NUMBER THEORY AND CODES

Álvaro Pelayo
WUSTL

Talk Goal

To develop codes of the sort

- can tell the world how to put messages in code (**public key cryptography**)
- only you can decode them

Structure of Talk

Part I: Number theory background

Part II: RSA Codes

- **R** for Ronald Rivest
- **S** for Adi Shamir
- **A** for Leonard Adleman

PART I: NUMBER THEORY BACKGROUND

Integer Numbers

$\ldots\ldots, -3, -2-1, 0, 1, 2, 3, 4, 5, \ldots\ldots$

Divisibility

s is a **divisor** of t if there is an integer k such that

$$t = k \cdot s$$

Examples

- 1 is a divisor of every number m, since $m = m \cdot 1$
- 3 and 4 are divisors of 12 since $12 = 3 \cdot 4$
- 3 is <u>not</u> a divisor of 10 since

  $$10 = k \cdot 3$$

  is never true, for k integer

Prime Numbers

an integer p greater than 1 is **prime** if

the only divisors of p are 1 and p

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, \ldots$$

Not prime

- 4 because 2 is divisor
- 6 because 2 and 3 are divisors
- 8 because 2 and 4 are divisors
- ...

Factorization into primes

Any positive integer m can be written uniquely as

$$m = p_1^{k_1} \cdot p_2^{k_2} \cdot p_3^{k_3} \cdots p_n^{k_n}$$

with p_1, p_2, p_3, ..., p_n primes and

$$1 < p_1 < p_2 < p_3 < \ \ldots < p_n$$

Examples

- $8 = 2^3$
- $12 = 2^2 \cdot 3$
- $28 = 2^2 \cdot 7$
- $90 = 2 \cdot 3^2 \cdot 5$

Greatest common divisor of a, b, call it gcd(a, b)

- Look at the list of divisors of a
- Look at the list of divisors of b
- gcd(a, b) is the greatest number which is in both lists

Example: what is gcd(8, 12)?

- Divisors of 8: $\mathbf{1}$, $\mathbf{2}$, $\mathbf{4}$, 8
- Divisors of 12: $\mathbf{1}$, $\mathbf{2}$, 3, $\mathbf{4}$, 6, 12
- Common divisors of 8 and 12: 1, 2, 4
- gcd(8, 12) $=$ 4

Very slow method if a, b large, need better method:

Euclidean Algorithm

Euclidean Algorithm

Goal: given a, b, to find gcd(a, b)

- **Step 1**: divide large number by small number
- **Step 2**: divide small number by remainder
- **Step 3**: keep dividing until 0 remainder

One can verify: gcd(a, b) = last nonzero remainder

Example: take $a = 1001$, $b = 343$.

$$1001 = 2 \cdot 343 + 315$$
$$343 = 1 \cdot 315 + 28$$
$$315 = 11 \cdot 28 + 7$$
$$28 = 4 \cdot 7 + 0$$
$$\text{gcd}(1001,\, 343) = 7$$

Algorithm Backwards:

$$\begin{aligned} \text{gcd}(a,\, b) = 7 &= 315 - 11 \cdot 28 \\ &= 315 - 11(343 - 1 \cdot 315) \\ &= 12 \cdot 315 - 11 \cdot 343 \\ &= 12(1001 - 2 \cdot 343) - 11 \cdot 343 \\ &= 12 \cdot 1001 - 35 \cdot 343 \\ &= 12 \cdot a + (-35) \cdot b \end{aligned}$$

Congruence between two numbers

$$a \equiv b \pmod{N} \quad \text{if } N \text{ is a divisor of } b - a$$

Examples:

- $16 \equiv 1 \pmod 3$
- $21 \equiv 5 \pmod 8$

Number modulo an integer (slightly informal)

$$[a]_N := \text{ remainder of dividing } a \text{ by } N$$

$$0 \leq [a]_N < N$$

($a \geq 0$, otherwise add a multiple of N to a)

Examples:

- $[10]_2 = 0,\ [17]_5 = 2,\ [32]_5 = 2,\ [-4]_{10} = 6,\ [-47]_5 = 3$
- $[17,213]_{10} = 3,\ [43,596]_{100} = 96$
- If $0 \leq a < N$, $[a]_N = a$, for example:

$$[1]_4 = 1,\ [2]_4 = 2,\ [3]_4 = 3$$

From the definition: $[a]_N = [b]_N$ if and only if $a \equiv b \pmod N$
Also: $[a \cdot b]_N = [a]_N \cdot [b]_N$, $[\text{a+b}]_N = [a]_N + [b]_N$

PART II: RSA MESSAGE ENCODING

From words to numbers

- $A = 01$
- $B = 02$
- $\ldots$
- $Z = 26$
- 00 for space

A message is large number, about 200 digits

Example of message

$$x = \text{THIS COURSE IS NICE}$$

in code is

$$x = 200809190003142118190500091900140903 05$$

Idea of RSA Codes

- **Start with**: message x ($\simeq$ 200 digits),

- **Construct:** Encoding Function

$$E\left([\text{integer}]_N\right) \;=\; [\text{another integer}]_N$$

(N is a large number of our choice, about 10^{200} digits)

- **You send**: encoded message $E([x]_N)$

- **Receiver gets**: $E([x]_N)$

- **Receiver decodes it** using the inverse of E, call it D

$$D\left(E\left([x]_N\right)\right) \;=\; [x]_N$$

Properties E and D must satisfy

- E **easy** to calculate (**PUBLIC**)
- D **hard** to calculate (**SECRET**)

- **Easy**: small computer time ($<$ 1 second)
- **Hard**: large computer time (quadrillions of years)

How does one find

Encoding Function E ?

and

Decoding Function D ?

Using the method invented by

Rivest, Shamir and Adleman:

RSA method

RSA method to find E and D

- **Step 1**. Choose large prime numbers p, q ($\simeq$ 100 digits)
  Example. $p = 11$, $q = 13$,

- **Step 2**. Let $N = p \cdot q$
  Example. $N = p \cdot q = 11 \cdot 13 = 143$

- **Step 3**. Let $A = (p - 1) \cdot (q - 1)$
  Example. $A = (p-1) \cdot (q-1) = (11-1) \cdot (13-1) = 120$

- **Step 4**. Pick $1 \leq e < A$ with $\gcd(e, A) = 1$
  Example. $e = 53$ no common divisors with $A = 120$

Step 5. Define the Encoding Function

$$E\ ([x]_{143})\ =\ [x^e]_{143}$$

Example.

$$E([x]_{143}) = [x^{53}]_{143}$$

(From now on, we will write to $[x^{53}]_{143} = [x^{53}]$)

Observation:

$$[x^e] = [x \cdot \ldots \text{(e times)} \ldots \cdot x]$$

- **Step 6**. Find the solution $1 \leq d < A$ to $e \cdot d \equiv 1 \pmod{A}$
  Euclidean Algorithm backwards for e, A gives d, f:

  $$e \cdot d + A \cdot f = \text{gcd}(e, A) = 1,$$

  hence $e \cdot d = 1 - A \cdot f$, and therefore

  $$e \cdot d \equiv 1 \pmod{A}$$

  Example. Need to solve $53 \cdot d \equiv 1 \pmod{120}$

  $$\begin{aligned} 120 &= 2 \cdot 53 + 14 \\ 53 &= 3 \cdot 14 + 11 \\ 14 &= 1 \cdot 11 + 3 \\ 11 &= 3 \cdot 3 + 2 \\ 3 &= 1 \cdot 2 + 1 \\ 2 &= 2 \cdot 1 + 0, \quad \text{hence} \end{aligned}$$

  $$\begin{aligned} 1 &= 3 - 2 \\ &= 3 - (11 - 3 \cdot 3) \\ &= 4 \cdot 3 - 11 \\ &= 4(14 - 11) - 11 \\ &= 4 \cdot 14 - 5 \cdot 11 \\ &= 4 \cdot 14 - 5(53 - 3 \cdot 14) \\ &= 19 \cdot 14 - 5 \cdot 53 \\ &= 19(120 - 2 \cdot 53) - 5 \cdot 53 \\ &= 19 \cdot 120 - 43 \cdot 53, \quad \text{hence} \end{aligned}$$

$$(-43) \cdot 53 = 1 - 19 \cdot 120$$

$$(-43) \cdot 53 \equiv 1 \pmod{120}$$

Since $[-43]_{120} = [77]_{120}$,

$$d = 77$$

- **Step 7**. Define the Decoding Function

$$D\ ([x])\ =\ [x^d]$$

Example.

$$D([x]) = [x^{77}]$$

End of RSA Method

Why is D the inverse of E?

$$D\ (E([x])) = E\ (D([x])) = [x^{e \cdot d}]$$

Using a theorem (by Fermat) one can check:

$$[x^{e \cdot d}] = [x]$$

Computation of encoded message $E([97]) = [97]^{53}$

Step 1. Decompose $e = 53$ in sum of powers of 2

$$53 = 32 + 16 + 4 + 1 = 2^5 + 2^4 + 2^2 + 2^0$$

Step 2. Express $E([97])$ as a product

$$\begin{aligned} E([97]) &= [97]^{53} = [97]^{1+4+16+32} \\ &= [97]^1 \cdot [97]^4 \cdot [97]^{16} \cdot [97]^{32} \end{aligned}$$

Step 3. Compute [97] to the above powers of 2

$$[97]^2 = [-46]^2 = [2116] = [114] = [-29]$$

$$[97]^4 = [-29]^2 = [841] = [126] = [-17]$$

$$[97]^8 = [-17]^2 = [289] = [3]$$

$$[97]^{16} = [3]^2 = [9]$$

$$[97]^{32} = [9]^2 = [81] = [-62]$$

Step 4. Final computation

$$\begin{aligned} E([97]) &= [97]^{53} \\ &= [97]^1 \cdot [97]^4 \cdot [97]^{16} \cdot [97]^{32} \\ &= [97] \cdot [-17] \cdot [9] \cdot [-62] \\ &= [-46] \cdot [-17] \cdot [9] \cdot [-62] \\ &= -[46 \cdot 17] \cdot [9 \cdot 62] \\ &= [782] \cdot [558] \\ &= -[67][-14] \\ &= [67 \cdot 14] = [938] = [80] \end{aligned}$$

Computation of decoded message $D([80]) = [80]^{77}$

Using same method as earlier

$77 = 1 + 4 + 8 + 64$

$$\begin{aligned}
[80]^{77} &= [80] \cdot [80]^4 \cdot [80]^8 \cdot [80]^{64} \\
&= [80] \cdot [-62] \cdot [-17] \cdot [-62] \\
&= [1360] \cdot [3884] \\
&= -[73] \cdot [-17] \\
&= [73] \cdot [17] \\
&= [1241] \\
&= [97]
\end{aligned}$$

The original message!

Exercise 1: In constructing a code with $p = 17$, $q = 19$ suppose that the encoding exponent is $e = 35$. What should the decoding exponent d be?

Exercise 2:
(A) Decode the message 127 using the code in exercise 1.
(B) Encode the message found in (A), and check the result is precisely 127.

How can one break the code?

- If can factor N into $p \cdot q \to$ can find d, and function D
- **TELL** e, N **everyone**: they can send encoded messages
- **KEEP** d **secret**: you only can decode them

- < 1 sec to find $E([x])$ if e known, or $D([x])$ if d known
- quadrillions of years to find $D([x])$ if d NOT known (based on current known algorithms)

ADDITIONAL MATERIAL:
Verification that D is inverse of E

Need: Fermat's little theorem:

if p is prime and $[a]_p \neq 0$, $a^{p-1} \equiv 1 \bmod p$

First, $D(E([x])) = D([x^e]) = [(x^e)^d] = [x^{d \cdot e}]$

<u>We want to check</u>: $[x^{d \cdot e}] = [x]$, equivalently $x^{d \cdot e} - x \equiv 0 \pmod{N}$

Hence we need to check that $N = p \cdot q$ is a divisor of $x^{d \cdot e} - x$

Enough to check that p is divisor of $x^{d \cdot e} - x$, i.e. $[x^{d \cdot e} - x]_p = 0$

<u>We know</u>:

$d \cdot e \equiv 1 \bmod A$, so there exists k such that $d \cdot e = 1 + k \cdot A$

Since $A = (p-1) \cdot (q-1)$, $k \cdot A = (p-1) \cdot m$, where $m = k \cdot (q-1)$

<u>Therefore</u>:

$x^{d \cdot e} - x = x^{1+k \cdot A} - x = x(x^{k \cdot A}) - x = x(x^{(p-1) \cdot m}) - x = x(x^{p-1})^m - x$

$[x^{d \cdot e} - x]_p = [x(x^{p-1})^m - x]_p = [x(1)^m - x]_p = [x(1) - x]_p = 0$