Powers in Modular Arithmetic, and RSA Public Key Cryptography

Lecture notes for Access 2006, by Nick Korevaar.

It was a long time from Mary Queen of Scotts and substitution ciphers until the end of the 1900's. Cryptography underwent the evolutionary and revolutionary changes which Simon Singh chronicals in *The Code Book*. By the mid 1970's there were amazingly complicated encryption algorithms which could be made essentially unbreakable. For example, in Chapter 6, Singh mentions the Lucifer cipher, a special version of which is known as the Data Encryption Standard, or DES.

However, no matter how convoluted the the encryption methods were, and how frequently the keys were changed for security reasons, all methods required that both parties to the message possessed *the* key for encryption and decryption.... and it was just assumed, because this had always been the case, that if you possessed the method to encrypt a message, then this was equivalent after perhaps a little work, to also knowing how to decrypt it. By the mid 1970's there were thousands of couriers flying all over the world, whose only job was to transfer cipher keys.

As the the precursor to the internet, namely the ARPAnet, was beginning to grow, Whitfield Diffie and Martin Hellman, as well as others, realized the huge potential for electronic transactions, together with the need for assured security. Diffie-Hellman were perhaps the first to realize that there was an entirely new way to think of cryptography; that perhaps there were encryption keys which you could let everyone in the world know, but for which you could never the less keep secret the decryption key. This would solve the problem of key distribution, since if you wanted to receive secure messages you could tell the world how to encrypt anthing they wanted to send you, but only you would know the decryption key which could stay safely at home. Diffie-Hellman called such encryption keys, "trapdoor", or "one way" functions, because knowing the encryption function did not automatically allow clever people to work out the decryption function. In 1977, Ronald Rivest, Adi Shamir and Leonard Adleman described one of the easiest one-way functions, and the resulting method of public key cryptography is called RSA, in their honor. As we shall see, this method relies on number theory and modular arithmetic, and will use everything we've been talking about so far.

Remark 1. If you use the encryption function $f(x) \equiv x + a \mod N$, which corresponds to Caesar shifts, anyone can deduce the decryption function $g(x) \equiv x - a \mod N$. Similarly, if you specify the encryption function $f(x) \equiv ax \mod N$, for $gcd(a, N) = 1$, then a lot of people could use the Euclidean algorithm to find the multiplicative inverse b of a, $\mod N$, and use the decryption function $g(x) \equiv bx \mod N$. So neither of these examples is a one-way function!

Remark 2. In our examples we've been assigning numbers to each letter of a plaintext and then using modular arithmetic to construct a cipher. In practice this just amounts to a substitution cipher, and so can be broken easily with frequency analysis. What we will do for RSA cryptography is to use HUGE moduli N and break messages into packets whose lengths fall into the residue range of N. For example, if N has at least 13 digits, then you can use the table on page 9 of Tom Davis' notes, *Cryptography*, to convert any 6-character expression into a number in the residue range of N, since each character is represented with two digits.

Example 1. Use the Davis table on page 9 of his notes to convert the two word sentence *Hello there!* into two number packets, each of which is less than 10^{12}.

Example 2. For an example we can work by hand, let our modulus $N = 11$. Let our candidate encryption function be $f(x) \equiv x^2 \mod 11$. Complete the table below and explain why this function won't work to encrypt the numbers in our residue range.

x	0	1	2	3	4	5	6	7	8	9	10
x^2	0	1	4	9	5						

Example 3. Keeping $N = 11$, show that the function $f(x) \equiv x^3 \mod 11$ does encrypt (permute) the residue numbers:

x	0	1	2	3	4	5	6	7	8	9	10
x^3	0	1	8								

Example 4. We might hope that if our encryption function is $f(x) \equiv x^e \mod N$, then our decryption function is $g(x) \equiv x^d \mod N$, for some power d. For the encryption function in the previous example $e = 3$. We shall now deduce a possible value for the decryption power d: Since $f(2) \equiv 8 \mod 11$, we want $g(8) \equiv 2$, i.e.

$$8^d \equiv 2 \mod 11.$$

Compute successive powers of 8 until you are able to solve this equation for d. (Hint: $d = 7$.)

Exercise 1. But we need to check that the decryption power $d = 7$ works for every x in our residue range! Let group number x check that this is so, for the residue number $x + 1$, except that group 1 gets to check $x = 10$, since we just did $x = 2$. Be clever to minimize your computing!

Exercise 2. RSA cryptography uses moduli $N = pq$, where p and q are (HUGE) prime numbers. For $p = 3$, $q = 5$, $N = 15$, use the $\bmod 15$ table of powers to figure out good and bad encryption powers e.

Exercise 3. $f(x) \equiv x^3 \mod 15$ is a good encryption function. Figure out a possible decryption function $g(x) \equiv x^d \mod 15$.

All is Explained

We've been doing a lot of experimentation with modular arithmetic, which is a great way to get ideas about what might be true. Number theory has been a favorite for many famous mathematicians, and so some of their names are attached to the following important theorems, which maybe they were also led to by experimentation. These results are the underpinning of RSA cryptography.

Theorem 1 (Fermat's Little Theorem). If p is a prime and if $0 < a < p$ is a residue number, then $a^{p-1} \equiv 1 \mod p$.

Proof. Pick any non-zero residue a as above, and consider the corresponding row of the mod p multiplication table. (You can make this less abstract by using the mod 7 table as an example.) Since a has a multiplicative inverse mod p, $ax \equiv ay$ only when $x \equiv y$. Therefore the residues across the row, namely the numbers

$$1a, 2a, 3a, ..., (p-1)a$$

must all be different, i.e. a permuation of the non-zero residues $1, 2, ..., (p-1)$. Thus the product of all these terms satisfies

$$(1a)(2a)...(p-1)a \equiv (1)(2)...(p-1) \mod p,$$
$$a^{p-1}(1)(2)...(p-1) \equiv (1)(2)...(p-1) \mod p.$$

Multiply both sides of this equation by the multiplicative inverses of $2, 3, ...(p-1)$, i.e. cancel the term $(2)(3)...(p-1)$ from both sides of the equation. Deduce

$$a^{p-1} \equiv 1 \mod p.$$

□

Corollary 1. if $f(x) \equiv x^e \mod p$, and d is a multiplicative inverse of e, mod $p-1$, then $g(x) \equiv x^d \mod p$ is the inverse function of f (on the set of residue numbers).

Proof. Notice that if $x = 0$ the result holds. Thus we can assume $x = a$, a non-zero residue. Since e and d are multiplicative inverses mod $p-1$, we have

$$ed = 1 + m(p-1)$$

for some counting number m. Thus

$$g(f(a)) \equiv g(a^e) \equiv (a^e)^d \equiv a^{ed}$$
$$\equiv a^{1+m(p-1)} \equiv a^1(a^{p-1})^m \equiv a(1^m) \equiv a,$$

by Fermat's Little Theorem! This shows that g is the inverse function to f. □

Example 5. For $p = 11$ and $e = 3$, we found $d = 7$. Notice that 7 is a multiplicative inverse for 3, mod 10.

Theorem 2 (Euler-Fermat Theorem). If $N = pq$ is a product of two prime numbers, define $N_2 = (p-1)(q-1)$. If a is any residue number mod N which has no common factors with N, then

$$a^{N_2} \equiv 1 \mod N$$

.

Proof. The idea of the Euler-Fermat Theorem is very similar to that in Fermat's little theorem, and can be illustrated in the mod 15 multiplication table. Since $(a, N) = 1$, a has a multiplicative inverse. (We say a is a "unit", for short). Consider the list of the products of a with all the other units, taken from row a of the multiplication table. You must obtain a permutation of the original unit collection because $ax \equiv ay$ only if $x \equiv y$. Thus the product of all terms which are a times a unit must just equal the product of all units. Cancel the unit terms as previously to deduce $a^{N_2} \equiv 1 \mod N$, where N_2 is the number of units. Since N_2 is equivalently the number of non-zero residues which don't have factors of p or q, we can count N_2 by starting with the the number of non-zero residues, $pq - 1$, and subtracting off those which are multiples of p or q. Precisely,

$$N_2 = pq - 1 - (q - 1) - (p - 1) = pq - p - q + 1 = (p - 1)(q - 1).$$

This completes the proof. □

Corollary 2. Let $N = pq$, $N_2 = (p-1)(q-1)$ as above. If e is relatively prime to N_2. Then $f(x) \equiv x^e \mod N$ has an inverse function $g(x) \equiv x^d \mod N$, where d is a multiplicative inverse of $e, \mod N2$

Proof. If $x = a$ is a unit, then

$$g(f(a)) \equiv (a^e)^d \equiv a^{ed} \equiv a^{1+mN_2} \equiv a(1^{N_2})^m \equiv a(1)^m \equiv a.$$

If x is not a unit, then there is a special argument which checks this corollary. You can find it in the original paper by Rivest-Shamir-Adelman. (There is a link to the RSA paper on our home page.) It depends on the fact that all residue numbers x, not just units, satisfy an equation which looks like the Euler-Fermat identity multiplied by x:

$$x^{N_2+1} \equiv x \mod N.$$

□

Example 6. For $p = 3$ and $q = 5$ we have $N = 15$, $N_2 = 8$. When we used the encryption power $e = 3$, we found the decryption power $d = 3$. Notice that $ed \equiv 1 \mod 8$. Also notice, from the power table, that $x^9 \equiv x$ for all residues x.