Number Theory & Cryptography

Note Title 12/2/2013

- Oral grading tomorrow
  email if you need a spot
-

Algorithmic Number Theory

Increasingly of vital importance in computer science.

Why?

crypto

Some Facts & Definitions:

- Given a & b, $a|b$ means "a divides b"

$$\Rightarrow \exists k \in \mathbb{Z} \text{ s.t. } a \cdot k = b$$

Thm: Consider $a, b, c \in \mathbb{Z}$.

1) If $a|b$ & $b|c$, then $a|c$.
2) If $a|b$ & $a|c$, then $a|(ib + jc)$ for all $i, j \in \mathbb{Z}$
3) If $a|b$ and $b|a$, then $a = b$ or $a = -b$.

A number p is **prime** if its only divisors are 1 and p

(so $d \mid p \Rightarrow d = 1$ or $d = p$.)

If not prime, a number is **composite**.

Fund. Thm of Arithmetic

Take $n > 1$ an integer.
$\exists$ unique sets $\{p_1, \dots, p_k\}$ and $\{e_1, \dots, e_k\}$ such that

$$n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$$

prime factorization

Greatest Common Divisor

$\gcd(a,b)$ = largest number dividing both a & b.

If $\gcd(a,b) = 1$, a & b are relatively prime.

Aside: When is relatively prime useful?

key exchanges

hashing

GCD algorithm

<u>Key lemma:</u> Let a & b be 2 positive integers. For any $r \in \mathbb{Z}$,

$$\gcd(a,b) = \gcd(b, a-rb)$$

<u>pf:</u> $d = \gcd(a,b)$ & $c = \gcd(b, a-rb)$

<u>Goal:</u> Show $d \le c$ and $c \le d$.

1. d divides a & b.
   so d divides $a-rb$ (by ii of prev theorem)
   so $d \mid b$ and $d \mid a-rb$
   $\Rightarrow$ $d \le c$

pf cont

② know $c \mid b$ and $c \mid a - rb$

$$\frac{a-rb}{c} = \frac{a}{c} - \underbrace{\frac{rb}{c}}_{\text{this is an integer, } c \mid b}$$

so $\frac{a}{c}$ must be an integer

$\Rightarrow c \mid a$ (and knew $c \mid b$)

$c \le d$ since d is $\gcd(a, b)$

$\Rightarrow c \le d$ & $d \le c \Rightarrow d = c$ $\boxtimes$

Modulo:

if $r = \underbrace{a \bmod n}$

(so $r \in \{1 .. n\}$)

then $\exists q$ s.t. $a = qn + r$

so $r = \underbrace{a - qn}$

Looks like our last Lemma...

Euclid's algorithm

EuclidGCD(a, b):

If b = 0
    return a
else
    EuclidGCD(b, $\underbrace{a \bmod b}_{= \, a - q \cdot b}$)

Correctness: See lemma!

Runtime: How many recursive calls?

Note: EuclidGCD(a, b)

↳ EuclidGCD(b, a mod b)

- first input always bigger
- first input is decreasing

Let a_i = first input of i^{th} recursive call.

Have: $a_{i+2} = a_i \bmod a_{i+1}$

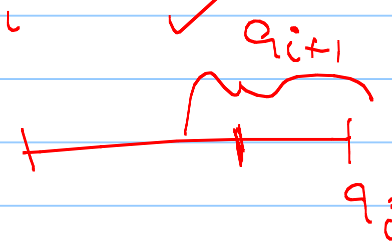
Claim: for all $i > 2$, $a_{i+2} < \frac{1}{2} a_i$

pf: 2 cases

Case 1: if $a_{i+1} \leq \frac{1}{2} \cdot a_i$

then $a_{i+2} < a_{i+1} \leq \frac{1}{2} a_i$ ✓

Case 2: if $a_{i+1} > \frac{1}{2} \cdot a_i$



then $a_{i+2} = a_i \bmod a_{i+1}$

$\Rightarrow a_{i+2} = a_i - a_{i+1} < \frac{1}{2} a_i$ ∎

Conclusion: Every 2 iterations, input goes down by $\frac{1}{2}$.

$\Rightarrow$ $O(\log n)$ iterations

Modular Arithmetic

Let $\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$

(often called residues mod n)

This is a group we often work in:

- finite
- has associativity, commutativity, identity elements, etc.

$a+b = b+a$

Inverses: additive versus multiplicative

Additive Inverses:

Additive identity is 0:

$(x+0) \bmod n = x$

Additive inverse: always exists!

For each $x \in \mathbb{Z}_n$, can find $y \in \mathbb{Z}_n$ s.t. $x+y=0$

<u>Ex</u>: 5 mod 11 $\leftarrow \mathbb{Z}_{11} = \{0, \ldots, 10\}$

inverse: $5 + \boxed{6} = 0$

Multiplicative Inverses

Given $z \in \mathbb{Z}_n$, multiplicative inverse z^{-1} is a number s.t.

$$z \cdot z^{-1} \bmod n = 1.$$

Ex: 5 modulo 9 $\{0, \ldots, 8\} = \mathbb{Z}_9$

inverse? 2 $\Rightarrow 2 \cdot 5 = 10 \bmod 9 = 1$

Ex: 3 modulo 9

inverse? none

Thm: An element $x \in \mathbb{Z}_n$ has an inverse in $\mathbb{Z}_n$
$\iff \gcd(x, n) = 1$.

$\gcd(5, 9) = 1$

$\gcd(3, 9) = 3$

$\phi(n)$ counts the number of elements of $\mathbb{Z}_n$ which have an inverse

Side note: Why do we care?

1. Why not use $\mathbb{R}$?

   uncountable, error

2. Why not use $\mathbb{Z}$?

   infinite (vs $\mathbb{Z}_n$)

3. Why good for cryptography?

AES

Example: Advanced Encryption Standard

History:

- In 1996, NIST issued a call to replace 3DES.
- In 1998, 15 algorithms were submitted.
- NIST spent years having open tests done on all submissions.
- The winner was Rijndael, developed by 2 Belgian cryptographers.
- Officially approved in 2001.

How AES works:

- All computation done over $\mathbb{Z}_{256}$.

Essentially, 4 operations: (performed repeatedly)

1) Substitute bytes
2) Permute
3) Mix Columns
4) Add round key (an XOR with part of secret key - changes each round)

Note: This type of symmetric encryption requires a secret, shared key.

Algorithmically, fairly uninteresting, although highly useful & secure

Runtime: Generally linear in length of the message.

More interesting: How to get a secret key?

Public Key cryptosystem:

Encryption scheme E, decryption D

Goals (Diffie-Hellman '76)

1. $D(E(M)) = M$ & $E(D(M)) = M$ ← inverse
2. Both E & D are easy to compute.
3. Given E, "hard" to derive D.

Caution: Generally, D & E are linked to some hard problem.

One early system, the Merkle-Hellman, was based on the knapsack problem (known to be NP-Hard).

However, the problems turned out to be an easy-to-solve subclass of knapsack, so it was found vulnerable to attack.

RSA: Rivest, Shamir & Adleman

- Tied to factoring large numbers.

Need a bit more number theory:

Euler's totient function $\phi(n) =$
# of positive integers $\leq n$
that are relatively prime to n.

If p is prime:

$$\phi(p) = p-1$$

$$1 \ldots p-1$$

$\phi(n)$ when n is not prime:

If $n = pq$ (both prime), interesting special case.

pq possible numbers

↳ q of them are multiples of p ($p, 2p, 3p, \ldots, qp$)

& p are multiples of q. ($q, 2q, \ldots, pq$)

so $\phi(pq) = pq - q - (p-1)$

$= (p-1)(q-1)$

Why we care:

Closely tied to the set of numbers which have a multiplicative inverse in $\mathbb{Z}_n$!

Euler's thm: n a positive integer, & $x \in \mathbb{Z}_n$ s.t. $\gcd(x, n) = 1$.

Then $x^{\phi(n)} \equiv 1 \mod n$

$x^{\phi(n)}$ is x's inverse