Indices

Recall that if a and m are relatively prime, the order of a modulo m is defined to be

$$e_m(a) := \min\{x > 0 \mid a^x \equiv 1 \pmod{m}\}$$

Last time we saw that

$$a, a^2, a^3, \ldots, a^{e_m(a)}$$

are all incongruent modulo m. This means, in particular, that if p is prime and g is a primitive root modulo p (ie., $e_p(g) = p-1$), then powers of g "generate" all of the congruence classes modulo p. That is,

$$g, g^2, g^3, \ldots, g^{p-1}$$

account for all of the non-zero congruence classes modulo p.

If we've chosen a prime p and primitive root g, then the power we raise g to in order to obtain the congruence class of a given number b is called the index of b ~~modulo~~ with respect to p and g

That index we will denote by $I(b)$:

$$I(b) := \min\{x>0 \mid g^x \equiv b \pmod{p}\}$$

For instance, if $p=7$ then 3 and 5 are the primitive roots of modulo 7. (These are the only primitive roots as the primitive root theorem tells us that 7 has $\varphi(7-1) = \varphi(6) = \varphi(2)\cdot\varphi(3) = 1\cdot 2 = 2$ primitive roots.)

Let's make a table of indices for each primitive root, 3 and 5. That is, for each number 1, 2, 3, 4, 5, 6, 7, we record what minimum x we need to raise 3 (or 5) to to get the given value. For example, the index of 2 wrt 3 is 2 because

$$3^2 = 9 \equiv 2 \pmod{7}$$

but the index of 2 wrt 5 is 4 as

$$5^4 = 625 \equiv 2 \pmod{7}$$

Indices wrt 3:

a	1	2	3	4	5	6
$I(a)$	6	2	1	4	5	3

Indices wrt 5:

a	1	2	3	4	5	6
I(a)	6	4	5	2	1	3

One useful property of these indices is that they obey the same rules as logarithms; modulo $p-1$.

Theorem

Suppose p is a prime, and $I(a)$ is the index of a wrt p and some chosen primitive root g. Then

- $I(ab) \equiv I(a) + I(b) \pmod{p-1}$
- $I(a^k) \equiv k\, I(a) \pmod{p-1}$.

Pf

- $g^{I(ab)} \equiv ab$
  $\equiv g^{I(a)}\, g^{I(b)}$
  $\equiv g^{I(a)+I(b)} \pmod{p}$
  $\Rightarrow g^{I(ab)-I(a)-I(b)} \equiv 1 \pmod{p}$
  $\Rightarrow I(ab) - I(a) - I(b) = (p-1)q$ for some q
  $\Rightarrow I(ab) \equiv I(a) + I(b) \pmod{p-1}$

- $g^{I(a^k)} \equiv a^k$
  $\equiv \underbrace{a \cdot a \cdot \ldots \cdot a}_{k \text{ times}}$
  $\equiv \underbrace{g^{I(a)} \cdots g^{I(a)}}_{k \text{ times}}$
  $\equiv \left[g^{I(a)}\right]^k \equiv g^{k I(a)} \pmod{p}$

$$\Rightarrow g^{I(a^k) - kI(a)} \equiv 1 \pmod{p}$$
$$\Rightarrow p-1 \mid I(a^k) - kI(a)$$
$$\Rightarrow I(a^k) \equiv kI(a) \pmod{p}$$

□

Once we have a table of indices, we can use it to help us solve congruences modulo a prime.

For example, suppose we wanted to find the values of x such that

$$4x^{13} \equiv 6 \pmod{7}$$
$$\Rightarrow I(4x^{13}) = I(6)$$
$$\Rightarrow I(4) + 13I(x) \equiv I(6) \pmod{6}$$
$$\Rightarrow 2 + 13I(x) \equiv 3 \pmod{6} \quad (\text{using } g=5)$$
$$\Rightarrow 13I(x) \equiv 1 \pmod{6}$$
$$\Rightarrow 6 \mid 13I(x) - 1$$
$$\Rightarrow 6y = 13I(x) - 1$$
$$\Rightarrow 13I(x) - 6y = 1$$

Now we can solve this equation by noting

$$13 = 6 \cdot 2 + 1$$
$$\Rightarrow 1 = 13 \cdot 1 - 6 \cdot 2$$

So we take $I(x) = 1$, meaning $x = 5$.

Because of the similarities between the index and the usual logarithmic functions, the index is sometimes called the discrete logarithm. That is, given a number b and a primitive root g modulo p, the discrete logarithm problem is concerned with finding a k such that

$$g^k \equiv b \pmod{p}$$

To date, no one knows an efficient way to find k on a "traditional" computer — the fastest currently known algorithms require exponential time in the number of digits of p. (There are efficient algorithms for solving the discrete log problem on quantum computers.)

Because the discrete log problem is hard to solve, it can be used as the basis of a cryptosystem. One such cryptosystem is called ElGamal

ElGamal is a public key cryptosystem, meaning we have a public key that we publish and which anyone can use to encrypt a message they want to send us, and a private key we use for decrypting messages.

Here's how we generate our public and private keys in ElGamal:

1. Pick a large prime p and a primitive root g
2. Pick a random number k and compute $S = g^k$ modulo p.
3. We keep k secret, it's our ~~public~~ private key
4. We publish g^k's, g, and p — these form the public key

If someone wants to send us a message, they do the following

1. Convert their message into a number (or collection of numbers), say $2 \leq M < p$.
2. Choose random number $1 \leq r < p$.
3. They then send us the ciphertext, a pair of numbers, (g^r, MS^r)

To decrypt a message (g^r, MS^r) we

1. Compute $[g^r]^k$ modulo p
2. Find u solving $ug^{rk} \equiv 1 \pmod{p}$
3. Compute uMS^r modulo p.

Note $uMS^r = uM(g^k)^r = ug^{rk}M \equiv M \pmod{p}$

Example of ElGamal

- Key generation
  1. Pick a prime: say $p = 100609$
  2. Find a primitive root: $g = 7$
  3. Pick a random k for our private key: $k = 958$
  4. The public key is (g^k, g, p)
     $(45822, 7, 100609)$

- Encryption: To encrypt "TO BE OR NOT TO BE"
  1. Convert message to numbers. The ASCII codes are
     $[84, 79, 66, 69, 79, 82, 78, 79, 84, 84, 79, 66, 69]$
     We'll put these together in 4 digit blocks:
     $[8479, 6669, 7982, 7879, 8484, 7966, 8900]$
  2. For each number we pick a random r and compute
     $(7^r, M \cdot 45822^r)$ ← mod p
  3. $[(15050, 62190), (23194, 45691), (36384, 56823), (61695, 4131), (84720, 4622), (15845, 11628), (378, 95330)]$

- To decrypt:
  1. Each piece of ciphertext is a pair (A, B)
     Compute A^k mod p:
     $[25913, 12578, 96821, 1764, 41485, 70441]$
  2. Find $(A^k)^{-1}$ mod p
     $[22460, 79183, 18160, 65464, 9595, 50660, 28478]$
  3. Compute $A^{k-1}B$ mod p
     [8479 6669 7982 7879 8484 7966 8900]