Algoritmos Aleatorizados

Verifying set equality

- String Matching – Rabin-Karp Algorithm

RANDOMIZED ALGORITHMS, 7.4-7.6

Verifying set equality

A ?= B

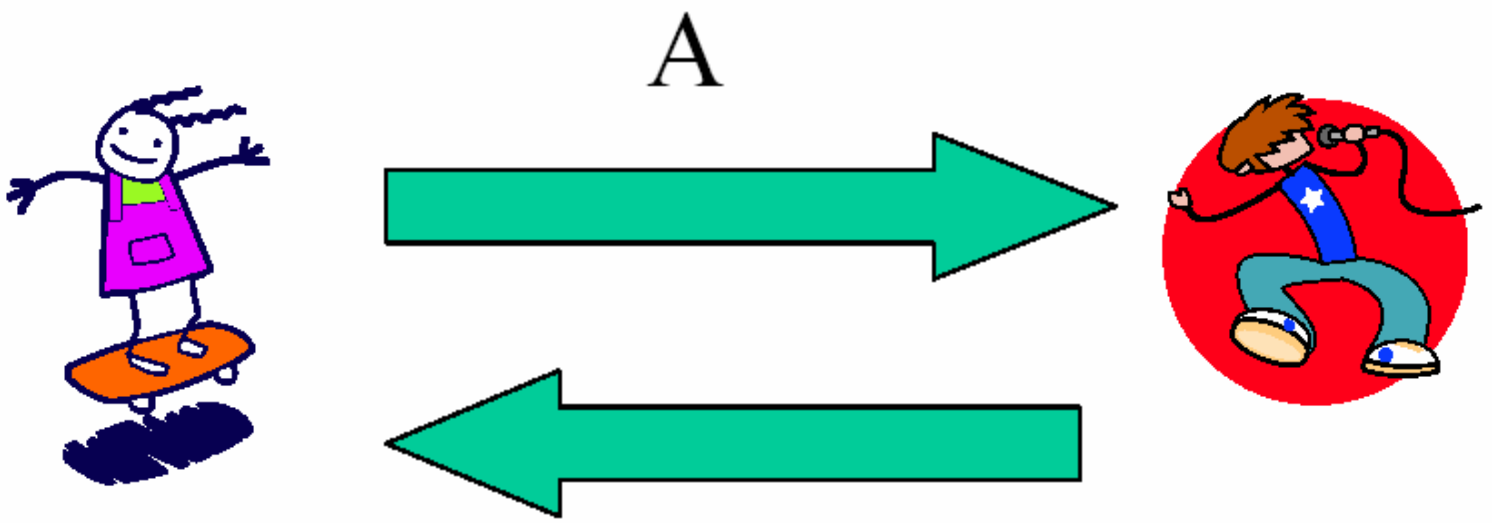
Alice

Bob

A=0100001111101……

B=010010111101….

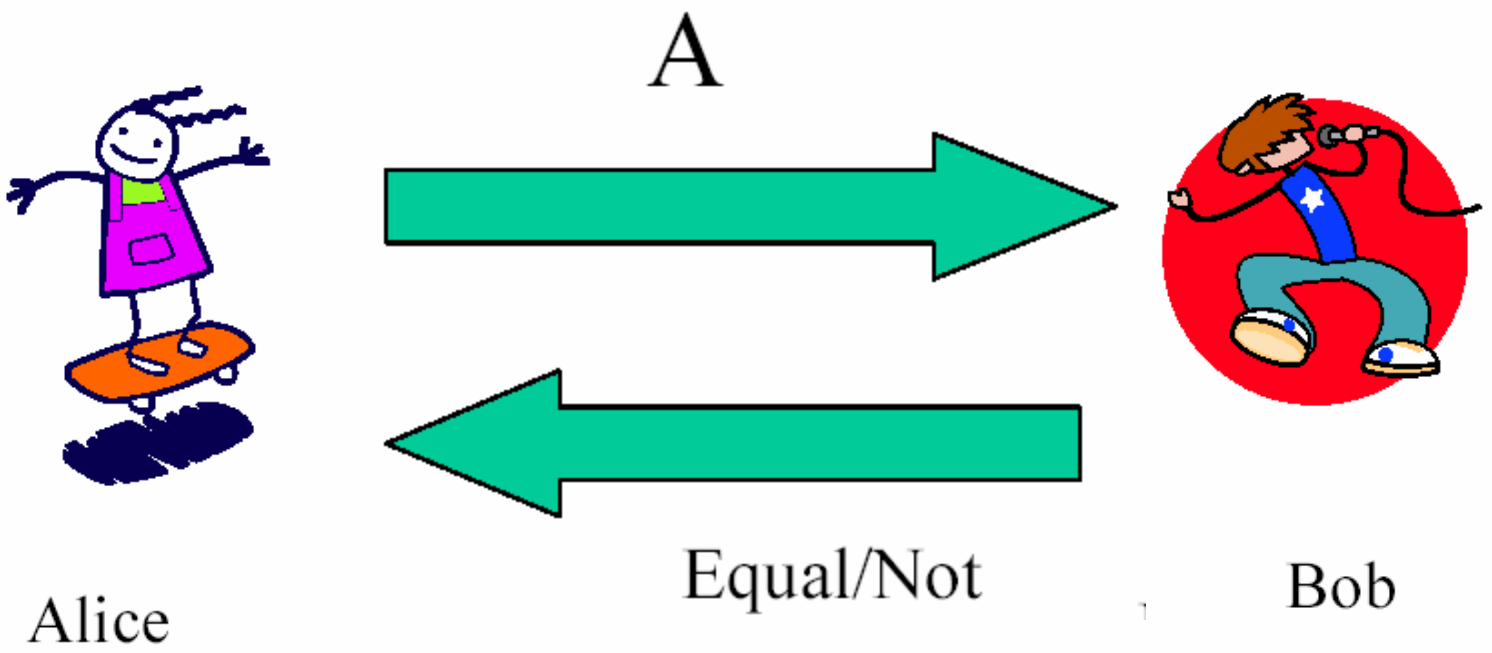
Verifying set equality



A=010000111101……

B=010010111101….

Verifying set equality

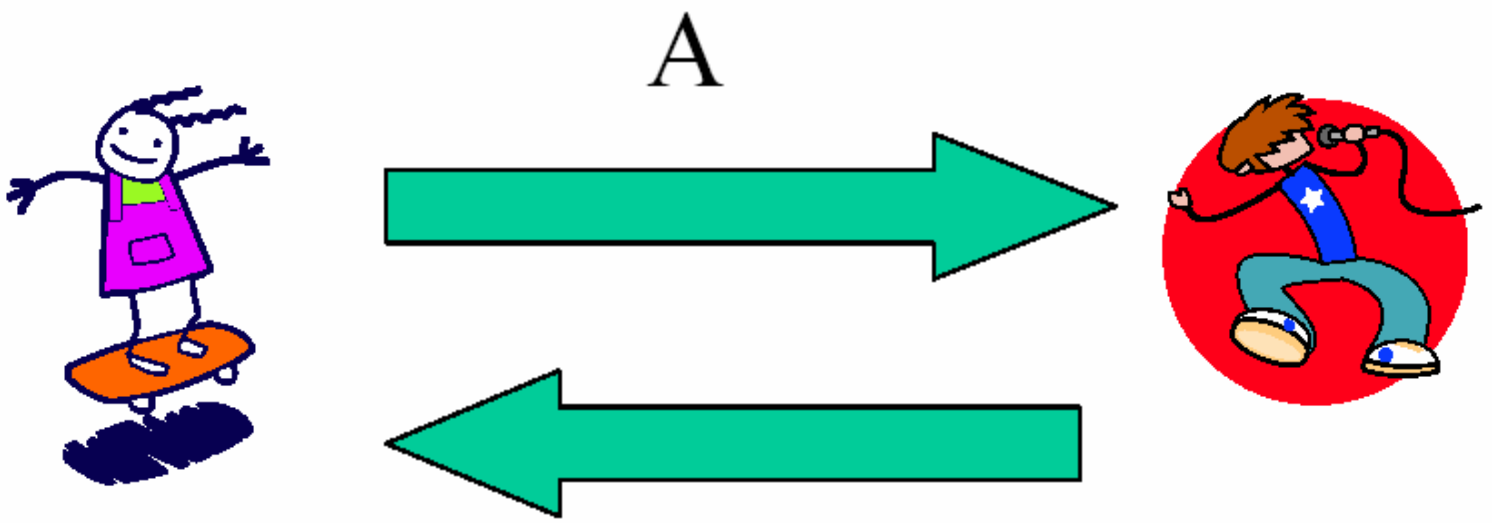


A=010000111101……

B=010010111101….

Communication Complexity = n bits (too high)

Verifying set equality



A=010000111101……
(n bits)

Sketch(A) – log n bits

B=010010111101….
(n bits)

Sketch(B) – log n bits

Fingerprinting

- Fewer bits transmitted by sending sketch instead of the whole database
- Small Probability of error
  - If the databases are equal, then always return "equal"
  - If unequal, then may return "equal" with some probability

Fingerprinting

Given bit string

$$A = a_1 a_2 .... a_n$$

Interpret as integer

$$a = a_1 .1 + a_2 .2 + ... a_n .2^{n-1}$$

Can't transmit a itself, since it is an n-bit integer

Fingerprint $F(a,q) = (a \bmod q)$ for some prime q

Fingerprinting Computation

Given "small" integer q easy to compute
F(a,q)

$$F(a,q) = (a_1.1 + a_2.2 + ...a_n.2^{n-1}) \bmod q$$

Use Horner's Multiplication rule to evaluate
F(a,q) in O(n) operations in (modulo q)

Protocol

Alice: Input a

Bob: Input b

1. Alice decides on prime q informs Bob
2. Alice computes F(a,q), sends to Bob
3. Bob computes F(b,q), compares with F(a,q)
4. If fingerprints unequal, then a and b unequal
5. If fingerprints equal, declare a and b to be equal

Prime Number q

- If Adversary knows q, then can easily construct a, b such that $a \neq b$ but $(a \bmod q) = (b \bmod q)$

- Choose q randomly
  - What is the probability that $(a \neq b)$ and $(a \bmod q) = (b \bmod q)$
  - We aren't ready to answer this yet!

False Positive

$a \neq b$

$(a \bmod q) = (b \bmod q)$

$(a-b) \bmod q = 0$

$q \mid a-b$

Since $a, b < 2^n$, $(a-b) < 2^n$

How many prime factors does (a-b) have?

Prime Divisors

Theorem: If c < 2^n, then c has less than n distinct prime factors

Proof by Contradiction:

Suppose c has m $\geq$ n prime factors

Each prime factor $\geq$ 2

c $\geq$ 2^n

Thus, there are no more than n "bad choices" for q

Density of Primes

For number k, let $p(k)$ = number of distinct primes less than or equal to k

Prime Number Theorem:

For large k, $p(k) \approx k/(\ln k)$

Sample Space

1. Choose $s = (tn\ (\log tn))$ for some parameter t

   Number of primes less than $s \approx s/\log s \approx tn$

2. Choose q = a random prime number in the range [1..s]

Probability of a bad prime

1. Number of "Bad" choices for $q < n$
2. Size of Sample Space = $t.n$
3. Probability that random q is bad = $O(1/t)$

4. Number of bits transmitted = $log(s)$
   $= O(log\ t + log\ n)$

5. Works well if $(t=n)$

Final Protocol Properties

- Compare equality of two sets of n bits each by transmitting only $O(\log n)$ bits
- Probability of false positive = $O(1/n)$
- Unresolved Issue: How to choose a random prime in the range $[0..n^2]$?

String Matching

Text is a long string of length n

$$T = x_1 x_2 \ldots . x_n$$

Pattern is a shorter string of length m

$$P = y_1 y_2 \ldots . y_m$$

Problem: Find all occurrences of P in T

String Matching

Many applications

–While using editor/word processor/browser

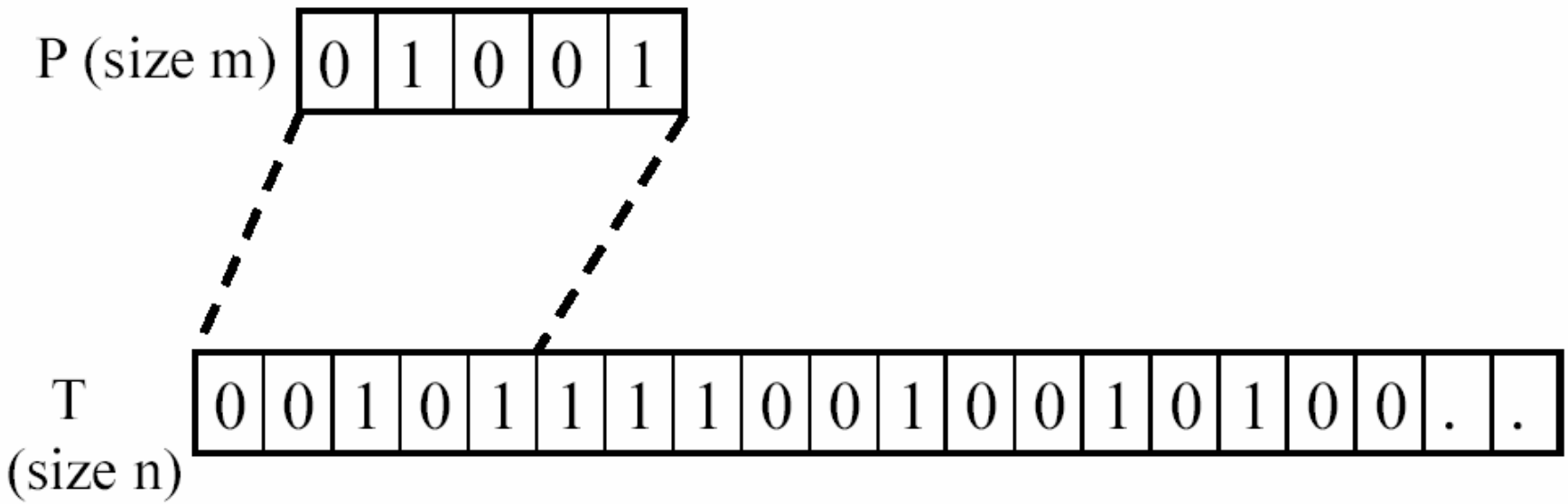
–Login name & password checking

–Virus detection

–Header analysis in data communications

–DNA sequence analysis

Naïve O(nm) algorithm



Compare P with (n-m) substrings of T, each comparison takes O(m) time

Rabin-Karp Algorithm

- O(m+n) run time, Monte Carlo algorithm
- O(m+n) Deterministic algorithms exist (Knuth-Morris-Pratt)
- Randomized Algorithm still useful
  - Very Simple
  - Can be used to detect patterns in multi-dimensional data

Fingerprinting

High Level:

1. Compute Fingerprint of P
2. Compute Fingerprints of all length m substrings of T
3. Lookup all matches in fingerprints (easy)
    1. Monte Carlo: return right away
    2. Las Vegas: Confirm fingerprint matches by actual text matches

Fingerprinting function

Interpret binary string Z of length k as a binary integer

$F(Z,q) = (Z \bmod q)$ for prime q

- Pattern P, $F(P,q) = (P \bmod q)$
- Text T
  For each $1 \leq i \leq (n-m+1)$, define $T(i) = T[i \ldots i+m-1]$
- Let Array A store the fingerprints of the T(i)'s

```
For (i=1 to n-m+1)
    A[i] = F(T(i),q)
```

Fingerprinting computation

Whole array A can be computed in O(n) operations

$$A[i+1] = F(T(i+1), q) = T(i+1) \bmod q$$

$$A[i] = F(T(i), q) = T(i) \bmod q$$

$$T(i+1) = (T(i) - 2^{m-1} T[i]) \cdot 2 + T[i+m]$$

The only expensive operation

Given A[i], A[i+1] can be computed in O(1) field operations

False Positives?

- Similar to previous analysis
- What is
  $Pr\ [F(T(i),q) = F(P,q) \mid T(i) \neq P]$?

  $\Pr[q \mid T(i)\text{-}P]$

Since $T(i) < 2^m$ it has less than m prime factors

No more than m "Bad" Primes

Sample Space

1. Choose $s = (tm\ (log\ tm))$ for some parameter t

   Number of primes less than $s \approx s/log\ s \approx tm$

2. Choose q = a random prime number in the range [1..s]

3. By our analysis, probability of False Positive = $O(1/t)$

False Positives

We want a high probability that every match is accurate

- Choose $s = (n^2 m (\log n^2 m))$
- Probability of error *on a particular instance* $= O(1/n^2)$
- No more than n matches
- By Union Bound,
  Probability of error *on some instance* $= O(1/n)$

Fingerprinting

- Find all matches of Pattern P in Text T in time $O(|P|+|T|)$
- Probability of False Positive = $O(1/|T|)$