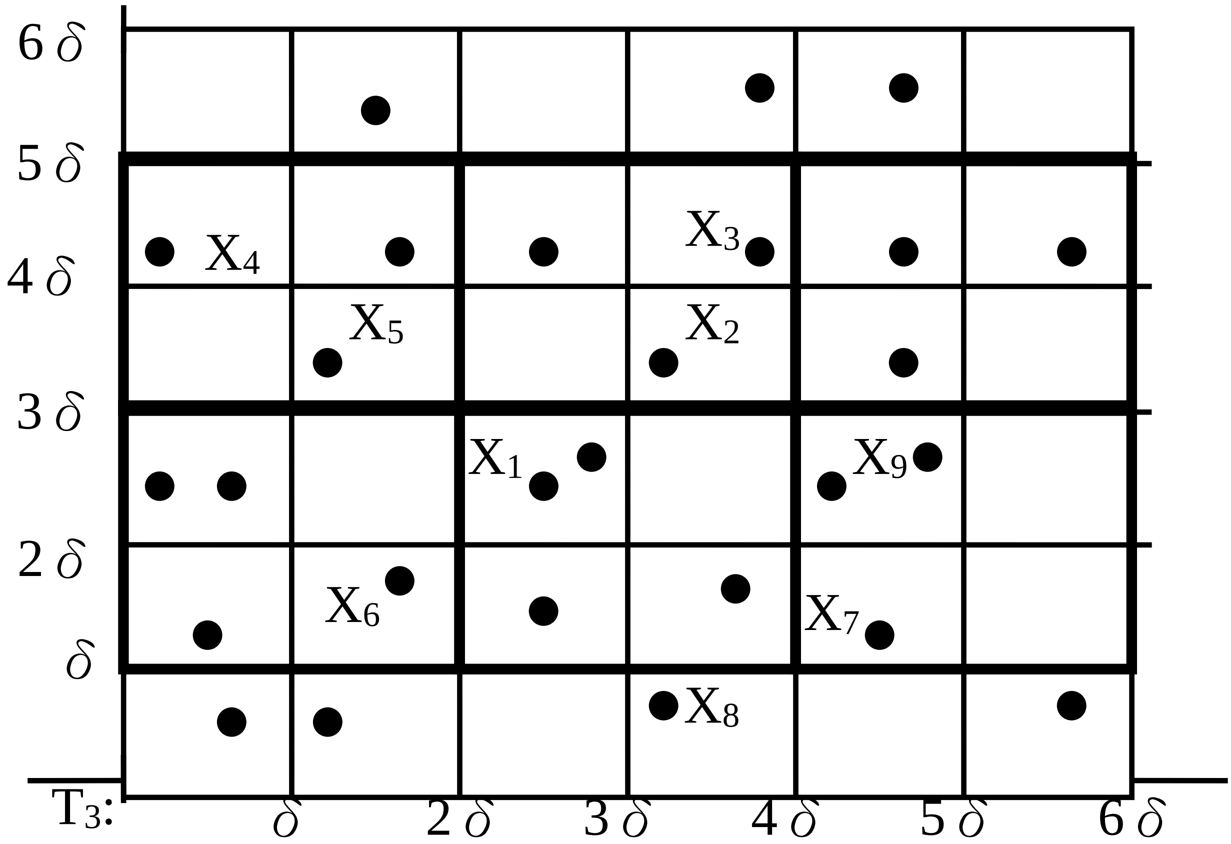
There are 27 points.

$S_1 = \{x_1, x_2, \ldots, x_9\}$, $\delta = d(x_1, x_2)$

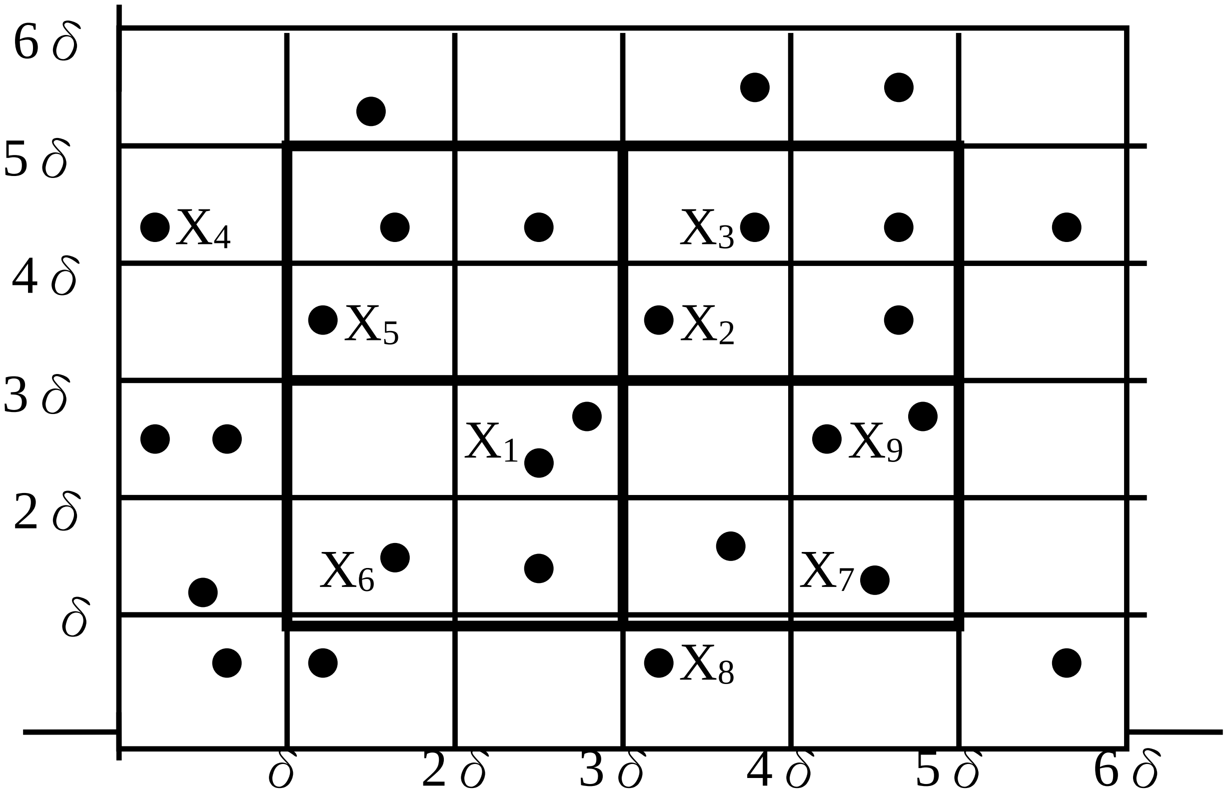
T$_1$:


6 δ
5 δ
4 δ
3 δ
2 δ
δ
X4
X3
X5
X2
X1
X9
X6
X7
X8
δ
2 δ
3 δ
4 δ
5 δ
6 δ

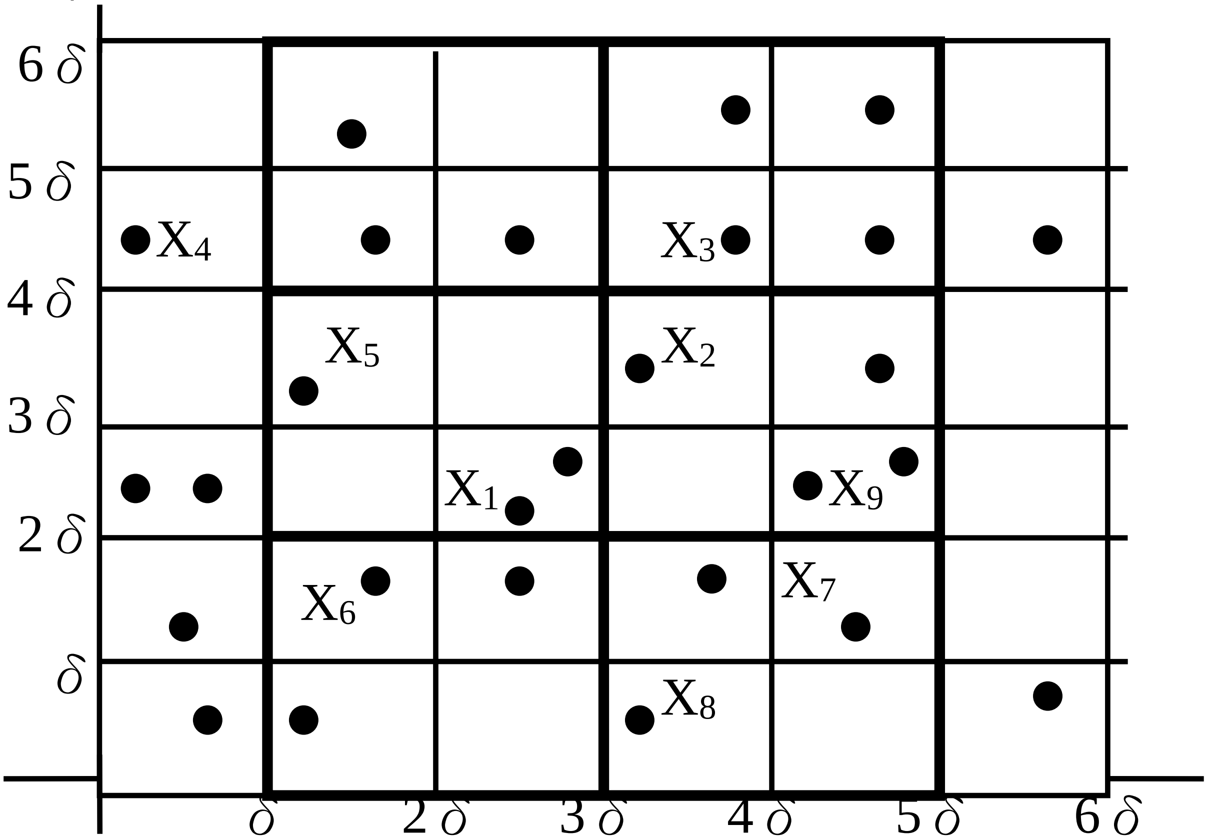
T$_2$:


6 δ
5 δ
4 δ
3 δ
2 δ
δ
X4
X3
X5
X2
X1
X9
X6
X7
X8
δ
2 δ
3 δ
4 δ
5 δ
6 δ

T$_3$:



T_4:



- time-complexity : O(n) in average

step 1: $O(n^{\frac{8}{9}}) + O(n^{\frac{2}{3}}) = o(n)$

randomly choose $(n^{\frac{2}{3}})^{\frac{2}{3}} = n^{\frac{4}{9}}$ from the $n^{\frac{2}{3}}$ points.

Straight forward method for the $n^{\frac{4}{9}}$ points: $O(n^{\frac{8}{9}})$

recursively applying the algorithm once: $O(n^{\frac{2}{3}})$

$$\left.\begin{array}{l}\text{step 2:}\\ \text{step 3:}\\ \text{step 4:}\end{array}\right\} O(n)$$

step 5: O(n)with probability $1-2e^{-cn^{\frac{1}{6}}}$

- How many distance computations in step 5?

δ: mesh-size in step 5

T: partition in step 5

N(T): # of distance computations in partition T

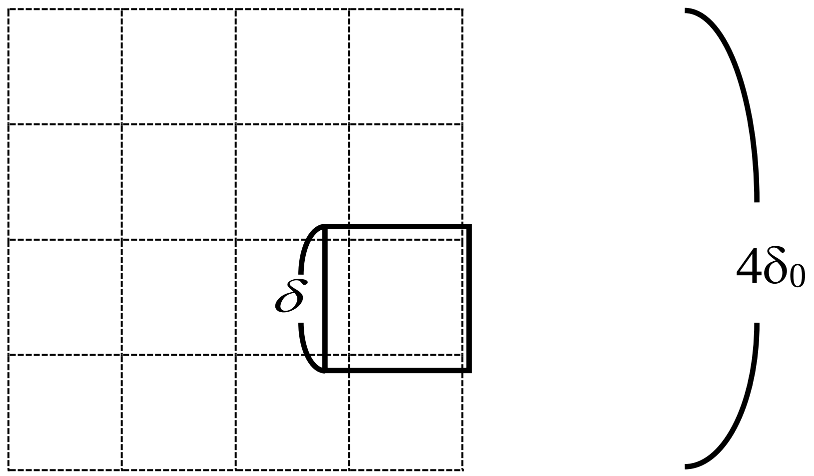
fact: There exists a particular partition T_0, whose mesh-size is δ_0 such that

(1) $N(T_0) \leq c_0 n$.

(2) The probability that $\delta \leq \sqrt{2}\,\delta_0$ is $1-2e^{-cn^{\frac{1}{6}}}$.

$T_1, T_2, \ldots, T_{16}$:

mesh-size: $4\delta_0$

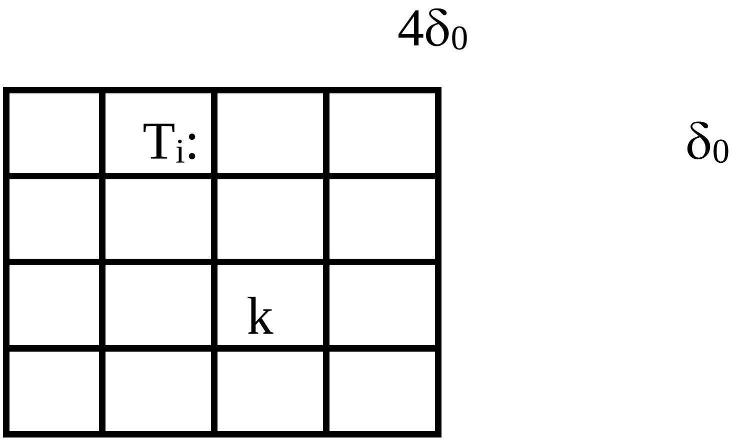


The probability that each square in T falls into at least one square of T_i , i = 1, 2, …, 16 is $1-2e^{-cn^{\frac{1}{6}}}$.

The probability that

$$N(T) \le \sum_{i=1}^{16} N(T_i)$$

is $1-2e^{-cn^{\frac{1}{6}}}$.



Let the square in T_0 with the largest number of elements among the 16 squares have k elements.

$$\frac{k(k-1)}{2} = O(k^2), \frac{16k(16k-1)}{2} = O(k^2)$$

$N(T_0) \le c_0 n \quad \Rightarrow \quad N(T_i) \le c_i n$

$N(T) \le \sum_{i=1}^{16} N(T_i) = O(n)$ with probability $1\text{-}2e^{-cn^{\frac{1}{6}}}$.

- A randomized algorithm to test whether a number is prime.

This problem is very difficult and no polynomial algorithm has been found to solve this problem.

traditional method:

use 2,3,… $\sqrt{N}$ to test whether N is prime.

input size of N : B=$\log_2$N (binary representation)

$\sqrt{N}$ =$2^{B/2}$, exponential function of B

$\therefore$ $\sqrt{N}$ can not be viewed as a polynomial function of the input size.

Algorithm 11.3 A Randomized Prime Number Testing Algorithm

Input: A positive number N, and a parameter m.

Output: Whether N is a prime or not, with probability of being correct $1-\varepsilon = 1-2^{-m}$.

Step 1. Randomly choose m numbers $b_1, b_2, \ldots, b_m$, $1 \le b_1, b_2, \ldots, b_m < N$ where $m \ge \log_2(1/\varepsilon)$.

Step 2. For each b_i, test whether $W(b_i)$ holds where $W(b_i)$ is defined as follows:

(1) $b_i^{N-1} \ne 1 \bmod N$

or (2) $\exists\ j$ [$\frac{N-1}{2^j} = k$ is an integer and the greatest common divisor of $(b_i)^k-1$ and N is not 1 or N.]

If any $W(b_i)$ holds, then return N as a composite number, otherwise, return N as a prime.

e.g.1. N = 12

choose 2, 3, 7

$2^{12-1} = 2048 \neq 1 \bmod 12 \Rightarrow$ 12 is a composite number.

e.g.2. N = 11

choose 2, 5, 7

(i) $2^{11-1}=1024 \equiv 1 \bmod 11$

$j=1$, $(N-1)/2^j=\frac{11-1}{2}=5$

GCD(2^5-1, 11) = 1

W(2) does not hold .

(ii) $5^{11-1}=9765625 \equiv 1 \bmod 11$

GCD(5^5-1, 11) = 11

W(5) does not hold .

(iii) $7^{11-1}=282475249 \equiv 1 \bmod 11$

GCD(7^5-1, 11) = 1

W(7) does not hold .

$\Rightarrow$ 11 is a prime number with the probability of correctness being $1-2^{-3}=\frac{7}{8}$.

<Theorem>

(1) If W(b) holds for any $1 \leq b < N$, then N is a composite number .

(2) If N is composite, then $\frac{N-1}{2} \leq |\{ b \mid 1 \leq b < N,$ W(b) holds $\}|$.

- A randomized algorithm for pattern matching
  pattern string : X   length : n

text string : Y   length : m   , $m \ge n$

to find the first occurrence of X as a consecutive substring of Y .

Assume that X and Y are binary strings.

e.g. X = 01001 , $Y = 101\underbrace{01001}_{X}11$

- straight forward method : O(m n)
- Knuth-Morris-Pratt's algorithm : O(m)
- the randomized algorithm : O(m k) with a mistake of small probability.

k:# of testings

$X = x_1 x_2 \ldots x_n \in \{0,1\}$

$Y = y_1 y_2 \ldots y_m \in \{0,1\}$

Let $Y(i) = y_i\, y_{i+1} \ldots. y_{i+n-1}$

A match occurs if X=Y(i) for some i .

binary values of X and Y(i):

$B(X) = x_1 \cdot 2^{n-1} + x_2 \cdot 2^{n-2} + \ldots + x_n$

$B(Y(i)) = y_i \cdot 2^{n-1} + y_{i+1} \cdot 2^{n-2} + \ldots + y_{i+n-1}$, $1 \le i \le m-n+1$

Let p be a randomly chosen prime number in $\{1,2,\ldots,nt^2\}$, where t = m - n + 1.
$(x_i)_p = x_i \bmod p$

fingerprints of X and Y(i):

$B_p(x) = (\,(\,(\,(x_1 \cdot 2)_p + x_2)_p \cdot 2)_p + x_3)_p \cdot 2 \cdots$

$B_p(Y(i)) = (\,(\,(\,(y_i \cdot 2)_p + y_{i+1})_p \cdot 2 + y_{i+2})_p \cdot 2 \cdots$

$\Rightarrow B_p(Y(i+1)) = (\,(\,(\,(B_p(Y_i) - (\,(2^{n-1})_p \cdot y_i)_p\,)_p \cdot 2)_p + y_{i+n})_p$

$= (\,(B_p(Y_i) - 2^{n-1} \cdot y_i) \cdot 2 + Y_{i+n})_p$

If X=Y(i), then $B_p(X) = B_p(Y(i))$, but not vice versa .

e.g. X = 10110 , Y = 110110

n = 5 , m = 6 , t = m - n + 1 = 2

suppose P=3 .

$B_p(X) = (22)_3 = 1$

$B_p(Y(1)) = (27)_3 = 0$

$\Rightarrow X \neq Y(1)$

$B_p(Y(2)) = (\,(0-2^4)\,2+0)_3 = 1 \Rightarrow X = Y(2)$

e.g. X = 10110 , Y = 10011 , P = 3

$B_p(X) = (22)_3 = 1$

$B_p(Y(1)) = (19)_3 = 1$

$\Rightarrow X = Y(1)$ WRONG!

(1) If $B_p(X) \neq B_p(Y(i))$, then $X \neq Y(i)$.
(2) If $B_p(X) = B_p(Y(i))$, we may do a bit by bit checking or compute k different fingerprints by using k different prime numbers in $\{1,2,\ldots nt^2\}$.

Algorithm 11.4 A Randomized Algorithm for Pattern Matching

Input: A pattern $X = x_1 x_2 \cdots x_n$, a text $Y = y_1 y_2 \cdots y_m$ and a parameter k.

Output: (1)No, there is no consecutive substring in Y which matches with X.

(2)Yes, $Y(i) = y_i y_{i+1} \ldots y_{i+n-1}$ matches with X which is the first occurrence.

If the answer is "No" , there is no mistake.

If the answer is "Yes" , there is some probability that a mistake is made.

Step 1. Randomly choose k prime numbers $p_1, p_2, \ldots, p_k$ from $\{1,2,\ldots,nt^2\}$, where $t = m - n + 1$.

Step 2. $i = 1$.

Step 3. $j = 1$.

Step 4. If $B(X)_{Pj} \neq (B(Y_i))_{pj}$, then go to step 5.

If $j = k$, return Y(i) as the answer.

$j = j + 1$.

Go to step 4.

Step5. If $i = t$, return "No, there is no consecutive substring in Y which matches with X."

$i = i + 1$.

Go to Step 3.

e.g. $X = 10110$, $Y = 100111$, $P_1 = 3$, $P_2 = 5$

$$\left.\begin{array}{l} B_3(X) = (22)_3 = 1 \\ B_5(X) = (22)_5 = 2 \\ B_3(Y(2)) = (7)_3 = 1 \\ B_5(y(2)) = (7)_5 = 2 \end{array}\right\} X \overset{?}{=} Y(2)$$

Choose one more prime number, $P_3 = 7$

$$B_7(x) = (22)_7 = 1$$

$$B_7(Y(2)) = (7)_7 = 0$$

$\Rightarrow X \neq Y(2)$

- How often does a mistake occur?

When a mistake occurs in X and Y(i), $B(X) - B(Y(i)) \neq 0$, and p_j divides $| B(X) - B(Y(i)) |$ for all p_j's.

Let $Q = \prod_{i \text{ where } p_j \text{ divides } |B(x)-B(Y(i))|} |B(x) - B(Y(i))|$

$Q < 2^{n(m-n+1)}$ $\left(\because B(x)<2^n, \text{ at most } (m-n+1)\ B(Y(i))\text{'s} \quad 2^n \underbrace{2^n \cdots 2^n}_{m-n-1} \right)$

<Theorem>

If $u \geq 29$ and $a<2^u$, then a has fewer than $\pi(u)$ diffferent prime number divisors where $\pi(u)$ is the number of prime numbers smaller than u.

Assume $nt \geq 29$.

$$Q < 2^{n(m-n+1)} = 2^{nt}$$

$\Rightarrow$ Q has fewer than $\pi(nt)$ different prime number divisors.

- If p_j is a prime number selected from $\{1, 2, \ldots, M\}$, the probability that p_j divides Q is less than $\frac{\pi(nt)}{\pi(M)}$

- If k different prime numbers are selected from {1, 2, …nt^2} , the probability that a mistake occurs is less than $\left(\frac{\pi(nt)}{\pi(nt^2)}\right)^k$ provided nt ≥ 29.
- How do we estimate $\left(\frac{\pi(nt)}{\pi(nt^2)}\right)^k$?

<Theorem> For all u ≥ 17, $\frac{u}{\ln u} \le \pi(u) \le 1.25506\frac{u}{\ln u}$

$$\frac{\pi(nt)}{\pi(nt^2)} \le 1.25506 \cdot \frac{nt}{\ln nt} \cdot \frac{\ln(nt^2)}{nt^2}$$

$$= \frac{1.25506}{t}\left(1 + \frac{\ln(t)}{\ln(nt)}\right)$$

e.g. n = 10 , m = 100 , t = m - n + 1 = 91

$$\frac{\pi(nt)}{\pi(nt^2)} \le 0.0229$$

Let k=4

$(0.0229)^4 \approx 2.75\times10^{-7}$

- A randomized algorithm for interactive proofs

two persons:

A: a spy

B: the boss of A

When A wants to talk to B , how does B know that A is the real A, not an enemy imitating A ?

Method I: a trivial method

B may ask the name of A's mother.

(a private secret)

disadvantage:

The enemy can collect the information ,and imitate A the next time.

Method II:

B may send a Boolean formula to A and ask A to determine its satisfiability. (an NP-complete problem).

It is assumed that A is a smart person and knows how to solve this NP-complete problem.

B can check the answer and know whether A is the real A or not.

disadvantage:

The enemy can study methods of mechanical theorem proving and sooner or later he can imitate A.

In Methods I and II, A and B have revealed too much.

Method III:

B can ask A to solve a quadratic nonresidue problem in which the data can be sent back and forth without revealing much information.

Definition:

$(x, y) = 1$, y is a quadratic residue mod x if $z^2 \equiv y \bmod x$ for some z, $0 < z < x$, $(x, z) = 1$ and y is a quadratic nonresidue mod x if otherwise.

Let

QR = $\{(x, y) \mid y$ is a quadratic residue mod $x\}$

QNR = $\{(x, y) \mid y$ is a quadratic nonresidue mod $x\}$

e.g. $x = 9$, $y = 7$

$1^2 \equiv 1 \bmod 9$

$2^2 \equiv 4 \bmod 9$

$3^2 \equiv 0 \bmod 9$

$4^2 \equiv 7 \bmod 9$

$5^2 \equiv 7 \bmod 9$

$6^2 \equiv 0 \bmod 9$

$7^2 \equiv 4 \bmod 9$

$8^2 \equiv 1 \bmod 9$

$(9,7) \in QR$

but $(9,5) \in QNR$

Method:

(1) A and B know x and keep x confidential .
B knows y.

(2) Action of B:

(i) Randomly choose m bits: $b_1, b_2, \ldots, b_m$ where m is the length of the binary representation of x.

(ii) Find $z_1, z_2, \ldots, z_m$ s.t. $(z_i, x)=1$ for all i .

(iii) Compute $w_1, w_2, \ldots, w_m$:

$w_i \leftarrow z_i^2 \bmod x$ if $b_i=0$

$w_i \leftarrow (z_i^2 \cdot y) \bmod x$ if $b_i=1$

(iv) Send $w_1, w_2, \ldots, w_m$ to A.

(3) Action of A:

(i) Receive $w_1, w_2, \ldots, w_m$ from B.

(ii) Compute $c_1, c_2, \ldots, c_m$:

$c_i \leftarrow 0$ if $(x, w_i) \in QR$

$c_i \leftarrow 1$ if $(x, w_i) \in QNR$

Send $c_1, c_2, \ldots, c_m$ to B.

(4) Action of B:

(i) Receive $c_1, c_2, \ldots, c_m$ from A.

If $(x, y) \in QNR$ and $b_i = c_i$ for all i, then A is the real A (with probability $1\text{-}2^{-m}$).