1. Fingerprinting and string comparison + pattern matching
2. primality testing

Schwartz-Zippel thm.

$$\Pr\left(Q(r_1,\dots,r_n)=0 \mid Q\not\equiv 0\right) \leq \frac{\deg(Q)}{|S|}$$

for $r_i \in_R S$.

Problem. Verify whether two strings $X, Y \in \{0,1\}^n$ are equal.

deterministically $O(n)$ $\quad X=(x_1,\dots,x_n)$
$\quad Y=(y_1,\dots,y_n)$

USE S-Z theorem

interpret X and Y as multivariable polynomials:

$$X(z_1,\dots,z_n) = \sum_{i=1}^n x_i z_i \mod p$$

$$Y(z_1,\dots,z_n) = \sum_{i=1}^n y_i z_i \mod p$$

$$X(z_1,\dots,z_n) - Y(z_1,\dots,z_n) \overset{?}{\equiv} 0$$

Choose $r \in \{0,1\}^n$ ←

by schwartz-zippel theorem

$$\Pr\left(X(r_1,\dots,r_n)-Y(r_1,\dots,r_n) \,\middle|\, [X-Y](\vec{z})\not\equiv 0\right) \leq \frac{\deg[X-Y]}{2} = \frac{1}{2}$$

Context: Database comparison

-> two distant databases X and Y are they the same?
-> is the above method efficiant in number of transmitted bits?
NO! random r needed to calculate the fingerprint is as long as the database.

Solution 1

Interpret both X and Y as numbers

$$\mathrm{num}(X) = \sum_{i=1}^{n} x_i 2^{i-1}$$

$$\mathrm{num}(Y) = \sum_{i=1}^{n} y_i 2^{i-1}$$

Compare:

$X \bmod p$ with $Y \bmod p$

if p is well chosen, fingerprints are small and probability of error is also small.

When does this give a wrong answer? ($X \neq Y$, but $X \equiv Y \mod p$)

$X - Y \equiv 0 \mod p$ (read as $X-Y$ is divisible by p).

$\Pi(k)$ - all primes smaller than k. $\boxed{\Pi(k) \approx \frac{k}{\ln k}}$

$\boxed{\text{for } k \geq 29 \quad \Pi(k) \leq 1.2\ldots \frac{k}{\ln k}}$

$$\Pr\left(X-Y\equiv 0 \bmod p \;\middle|\; X\neq Y\right)=\frac{\#\text{ bad primes}}{\#\text{ primes we choose from}}=\frac{n}{\pi(t)}\leq\frac{\ln t\cdot n}{t}$$

\# bad primes: How many divisors can $X-Y$ have at most?

what is the largest value of $X-Y$?

$X-Y<2^n$

Smallest number with n prime divisors

$$=\prod_{i=1}^{n} p_i > 2^n = \prod_{i=1}^{n} 2$$

$p_i > p_1 = 2$

p_i – ith smallest prime

\# bad primes $< n$

for $t = \boxed{t\cdot n\log(tn)}$

$$\Pr<\frac{\ln(t\cdot n\cdot\log(tn))\cdot n}{t\cdot n\log(tn)}$$

$$\in O\left(\frac{1}{t}\right)$$

How many bits do we need to send? for $t=n$ we need to send prime p $O(\log n)$ bits and the hash $O(\log n)$

what we did:

$$X=\sum_{i=0}^{n} x_i z^{i-1} \bmod p$$

method 1:

choose $z=2$ and randomize over p

method 2:

choose p and randomize over z

Method 2 can be analysed using polynomial comparison.

$X(z)$ and $Y(z)$ are polynomials, by S-Z

$$Pr\left[(X-Y)(r)=0 \,\middle|\, (X-Y)(z)\neq 0\right] \leq \frac{\deg(X-Y)}{|S|} = \frac{n-1}{|S|}$$

$r \in_R S$

to match method 1 we want this to be roughly $\frac{1}{n}$

$\Rightarrow |S|$ is roughly n^2. $\Rightarrow p$ needs to be larger than n^2.

How many bits do we need to send?

Prime $p \sim O(\log n)$

number $r < p \sim O(\log n)$

3rd method:

choose a random polynomial $P \bmod p$ and evaluate $P(\text{num}(X))$ and $P(\text{num}(Y))$ and compare. $\rightsquigarrow$ this is an idea behind universal hashing.

Pattern matching

X - a string of length n

Y - a string of length m with $m<n$ | WLOG $X, Y \in \{0,1\}^n$

Is Y a substring of X?

Is Y a substring of X:

Naive algoritum $\approx O(m \cdot n)$ comparisons

Better solutions (Knuth-Morris) $\approx O(m+n)$ comparisons

Rabin-Karp in $O(m+n)$ time = Monte Carlo

Main idea: Compare fingerprints.

Imagine calculating fingerprints is for free. How many comparisons?

Each fingerprint is $O(\log m)$ bits long

$$\approx O(n \cdot \log m)$$

this is not what is meant in the slides.

Strings are arrays of objects:

both X and Y are in memory of a computer in an indexed array, that is each x_i and y_i need to be addressed separately in the memory. This is the expensive operation.

So in analysis of Rabin Karp algorithm, the expensive operation is calculating the hash $num(Y) = \sum_{i=1}^{m} x_i 2^{m-i}$ mod p because need to access each bit of array Y. $\approx O(m)$

Comparison of fingerprints is comparison of two integers, therefore in $O(1)$.

Naive calculation of hashes results in $O(m \cdot n)$

What we need is cheaper fingerprint calculation:

let $X_j = (x_j, \dots, x_{j+m-1})$

$$F(x_j) = x_j \cdot 2^{m-1} + x_{j+1} \cdot 2^{m-2} + \dots + x_{j+m-1}$$

$$F(x_{j+1}) = x_{j+1} \cdot 2^{m-1} + x_{j+2} \cdot 2^{m-2} + \dots + x_{j+m}$$

$$F(x_{j+1}) = 2 \cdot [F(x_j) - 2^{m-1} \cdot x_j] + x_{j+m}$$

Time analysis

The hash of Y : $F(Y)$ m steps

First hash of X_1 : $F(X_1)$ m steps

n following hashes each in $O(1)$ n steps

$\Bigg\}\ O(m+n)$