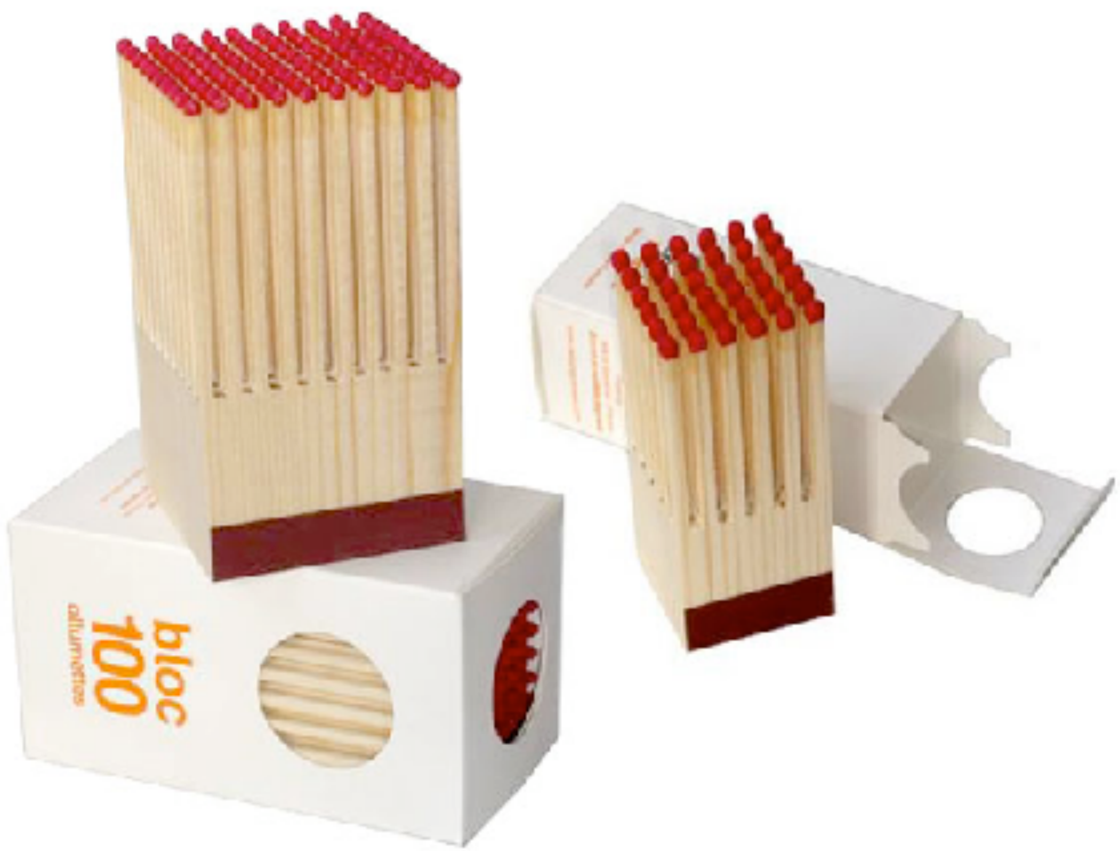
L23

4800

12.2.2016

abhi shelat



http://kitsunenoir.com/blogimages/bloc-matches.jpg

check procedure:

check procedure:

randomly pick 50 matches and light them

if one fails, reject the box.

if all succeed, accept it.

Pr that test fails

three cases to consider:

Pr of failure for n=100

9.911653021418339067376749 69
6883601495412210270643283767
8927852568890730299973273935
87632943101698342E-30

0.0000000000000000000000000000099

9.91165302141833906737674969
6883601495412210270643283767
8927852568890730299973273935
87632943101698342E-30

0.0000000000000000000000000000099

9.91165302141833906737674969
6883601495412210270643283767
8927852568890730299973273935
87632943101698342E-30

0.0000000000000000000000000000099

pr of royal flush:

1.53908E-6

pr that you...

Age in 1990	Total U.S.	White Male	White Female	Black Male	Black Female
20	0.102%	0.128%	0.045%	0.307%	0.074%

Using random coins can help overcome adversarial behavior

Using random coins can also simplify an algorithm

Fingerprinting

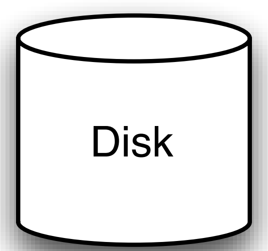
Fingerprinting

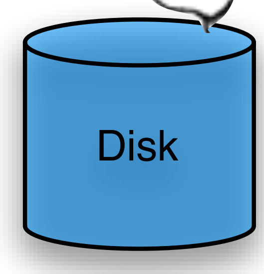
Alice

pick prime p

Bob

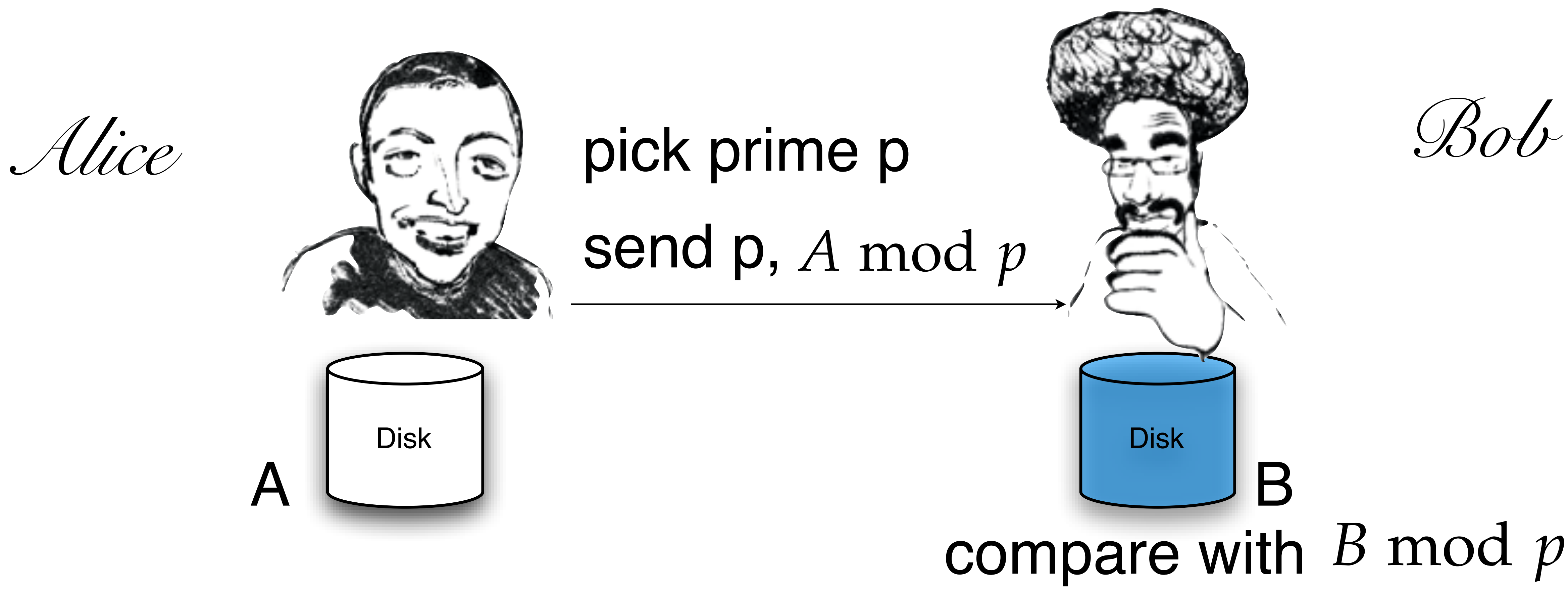
A





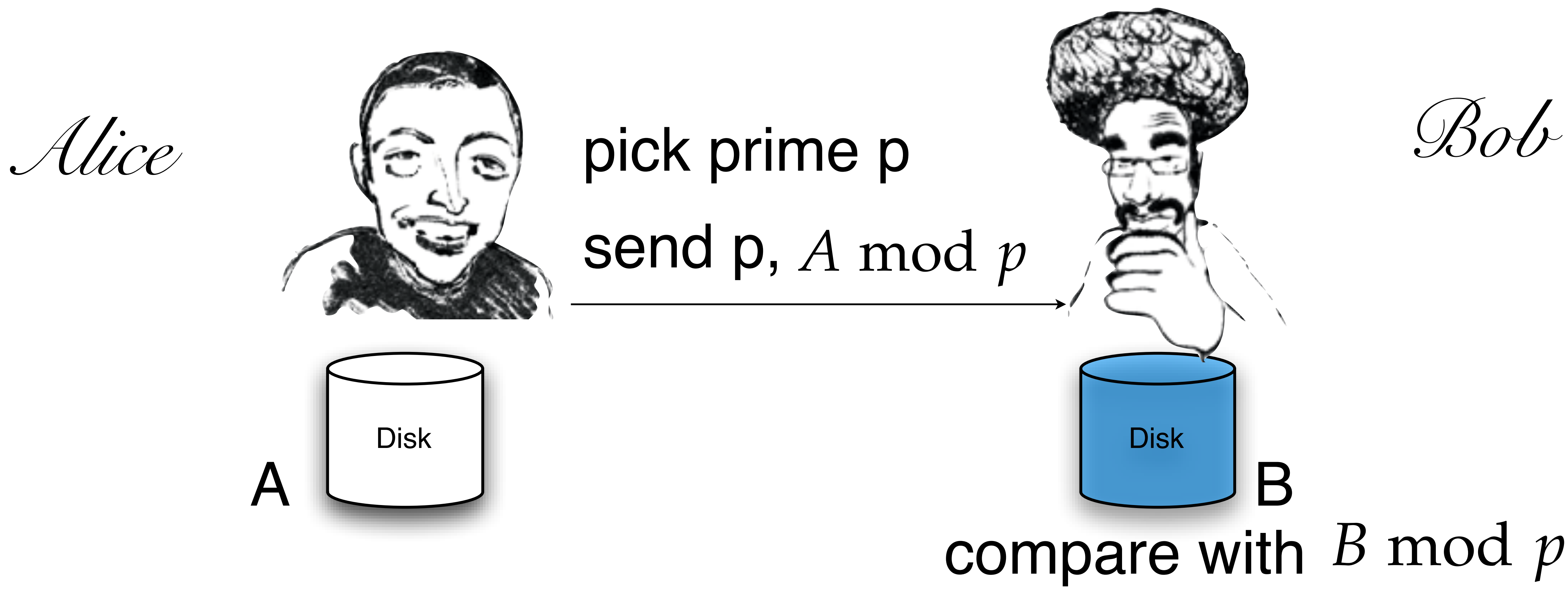
B

Fingerprinting



if A=B, then

Fingerprinting

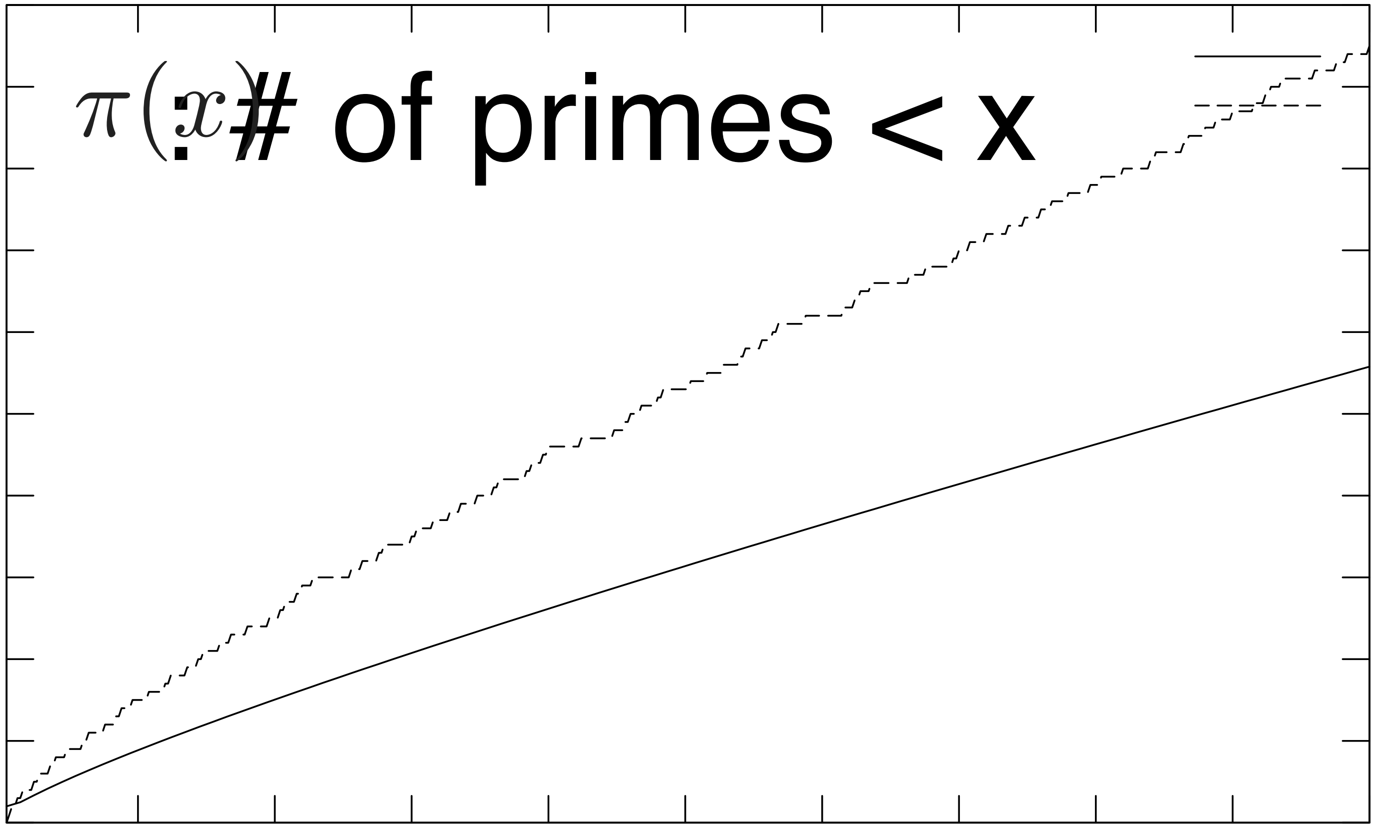


if A≠B, then

number of primes

number of primes

there are certainly infinitely many



lemma:

\# of prime divisors of $x < \log(x)$

Easy to pick primes

```
import java.io.*;
import java.math.*;
import java.util.*;

public class pr {
   public static void main(String args[]) {

      BigInteger prime = new BigInteger(128,80,new Random());
      System.out.println("prime is " +prime);

   }
}
```

```
abhis-MacBook-Pro:hw abhi$ java pr
prime is 194320298558336431416620955357714454897
abhis-MacBook-Pro:hw abhi$ java pr
prime is 250932337219632799561119530768795821559
abhis-MacBook-Pro:hw abhi$ java pr
prime is 208446315596042010374903390602426953283
abhis-MacBook-Pro:hw abhi$ java pr
prime is 277692390735250370111358788148532452689
abhis-MacBook-Pro:hw abhi$ java pr
prime is 178745644948876658400223198257146073499
```

pr of false match:

example params

Alice

randomly pick 64bit prime p

send p, h_A=A mod p →

Bob

A

Disk

Disk

B

Compute $h_B \leftarrow$ B mod p

If $h_A = h_B$ Output EQUAL

string matching

pattern

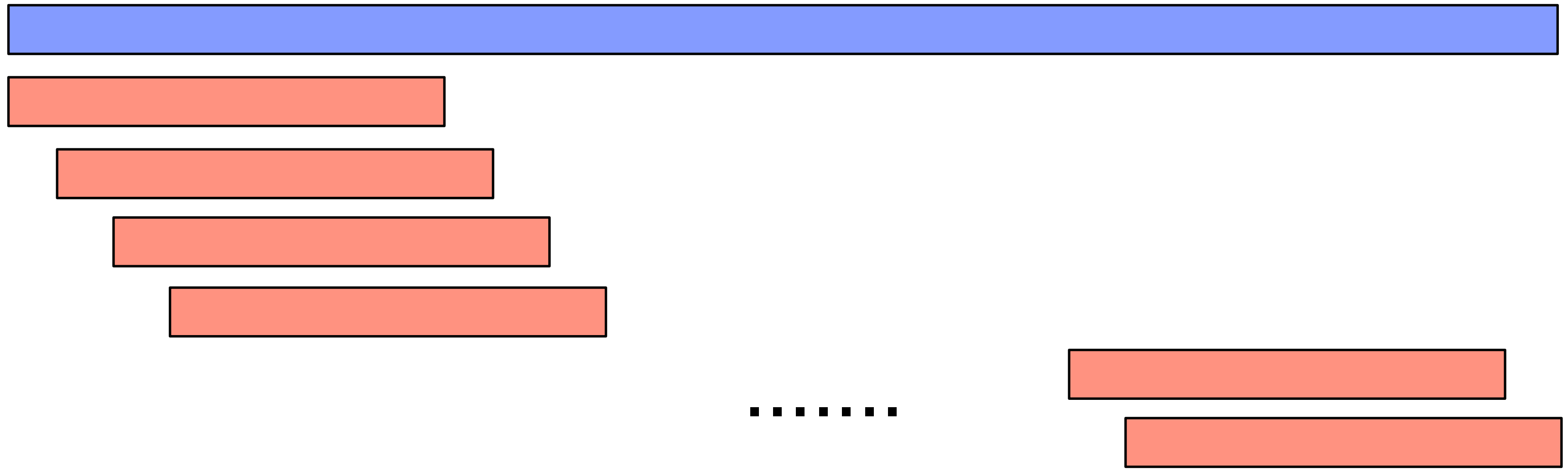
corpus

```
A squabble between a group fighting spam and a
Dutch company that hosts Web sites said to be
sending spam has escalated into one of the
largest computer attacks on the Internet,
causing widespread congestion and jamming
crucial infrastructure around the world.
Millions of ordinary Internet users have
experienced delays in services like Netflix or
could not reach a particular Web site for a
short time.
```

string matching

string matching

brute force:




```
for (int i = 0, j=0; i < n-m; i++) {
  while (j < m && t[i+j] == p[j]) { j++; }
  if (j == m) return i;
}
return -1;
```

simple algorithm

aaaaaaaaaaaaaaaaa
aaaaaab

brute force worst case:

simple algorithm

aaaaaaaaaaaaaaaaa

aaaaaab

aaaaaab

brute force worst case:

simple algorithm

aaaaaaaaaaaaaaaaa

aaaaaab

aaaaaab

aaaaaab

brute force worst case:

KMP algorithm

abcdabcdabcdefh

abcdabhi

KMP algorithm

abcdabcdabcdabcdefh

abcdabhi

KMP sliding rule

given that P[1....q] matches T[j...j+q],
but a mismatch occurs at j+q+1, then:

KMP sliding rule

given that P[1....q] matches T[j...j+q],
but a mismatch occurs at j+q+1, then:

find the longest prefix P[1…i] of P[1...q]
that is also a suffix of P[1...q]

abcdabhi

slide (q-i) so that P[1...i] matches T[j+(q-i),…]

0	1	2	3	4	5	6	7	8	9	10
x	y	x	y	y	x	y	x	y	x	x

0	1	2	3	4	5	6	7	8	9	10
x	y	x	y	y	x	y	x	y	x	x
0	0	1	2	0	1	2	3	4	3	1

new idea for string match

string matching

```
pick random t-bit prime

compute h = pattern mod prime

for i=1...n
    compute hi = next corpus ci mod prime
     if hi == h, output match
```

pick an 128-bit prime p

What is the probability of a false match at the first position?

pr of any mismatch:

string matching example

pattern

26535

Text

31415926535897931 2

string matching example

pattern

2 6 5 3 5

Text

3 1 4 1 5 9 2 6 5 3 5 8 9 7 9 3 1 2

Given that 31415 mod 17 = 16,
How can I compute 14159 mod 17?

Hint: 10000 mod 17 = 4

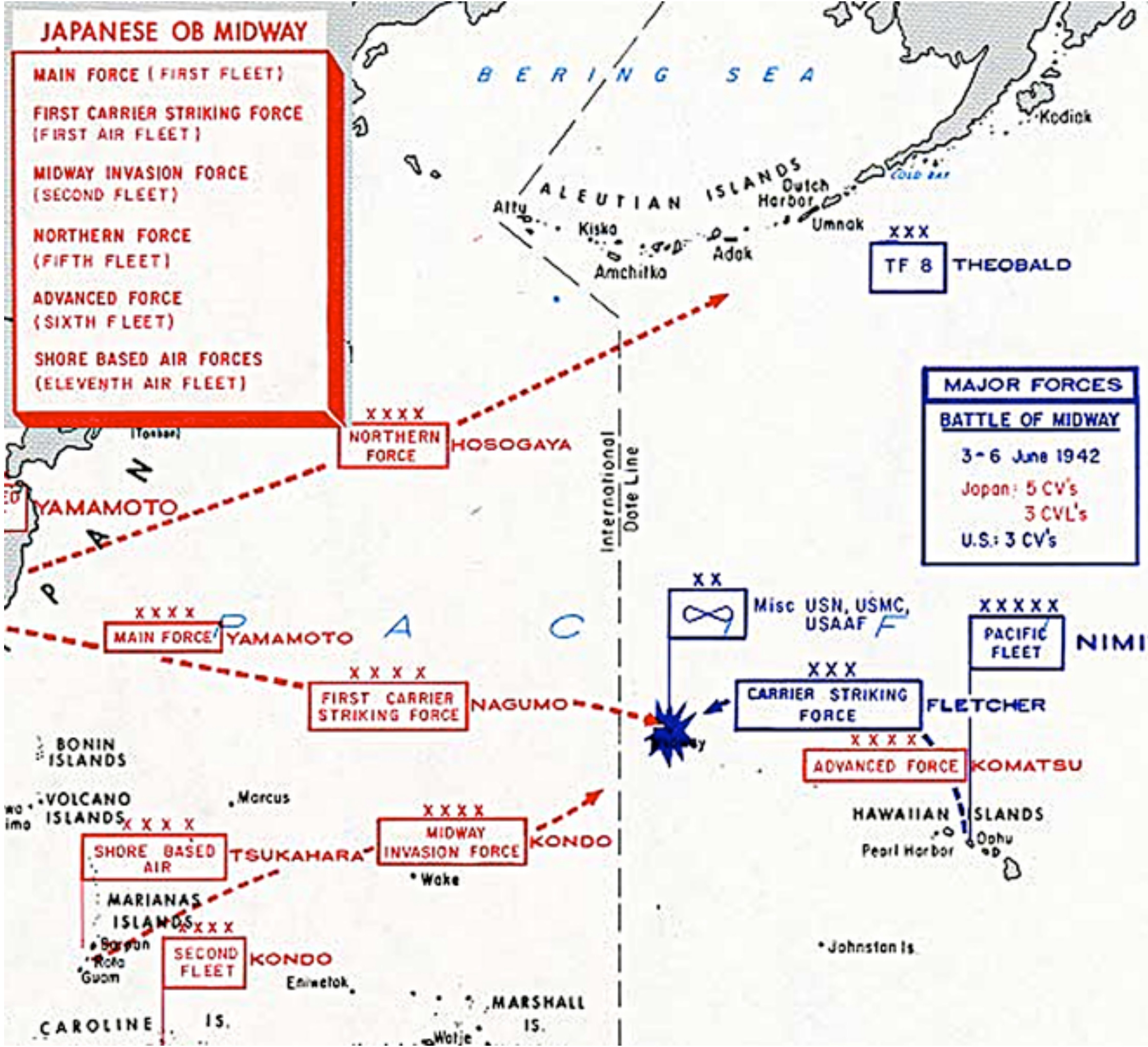
```
public static int search(String p, String t) {
   int M = p.length();
   int N = t.length();
   int dM = 1, h1 = 0, h2 = 0;
   int q = pickRandomPrime();
   int d = 256; // radix
   for (int j = 1; j < M; j++) // precompute d^M % q
      dM = (d * dM) % q;

   for (int j = 0; j < M; j++) {
      h1 = (h1*d + p.charAt(j)) % q; // hash of pattern
      h2 = (h2*d + t.charAt(j)) % q; // hash of text
   }
   if (h1 == h2) return i - M; // match found

   for (int i = M; j < N; i++) {
      h2 = (h2 - t.charAt(i-M)*dM) % q; // remove high order digit
      h2 = (h2*d + t.charAt(i)) % q; // insert low order digit
      if (h1 == h2) return i - M; // match found
   }
   return -1; // not found
}
```

june 1942

jn-25b

CMDR EDWARD T LAYTON
(FLEET INTELLIGENCE OFFICER)

LT CMDR JOSEPH ROCHEFORT
(COMBAT INTELLIGENCE UNIT)



JAPANESE OB MIDWAY
MAIN FORCE (FIRST FLEET)
FIRST CARRIER STRIKING FORCE (FIRST AIR FLEET)
MIDWAY INVASION FORCE (SECOND FLEET)
NORTHERN FORCE (FIFTH FLEET)
ADVANCED FORCE (SIXTH FLEET)
SHORE BASED AIR FORCES (ELEVENTH AIR FLEET)
BERING SEA
ALEUTIAN ISLANDS
Attu
Kiska
Amchitka
Adak
Dutch Harbor
Umnak
COLD BAY
Kodiak
XXX
TF 8
THEOBALD
MAJOR FORCES
BATTLE OF MIDWAY
3-6 June 1942
Japan: 5 CV's
3 CVL's
U.S.: 3 CV's
XXXX
NORTHERN FORCE
HOSOGAYA
YAMAMOTO
International Date Line
XX
Misc USN, USMC, USAAF
XXXXX
PACIFIC FLEET
NIMIT
XXXX
MAIN FORCE
YAMAMOTO
P A C I F
XXXX
FIRST CARRIER STRIKING FORCE
NAGUMO
XXX
CARRIER STRIKING FORCE
FLETCHER
XXXX
ADVANCED FORCE
KOMATSU
BONIN ISLANDS
VOLCANO ISLANDS
Marcus
HAWAIIAN ISLANDS
Oahu
Pearl Harbor
XXXX
SHORE BASED AIR
TSUKAHARA
XXXX
MIDWAY INVASION FORCE
KONDO
Wake
MARIANAS ISLANDS
Saipan
Rota
Guam
XXXX
SECOND FLEET
KONDO
Eniwetok
MARSHALL IS.
Johnston Is.
CAROLINE IS.
Wotje

MOD-EXP

$$(a, x, n) \rightarrow a^x \bmod n$$

MOD-EXP

$$(a, x, n) \to a^x \bmod n$$

$$a^x \bmod n = \prod_{i=0}^{\ell} x_i a^{2^i} \bmod n$$

1

MOD-EXP

$$(a, x, n) \rightarrow a^x \bmod n$$

Algorithm 2: ModularExponentiation(a, x, n)

Input: $a, x \in [1, n]$

```
r ← 1
while x > 0 do
    if x is odd then
        r ← r · a mod n
    x ← ⌊x/2⌋
    a ← a² mod n
Return r
```

1

MOD-EXP

$$(a, x, n) \to a^x \bmod n$$

$$a^x \bmod n = \prod_{i=0}^{\ell} x_i a^{2^i} \bmod n$$

Algorithm 2: ModularExponentiation(a, x, n)

Input: $a, x \in [1, n]$

$r \leftarrow 1$

while $x > 0$ **do**

  if x *is odd* **then**

    $r \leftarrow r \cdot a \bmod n$

  $x \leftarrow \lfloor x/2 \rfloor$

  $a \leftarrow a^2 \bmod n$

Return r

El Gamal Encryption

Gen:

Enc(PK,m):

Dec:

El Gamal Encryption

Gen: Pick random x. Output PK=g^x, SK=x.

Enc(PK,m)

Dec(c_1,c_2,SK)

El Gamal Encryption

Gen: Pick random x. Output $PK=g^x$, $SK=x$.

$Enc(PK,m)$ Pick random r. Output $(g^r, g^{rx} * m)$

$Dec(c_1,c_2,SK)$

Why is it secure?

Let (a,b,c) be random exponents chosen from [1,p-1]

$$(g^a, g^b, g^{ab})$$

$$(g^a, g^b, g^c)$$

prime is 23129630111058764318553907663148788693 3

1387498068869719542583902570469619096 53
3145275507192679957128023392767428157 2

1337363740564509032891199806994005198 18

1837239243879414762677311698612805397 51 prime is 3258066275885504310109470353800067921411

263788312045705026395665799012729562167
232351424716312897042950264984304468335

93298786459176480146160445046926050732

194375326202773113445261188688424185897