Fermat's Little Theorem

Recall: Simplification problems.

$$11^{1002} \equiv (-1)^{1002} = 1 \pmod{12}$$

simplify base

Saw we cannot simplify exponent:

$$2^5 \not\equiv 2^1 \pmod 4$$

But: There is a version of exponent simplification.

Fermat's Little Theorem: If p is prime and $p \nmid a$ then $a^{p-1} \equiv 1 \pmod p$.

Ex: • $5^{20} = (5^2)^{10} \equiv 1 \pmod{11}$

• $3^{14} = 3^2 \cdot (3^2)^6 \equiv 3^2 \equiv 2 \pmod 7$

Lemma: If p and a are coprime, the function $f: \{0, 1, \dots, p-1\} \to \{0, 1, \dots, p-1\}$ given by $f(x) = ax \bmod p$ is bijective.

Ex: $p=5$, $a=2$ $\quad f(x) = 2x \mod 5$

$f(0) = 0 \qquad f(3) = 1$

$f(1) = 2 \qquad f(4) = 3$

$f(2) = 4$

Proof: $\gcd(a,p)=1$, so a has an inverse, a^{-1}. Consider $g: \{0,1,\dots,p-1\} \to \{0,1,\dots,p-1\}$ given by $g(x) = a^{-1}x \mod p$. We notice $g(f(x)) = x$ and $f(g(x)) = x$, so $g = f^{-1}$. Since f is invertible, it is bijective. $\square$

Proof of FLT: Say p is prime, $p \nmid a$. Then, by lemma, $f: \{0,1,\dots,p-1\} \to \{0,1,\dots,p-1\}$ given by $f(x) = ax \mod p$ is bijective. So,

$$\{0,1,2,\dots,p-1\} = \{f(0), f(1), f(2), \dots, f(p-1)\}$$

But we always have $f(0) = 0$, so

$$① \quad \{1,2,\dots,p-1\} = \{f(1), f(2), \dots, f(p-1)\}.$$

$$= \{a \mod p, 2a \mod p, \dots, (p-1)a \mod p\} \quad ②$$

So, the product of elements in 1 must equal the product of elements in 2.

$$\Rightarrow 1\cdot 2 \cdots (p-1) = (a \bmod p) \cdots ((p-1)a \bmod p)$$

$$\Rightarrow 1\cdot 2 \cdots (p-1) \equiv a \cdot 2a \cdots (p-1)a \pmod{p}$$

$$\Rightarrow 1\cdot 2 \cdots (p-1) \equiv 1\cdot 2 \cdots (p-1)\cdot a^{p-1} \pmod{p}$$

Since p is prime, all numbers $1, 2, \ldots, p-1$ are coprime to p, so they have inverses mod p. So, we can cancel them!

$$\Rightarrow 1 \equiv a^{p-1} \pmod{p}. \qquad \square$$

Alternate FLT: If p is prime, $a^p \equiv a \pmod{p}$ for any a.

Proof: If $p \nmid a$ then $a^p = a \cdot a^{p-1} \overset{\text{FLT}}{\equiv} a \pmod{p}$.

If $p \mid a$, $a \equiv 0 \pmod{p}$. $\square$

This version can be nice in proofs since it works for any a.

Now, a big application!

Encryption

Setup: Alice wants to send a message to Bob.

We can assume the message is an integer $x > 0$.

Why? text $\longrightarrow$ binary $\longrightarrow$ integer

hi $\xmapsto{\text{ASCII}}$ 01101000 01101001 $\longmapsto$ 26724

Challenge: Eve reads all communication Alice to Bob. Want Bob to be able to conclude the message was x, but we can't let Eve do so!

General Idea: Pick a big N $(x < N)$.

Encryption function: $E: \{0, \dots, N-1\} \to \{0, \dots, N-1\}$

Decryption function: $D: \{0, \dots, N-1\} \to \{0, \dots, N-1\}$

Alice calculates $E(x)$, sends it to Bob.

Bob will use D to recover x.

2 Desired Properties:

1. $D(E(x)) = x$ → Bob can recover x

② Eve cannot find x from $E(x)$, even if she knows how E works.

Two encryption types

1. Private-key encryption: Alice and Bob meet up beforehand, share secret info that they use to compute E and D.

   Downside: Bob must meet with everyone who wants to send him messages!

   See CS 161 for examples.

2. Public-key encryption: Bob publicly broadcasts information that lets anyone find E, but keeps a secret so only he can find D / undo E.

Seems impossible: If Eve has enough info to find E herself, surely she can find x from $E(x)$...

Turns out it is possible!

RSA Encryption

Rivest-Shamir-Adleman (1977)

- Bob's Setup:

  a) Bob picks large primes p and q
  (in practice, 300-600 digits)

  b) Bob calculates $N = p \cdot q$

  c) Bob picks e so $\gcd(e, (p-1)(q-1)) = 1$.
  (in practice, e usually small, like 3)

  d) Bob computes $d = e^{-1} \bmod (p-1)(q-1)$.

  e) Bob publicly shares (N, e) "public key"
  Bob keeps d secret "private key"

- Encryption: Alice wants to send x to Bob

  a) Alice makes sure $x < N$

  b) Alice sends $E(x) = x^e \bmod N$

- Bob wants to read encrypted message y
  Bob calculates $D(y) = y^d \bmod N$

Ex: RSA with small numbers

Setup: a) Take $p=3$, $q=11$

b) Get $N=33$

c) Pick e. $(p-1)(q-1)=20$.

So, $e=3$ works since $\gcd(3,20)=1$.

d) Compute $d = e^{-1} \bmod (p-1)(q-1)$

$= 3^{-1} \bmod 20 = 7$

e) Public Key: $(N, e) = (33, 3)$

Private Key: $d=7$

Encrypt 29 : $E(29) = 29^e \bmod N$

$= 29^3 \bmod 33$

$= (-4)^3 \bmod 33 = 2$

Decrypt 2 : $D(2) = 2^d \bmod N$

$= 2^7 \bmod 33$

$= 128 \bmod 33 = 29$

It worked! Let's prove it always does.

Theorem: If $0<x<N$, then $D(E(x))=x$.

Proof: By definition, $D(E(x)) = (x^e \bmod N)^d \bmod N$.

So, $0 < D(E(x)) < N$. Since $0<x<N$, we just need to show $D(E(x)) \equiv x \pmod{N}$.

We can simplify after multiplying, so we just need to show $(x^e)^d \equiv x \pmod{N}$.

We know $d \equiv e^{-1} \pmod{(p-1)(q-1)}$, so

$ed \equiv 1 \pmod{(p-1)(q-1)}$.

$\Rightarrow ed - 1 = k(p-1)(q-1)$, $k \in \mathbb{Z}$

$\Rightarrow (x^e)^d = x^{ed} = x^{k(p-1)(q-1)+1}$

Also, $N = pq$, so need to show

$$x^{k(p-1)(q-1)+1} \equiv x \pmod{pq}$$

First, show $x^{k(p-1)(q-1)+1} \equiv x \pmod{p}$

If $p \mid x$, both sides are 0.

If $p \nmid x$, use FLT.

$$x^{k(p-1)(q-1)+1} = x \cdot (x^{k(q-1)})^{p-1} \equiv x \pmod{p}$$

So, $x^{k(p-1)(q-1)+1} \equiv x \pmod{p}$

Similarly, $x^{k(p-1)(q-1)+1} \equiv x \pmod{q}$

Consider the system in terms of variable n

$$\begin{cases} n \equiv x^{k(p-1)(q-1)+1} \pmod{p} \\ n \equiv x^{k(p-1)(q-1)+1} \pmod{q} \end{cases}$$

It has solutions $n = x^{k(p-1)(q-1)+1}$, $n = x$

But $\gcd(p,q) = 1$, so by CRT uniqueness,

$x^{k(p-1)(q-1)+1} \equiv x \pmod{pq}$. $\square$

So, encryption-decryption always works.

Next Q: Why can't Eve find x from $E(x)$?

Two sensible strategies for Eve:

1. Eve tries to find the private key d.
   - If Eve finds d and knows $y = E(x)$, she can do $D(y) = y^d \bmod N$.
   - To find d, Eve must compute $e^{-1} \bmod (p-1)(q-1)$. Eve knows e,

and knows $N = p \cdot q$, but to find $(p-1)(q-1)$ she must know p and q individually.

Security Point #1: When N is huge (300-600 digits) it cannot be easily factored (no fast algorithms!)

Even the world's fastest computers cannot do it in a human lifetime.

But: There is no proof factoring can't be fast! Just no one has found a way...

2. Eve could directly try to compute x from $E(x) = x^e \bmod N$.

eth root is fast: $\sqrt[e]{E(x)} = x$? No!

$E(x)$ finds x^e then takes remainder $\bmod N$.

Security Point #2: When x is huge there is no fast way to undo $x^e \bmod N$.

Note: If $x^e < N$, the $\bmod N$ does nothing and Eve can get x by taking $\sqrt[e]{E(x)}$.

Moral: RSA does work in practice, but it is brittle. If anything (even something subtle) is slightly wrong, it can break!

Final Q: Can Alice and Bob do things fast?

Mod powers: Repeated squares

Calculate d: Extended Euclid

Find p, q: Guess and check with big numbers.

Prime Number Theorem: About 1 in $\ln(n)$ numbers less than n are prime for big n.

For our size range, that's ~1 in 1000.