Class Notes

Kathryn Bafford

September 22, 2016

Euclid's Algorithm is fast.

If a>b, we can show that the total number of arithmetic operations required to compute gcd(a,b) is less than C $\log a$ for a constant C. In big O notation this is O($\log a$).

Modular Exponentiation:

<u>Example:</u> Suppose you want to compute 3^{521} (mod 19). If we compute 3^{521}, it is a number with 249 decimal digits. It would become difficult even for a computer.

Trick: Write the exponent in binary.

521 = 512 + 8 + 1

= $2^9+2^8+2^0$

In binary it is: 1000001001

Use repeated squaring and reduce (mod 19) after each step.

3^1 = 3 (mod 19)

3^2 = 9 (mod 19)

3^4 = 81 (mod 19) = 5 (mod 19)

3^8 = 6 (mod 19)

3^{16} = 17 (mod 19)

3^{32} = 4 (mod 19)

3^{64} = 16 (mod 19)

3^{128} = 9 (mod 19)

3^{256} = 5 (mod 19)

3^{512} = 6 (mod 19) ($3^{512} ¿ (3^8)(3^1) = (6)(6)(3) = ¿$ (17)(3) = 13 (mod 19)

We computed 3^{512} (mod 19) by doing 12 multiplications where every number was smaller than 19. In general, modular exponentiation lets us compute a^n (mod m) using at most 2 $\log n$ multiplications where no numbers are bigger than m.

General steps for Modular Exponentiation:

1. Write out exponent in binary. Find the position of the largest "1" in its binary representation. Call this "k"
2. Do repeated squaring of a (mod m) k times. Save all steps.
3. Multiply the results of repeated squaring for every position where there was a "1" in the binary representation in step 1. Reduce the answer (mod m).

Fermat's Little Theorem:

If p is a prime number and a is not divisible by p, then a^{p-1} = 1 (mod p).

<u>Example</u>: p = 5 and a = 2

2^4 = 16 = 1 (mod 5)

<u>Example</u>: p = 11 and a = 2

2^{10} = 1024 = 1 (mod 1)

~11 divides 1023 because 1-0+2-3 = 0~

<u>Non-example</u>: p = 6 and a = 2

2^5 = 32 = 2 (mod 6)

Proof:

Let S = {1, 2, 3, …, p-1} (all non-zero residues).

Define Ψa(x) = S --> S

Ψa(x) = ax (mod p)

Check that Ψa(x) is one-to-one and onto. Now suppose we have two elements x and y, x $\neq$ y, but Ψa(x) = Ψa(y). This means that ax = ay. A has an inverse (mod p). $a^{-1}ax$ = $a^{-1}ay$ which results in x = y. Since Ψa(x) is one-to-one x = y is a contradiction.

Compute now: Ψa(1)* Ψa(2)* … * Ψa(p-1)

= a * 2a * 3a * … * (p-1)a

= a^{p-1} (1 * 2 * 3 * … * (p-1))

Ψa(1)* Ψa(2)* … * Ψa(p-1) = 1 * 2 * 3 * … * (p-1)

a^{p-1} ~~(1 * 2 * 3 * … * (p-1))~~ = ~~(1 * 2 * 3 * … * (p-1))~~ (mod p)

a^{p-1} = 1 (mod p)

<u>Example</u>: Compute 2^{64} (mod 11)

By FLT, we know 2^{10} = 1 (mod 11)

2^{64} = $\overset{2}{\underset{¿}{(¿¿10)}}^{6}$ 2^4 (mod 11)

2^{64} = 1^6 2^4 (mod 11)

2^{64} = 16 (mod 11)

2^{64} = 5 (mod 11)

Like to have a version of FLT for composite numbers.

Euler's Theorem work for composite numbers. Need Euler's φ (phi) function. This counts how many residues (r) modulo n have gcd 1 with n. gcd(r,n) = 1

Example: φ(26) =12

No evens or 13

φ(27) = 18

φ(29) = 28

So if p is a prime number φ(p) = p-1

If n = p^k , $\varphi(n) = \varphi(p^k) = \left(\frac{p-1}{p}\right) p^k$

Example: 27 = 3^3

$\varphi(27) = \varphi(3^3) = \left(\frac{2}{3}\right) 3^3 = 2 * 3^2 = 18$

Using the Chinese Remainder Theorem, if m and n have gcd(m,n) = 1, then

φ(mn) = φ(m) * φ(n). $\varphi(p^k * q^l) = p^k * q^l \left(\frac{p-1}{p}\right)\left(\frac{q-1}{q}\right)$

Theorem: For any n

$$\varphi(n) = n\,\pi\left(\frac{p-1}{p}\right)$$