Mathematical preliminaries of crypto
More Concepts of Number Theory

Peeter Laud

March 22, 2017

1 Chinese Remainder Theorem

We finished the previous episode with the description of the method of solving a linear congruence after we had simplified it to a form $ax \equiv b \pmod{n}$. We received the answer in the form $x \equiv x_0 \pmod{n'}$, where n' was a divisor of n and $x_0 \in \{0, \ldots, n'-1\}$.

What if we have several such results regarding a variable x? This means, we have a system of congruences of the form

$$\begin{cases} x \equiv a_1 & (\text{mod } n_1) \\ x \equiv a_2 & (\text{mod } n_2) \\ \ldots\ldots\ldots\ldots\ldots\ldots & \\ x \equiv a_k & (\text{mod } n_k) \end{cases} \tag{1}$$

Theorem 1 (Chinese Remainder Theorem)**.** *If $n_i \perp n_j$ for each $1 \leq i < j \leq k$, then the system of congruences (1) has a unique solution* modulo $N = \prod_{i=1}^{k} n_i$.

The proof of the existence of the solution will also give us an algorithm for finding that solution.

Proof. For each $i \in \{1, \ldots, k\}$ let $m_i = N/n_i$. The numbers m_i are natural numbers, because n_i was one of the factors of N. We have $n_i \perp m_i$, because (1) m_i is the product of all n_j, where $i \neq j$, and (2) $n_i \perp n_j$ for all these n_j.

Let r_i be the inverse of m_i *modulo* n_i. In other words, r_i is such that $r_i m_i \equiv 1 \pmod{n_i}$. Such an r_i exists because $\overline{m_i}$ is invertible in the ring $\mathbb{Z}_{n_i}$. We can use the extended Euclid's algorithm to compute r_i.

Define

$$x_0 = \sum_{i=1}^{k} a_i r_i m_i \ . \tag{2}$$

Let us now show that x_0 is a solution to all congruences in (1). If we simplify the expression (2) *modulo* n_i, we'll get

$$x_0 = \sum_{j=1}^{k} a_j r_j m_j \overset{(*)}{\equiv} a_i r_i m_i \overset{(**)}{\equiv} a_i \pmod{n_i}$$

because

- All but one addend of the sum (2) is congruent to 0 *modulo* n_i. Indeed, if $j \neq i$ then m_j is a product that includes n_i as a factor. This explains the congruence $(*)$.
- $r_i m_i \equiv 1 \pmod{n_i}$ by the definition of r_i. This explains the congruence $(**)$.

It is also clear that any number of the form $x_0 + lN$ (where $l \in \mathbb{Z}$) also satisfies (1) because N is a multiple of all n_i.

Let us now show that the solution to (1) is unique *modulo* N. Assume the contrary, let both x_1 and x_2 be solutions of the system of congruences. The congruences $x_1 \equiv a_i \pmod{n_i}$ and $x_2 \equiv a_i \pmod{n_i}$ imply $x_1 - x_2 \equiv 0 \pmod{n_i}$, which is equivalent to $n_i \mid (x_1 - x_2)$. This holds for all $i \in \{1, \ldots, k\}$. Hence also $\operatorname{lcm}(n_1, \ldots, n_k) \mid (x_1 - x_2)$. The numbers $n_1, \ldots, n_k$ are pairwise mutually prime, hence their least common multiple is equal to their product N. We get $N \mid (x_1 - x_2)$ or $x_1 \equiv x_2 \pmod{N}$. □

Let us do an example and solve the following system of congruences:

$$\begin{cases} x \equiv 2 & \pmod 3 \\ x \equiv 2 & \pmod 4 \\ x \equiv 1 & \pmod 5 \end{cases}$$

Here $a_1 = a_2 = 2$, $a_3 = 1$, $n_1 = 3$, $n_2 = 4$, $n_3 = 5$. The moduli are pairwise mutually prime. Hence the system has a unique solution modulo $N = n_1 n_2 n_3 = 60$.

Following the proof above, let us compute the numbers $m_1 = N/n_1 = 20$, $m_2 = N/n_2 = 15$, $m_3 = N/n_3 = 12$. As next, we have to compute the numbers r_i which are the inverses of m_i *modulo* n_i. These are normally computed using the extended Euclid's algorithm, which we do not show here. But the results are

$$\begin{aligned} r_1 &= 20^{-1} \equiv 2^{-1} \equiv 2 \pmod 3 \\ r_2 &= 15^{-1} \equiv 3^{-1} \equiv 3 \pmod 4 \\ r_3 &= 12^{-1} \equiv 2^{-1} \equiv 3 \pmod 5 \ . \end{aligned}$$

We can now compute

$$x_0 = a_1 r_1 m_1 + a_2 r_2 m_2 + a_3 r_3 m_3 = 2 \cdot 2 \cdot 20 + 2 \cdot 3 \cdot 15 + 1 \cdot 3 \cdot 12 = 206 \equiv 26 \pmod{60} \ .$$

Hence the solution to the system is $x \equiv 26 \pmod{60}$.

How can we verify that we have found the correct answer? We can just check that $x_0 \bmod n_i = a_i$ for all i. Hence check: $26 \bmod 3 = 2$, $26 \bmod 4 = 2$, $26 \bmod 5 = 1$. We have not made any errors during our computation (or if we did any, those canceled out).

Exercise 1. *Solve the following systems of congruences and verify the results:*

$$\textit{(a)} \begin{cases} x \equiv 0 & \pmod 2 \\ x \equiv 4 & \pmod 7 \\ x \equiv 7 & \pmod 9 \end{cases} \qquad \textit{(b)} \begin{cases} x \equiv 1 & \pmod 3 \\ x \equiv 1 & \pmod 4 \\ x \equiv 0 & \pmod 5 \\ x \equiv 0 & \pmod 7 \end{cases}$$

Exercise 2. *Generate and solve such systems of congruences yourself.*

What if some moduli are not mutually prime? In this case we can no longer directly use the previous algorithm, but we can try to simplify the system (1) in the following general manner. The first congruence states $x \equiv a_1 \pmod{n_1}$. This congruence is satisfied by all integers of the form $x = n_1 x_1 + a_1$, where $x_1 \in \mathbb{Z}$. Let us substitute this into the second congruence, giving us $n_1 x_1 + a_1 \equiv a_2 \pmod{n_2}$. This congruence may have no solutions for x_1, or it has a solution of the form $x_1 \equiv a_2' \pmod{n_2'}$, where n_2' is some divisor of n_2. If there are no solutions then the whole system has no solutions. If there is a solution, then we can express x_1 as $x_1 = n_2' x_2 + a_2'$, where $x_2 \in \mathbb{Z}$. We can substitute it back to the expression of x, giving us $x = n_1(n_2' x_2 + a_2') + a_1$; this choice of x satisfies the two first congruences of (1). We can continue by substituting this into the third congruence, etc.

For example, let us solve the following system of congruences.

$$\begin{cases} x \equiv 3 & \pmod{4} \\ x \equiv 5 & \pmod{6} \\ x \equiv 6 & \pmod{7} \\ x \equiv 5 & \pmod{10} \end{cases}$$

The first congruence gives us

$$x = 4x_1 + 3 \qquad \text{for any } x_1 \in \mathbb{Z} \ .$$

Substituting it into the second congruence, we get

$$4x_1 + 3 \equiv 5 \pmod{6} \Rightarrow 4x_1 \equiv 2 \pmod{6} \Rightarrow$$
$$2x_1 \equiv 1 \pmod{3} \Rightarrow x_1 \equiv 2 \pmod{3} \ .$$

This congruence gives us

$$x_1 = 3x_2 + 2 \qquad \text{for any } x_2 \in \mathbb{Z}$$
$$x = 4(3x_2 + 2) + 3 = 12x_2 + 11 \qquad \text{for any } x_2 \in \mathbb{Z} \ .$$

Substituting it into the third congruence, we get

$$12x_2 + 11 \equiv 6 \pmod{7} \Rightarrow 5x_2 \equiv 2 \pmod{7} \Rightarrow x_2 \equiv 6 \pmod{7} \ .$$

This congruence gives us

$$x_2 = 7x_3 + 6 \qquad \text{for any } x_3 \in \mathbb{Z}$$
$$x = 12(7x_3 + 6) + 11 = 84x_3 + 83 \qquad \text{for any } x_3 \in \mathbb{Z} \ .$$

Substituting it into the fourth congruence, we get

$$84x_3 + 83 \equiv 5 \pmod{10} \Rightarrow 4x_3 \equiv 2 \pmod{10} \Rightarrow$$
$$2x_3 \equiv 1 \pmod{5} \Rightarrow x_3 \equiv 3 \pmod{5} \ .$$

This congruence gives us

$$\begin{aligned} x_3 &= 5x_4 + 3 && \text{for any } x_4 \in \mathbb{Z} \\ x &= 84(5x_4 + 3) + 83 = 420x_4 + 335 && \text{for any } x_4 \in \mathbb{Z} \ . \end{aligned}$$

Thus the solution to this system of congruences is $x \equiv 335 \pmod{420}$.

Exercise 3. *Generate and solve such systems of congruences yourself.*

2 Using CRT to speed up modular computations

When performing computations *modulo* $n = n_1 n_2$, where $n_1 \perp n_2$, and when the factors n_1 and n_2 are known to us, the Chinese Remainder Theorem allows us to perform those computations *modulo* n_1 and n_2. This is important because n_1 and n_2 are shorter than n and thus the operations *modulo* them take less time and space than the operations *modulo* n. If necessary, we can afterwards use Chinese Remainder Theorem to recover the result *modulo* n. Obviously, the method can be generalized to a larger number of factors of n. This method is made even more attractive by the fact that most of the algorithm for solving the system of congruences can be executed without even knowing the computation to be performed — the numbers m_i and r_i (and also the products $r_i m_i$) depend only on the moduli n_i. Only the computation of the linear combination (2) requires the results of the computation a_i *modulo* n_i.

For example, let us compute

$$E = (25 + 48) \cdot (63 - 14) \cdot (145 + 73 - 213) \pmod{420} \ . \tag{3}$$

The modulus is equal to $420 = 3 \cdot 4 \cdot 5 \cdot 7$. Hence we could compute E *modulo* 3, 4, 5, and 7, and afterwards combine the results. Let us perform these computations.

$$\begin{aligned} E &\equiv (1 + 0) \cdot (0 - 2) \cdot (0 + 1 - 0) \equiv (-2) \equiv 1 \pmod{3} \\ E &\equiv (1 + 0) \cdot (3 - 2) \cdot (1 + 1 - 1) \equiv 1 \pmod{4} \\ E &\equiv (0 + 3) \cdot (3 - 4) \cdot (0 + 3 - 3) \equiv 0 \pmod{5} \\ E &\equiv (4 + 6) \cdot (0 - 0) \cdot (5 + 3 - 3) \equiv 0 \pmod{7} \ . \end{aligned}$$

We now have to find E *modulo* 420. You have already found this value in exercise 1(b).

Exercise 4. *Compute E directly according to (3) and verify that the application of CRT gave the correct answer.*

As another example, let us compute $3^{20} \bmod 70$. We have $70 = 2 \cdot 5 \cdot 7$, hence we should compute 3^{20} *modulo* 2, 5, and 7, and then combine the results. We can also precompute:

$n_1 = 2$	$n_2 = 5$	$n_3 = 7$
$m_1 = 35$	$m_2 = 14$	$m_3 = 10$
$r_1 = 1$	$r_2 = 4$	$r_3 = 5$
$r_1 m_1 = 35$	$r_2 m_2 = 56$	$r_3 m_3 = 50$

Now, let us consider the powers of 3 according to various moduli.

$$\begin{aligned} 3^0, 3^1, 3^2, \ldots &= 1, 1, 1, \ldots \pmod 2 \\ 3^0, 3^1, 3^2, \ldots &= 1, 3, 4, 2, 1, 3, 4, \ldots \pmod 5 \\ 3^0, 3^1, 3^2, \ldots &= 1, 3, 2, 6, 4, 5, 1, 3, 2, \ldots \pmod 7 \ . \end{aligned}$$

By noticing the periods, we get $3^{20} \equiv 1 \pmod 2$, $3^{20} \equiv 1 \pmod 5$, $3^{20} \equiv 2 \pmod 7$. Hence

$$3^{20} \equiv 1 \cdot 35 + 1 \cdot 56 + 2 \cdot 50 = 191 \equiv 51 \pmod{70} \ .$$

Exercise 5. *Compute* $5^{22} \bmod 72$. *Also, perform other computations like this.*

The correctness of these methods of computation is justified by the following theorem.

Theorem 2. *Let* $n = n_1 \cdots n_k$, *where* $n_i \perp n_j$ *for all* $1 \leq i < j \leq k$. *Then* $\mathbb{Z}_n \cong \mathbb{Z}_{n_1} \times \cdots \times \mathbb{Z}_{n_k}$.

Here $\times$ denotes the direct product of rings — if $R_1, \ldots, R_k$ are rings, then $R_1 \times \cdots \times R_k$ is the set of tuples $(r_1, \ldots, r_k)$, where $r_i \in R_i$ and the operations are defined componentwise. It is easy to check that the direct product is again a ring. The notation $R \cong R'$ means that the rings R and R' are *isomorphic* — there exists a bijective mapping f from R to R', that is a *homomorphism* — it respects the operations of the ring.

Proof. We have to find this isomorphism f from $\mathbb{Z}_n$ to $\mathbb{Z}_{n_1} \times \cdots \times \mathbb{Z}_{n_k}$. We choose it as follows:

$$f(\overline{a}) = (\overline{a \bmod n_1}, \ldots, \overline{a \bmod n_k}) \ .$$

The mapping f is quite clearly a homomorphism because of the properties of the modular arithmetic.

What is the inverse f^{-1} of the mapping f? If we are given the numbers $a_1, \ldots, a_k$, where $a_i \in \mathbb{Z}_{n_i}$, how do we find a number $a \in \mathbb{Z}_n$, such that $a \bmod n_i = a_i$? By using the Chinese Remainder Theorem, of course! Thus the inverse mapping f^{-1} exists and f is a bijective mapping, hence an isomorphism. □

3 Euler's φ-function

Recall that if R is a ring, then R^* denoted the set of all invertible elements of R. We want to know: what is the size of $\mathbb{Z}_n^*$? In other words, how many numbers (*modulo* n) are there, that are mutually prime to n? Let us define

$$\varphi(n) = |\{k \mid 1 \leq k \leq n, \gcd(k, n) = 1\}| \ . \tag{4}$$

Euler's totient function

The function φ is called *Euler's totient function* (or Euler's φ-function). The answer to our question before is "$\varphi(n)$". From the definition we can directly compute $\varphi(1) = \varphi(2) = 1$, $\varphi(3) = \varphi(4) = 2$, $\varphi(5) = 4$, $\varphi(6) = 2$, etc. Note that if $n \geq 2$ then the condition "$k \leq n$" in (4) may be replaced with "$k < n$" because $\gcd(n, n) = n$.

Exercise 6. *Tabulate the values of φ up to $n = 20$.*

Exercise 7. *Show that if p is a prime number, then $\varphi(p) = p - 1$.*

Before considering how to actually compute $\varphi(n)$ without verifying the mutual primality of n and all smaller numbers one by one, let us state the most important result (for cryptographic constructions) that contains this function.

Theorem 3 (Euler)**.** *If* $\gcd(a, n) = 1$*, then* $a^{\varphi(n)} \equiv 1 \pmod{n}$.

Proof. This theorem is actually a simple corollary of Lagrange's theorem about the orders of subgroups that we saw in the previous study unit. The cardinality of $\mathbb{Z}_n^*$ is $\varphi(n)$. The condition $\gcd(a, n) = 1$ means $a \in \mathbb{Z}_n^*$. Let o be the order of a in $\mathbb{Z}_n^*$. Lagrange's theorem states $o \mid \varphi(n)$. Let $k = \varphi(n)/o$. Thus

$$a^{\varphi(n)} = a^{ok} = (a^o)^k \equiv 1^k = 1 \pmod{n} \ .$$

□

Proof. Let us also give a second proof to Euler's theorem; this time without using Lagrange's theorem. We have $a \in \mathbb{Z}_n^*$. Consider the set

$$X = \{a \cdot x \mid x \in \mathbb{Z}_n^*\} \ .$$

We have $X \subseteq \mathbb{Z}_n^*$ because if x is invertible in $\mathbb{Z}_n$, then so is ax as a product of two invertible elements. We now show that X is actually equal to $\mathbb{Z}_n^*$. For this, it is sufficient to show that if $x_1 \neq x_2$ then $a \cdot x_1 \neq a \cdot x_2$. If this were not the case then we would have $a(x_1 - x_2) = 0$ in $\mathbb{Z}_n$. Multiplying both sides with a^{-1} (*modulo* n) gives us $x_1 - x_2 = 0 \cdot a^{-1} = 0$ and $x_1 = x_2$.

In $\mathbb{Z}_n$, we now have

$$\prod_{x \in \mathbb{Z}_n^*} x \overset{(*)}{=} \prod_{x \in X} x \overset{(**)}{=} \prod_{x \in \mathbb{Z}_n^*} ax = \left(\prod_{x \in \mathbb{Z}_n^*} a \right) \cdot \left(\prod_{x \in \mathbb{Z}_n^*} x \right) \overset{(***)}{=} a^{\varphi(n)} \prod_{x \in \mathbb{Z}_n^*} x,$$

where $(*)$ follows from the equality of $\mathbb{Z}_n^*$ and X, $(**)$ follows from the definition of X above, and $(***)$ follows from $|\mathbb{Z}_n^*| = \varphi(n)$. The quantity $\left(\prod_{x \in \mathbb{Z}_n^*} x\right)$ is invertible in $\mathbb{Z}_n$ because it is a product of invertible elements. Thus we can cancel it out from this equality and obtain $1 = a^{\varphi(n)}$ in $\mathbb{Z}_n$. □

While the most important conclusions of this result state that certain constructions (e.g. RSA encryption and signatures) work, it is also applicable to more practical tasks. For example, let us compute $5^{75} \bmod 19$. We have $\varphi(19) = 18$. Euler's theorem implies $5^{18} \equiv 1 \pmod{19}$. Thus

$$5^{75} = 5^{4 \cdot 18 + 3} = (5^{18})^4 \cdot 5^3 \equiv 1^4 \cdot 5^3 = 5^3 = 25 \cdot 5 \equiv 6 \cdot 5 = 30 \equiv 11 \pmod{19} \ .$$

Exercise 8. *Compute* $7^{46} \bmod 18$ *and* $6^{138} \bmod 16$.

From Euler's theorem, and from the values of φ at the points where we already know how to compute it, we obtain the following simple corollary:

Theorem 4 (Fermat's little theorem)**.** *If p is a prime number and $p \nmid a$, then $a^{p-1} \equiv 1 \pmod{p}$.*

Proof. Follows from Euler's theorem. If p is prime, then $\varphi(p) = p - 1$. If p is prime then $p \nmid a$ implies $a \perp p$. □

In the following we'll see how to compute $\varphi(n)$ for any $n \in \mathbb{N}$. However, in order to perform this computation, we must know how n can be expressed as a product of powers of prime numbers: let $p_1, \ldots, p_k$ be different prime numbers, and $e_1, \ldots, e_k \in \mathbb{N}$, such that

$$n = \prod_{i=1}^{k} p_i^{e_i} \quad .$$

Lemma 1. *If n is a* prime power*, i.e. $n = p^e$, then $\varphi(n) = \left(1 - \frac{1}{p}\right) \cdot n = p^{e-1}(p-1)$.*

Proof. The only prime number that divides n is p. Hence the only numbers that do not belong to the set mentioned in (4) are those that are divisible by p. These numbers are $p, 2p, 3p, \ldots, p^{e-1}p = n$. There is a total of n/p numbers between 1 and n that are divisible by p. The rest belong to the set and contribute to $\varphi(n)$. □

Lemma 2. *If $m \perp n$, then $\varphi(mn) = \varphi(m)\varphi(n)$.*

Proof. This lemma claims that if $m \perp n$, then $|\mathbb{Z}_{mn}^*| = |\mathbb{Z}_m^*| \cdot |\mathbb{Z}_n^*|$. Theorem 2 showed that $\mathbb{Z}_{mn} \cong \mathbb{Z}_m \times \mathbb{Z}_n$. If these rings are isomorphic, then they also have the same number of invertible elements — $|\mathbb{Z}_{mn}^*| = |(\mathbb{Z}_m \times \mathbb{Z}_n)^*|$. To prove the claim of the current lemma, it is sufficient to show that if R_1 and R_2 are finite rings, then $|(R_1 \times R_2)^*| = |R_1^*| \cdot |R_2^*|$.

But

$$(R_1 \times R_2)^* = \{(r_1, r_2) \,|\, r_1 \in R_1^*, r_2 \in R_2^*\} \quad .$$

Indeed, if $r_1 \in R_1^*$ and $r_2 \in R_2^*$, then (r_1, r_2) is invertible in $R_1 \times R_2$ — its inverse is (r_1^{-1}, r_2^{-1}). Similarly, if $(r_1, r_2) \in (R_1 \times R_2)^*$, then it has an inverse (r_1', r_2'). Then r_i' is the inverse of r_i in R_i and hence $r_i \in R_i^*$. Thus the cardinality of $(R_1 \times R_2)^*$ is equal to the product of cardinalities of R_1^* and R_2^*. □

The previous two lemmas allow us to compute $\varphi(n)$ for any n. The next theorem states the result of this computation.

Theorem 5. *If $n = \prod_{i=1}^{k} p_i^{e_i}$, where p_i are different prime numbers and $e_i \in \mathbb{N}$, then*

$$\varphi(n) = \prod_{i=1}^{k} p_i^{e_i - 1}(p_i - 1) = n \cdot \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right) \quad .$$

Proof. Immediate from the previous two lemmas:

$$\varphi(n) = \varphi\left(\prod_{i=1}^{k} p_i^{e_i}\right) \overset{(*)}{=} \prod_{i=1}^{k} \varphi(p_i^{e_i}) \overset{(**)}{=} \prod_{i=1}^{k} p_i^{e_i-1}(p_i - 1) =$$

$$\prod_{i=1}^{k} p_i^{e_i} \frac{p_i - 1}{p_i} = \prod_{i=1}^{k} p_i^{e_i} \cdot \prod_{i=1}^{k} \frac{p_i - 1}{p_i} \overset{(***)}{=} n \cdot \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right) \quad .$$

Here $(*)$ follows from Lemma 2 and $(**)$ follows from Lemma 1. The equality $(***)$ merely makes use of the definition of p_i and e_i as the prime decomposition of n. □

Exercise 9. *Compute $\varphi(n)$ for various values of n. E.g. $n = 21$, $n = 98$, $n = 360$, $n = 100$.*

Exercise 10. *Perform modular exponentiations. E.g. compute* $19^{242} \bmod 100$.

4 Joint applications of CRT and φ

In the RSA cryptosystem, the owner of the secret key has to rise the values it receives to the power d *modulo* $n = pq$, where p and q are large prime numbers. As p and q may be known to the owner of the secret key, this task can be simplified as follows.

Let m be the value, for which $c = m^d \bmod n$ has to be computed. To do this, we may compute $c_p = m^d \bmod p$ and $c_q = m^d \bmod q$, and then use CRT to find c. The value of d may theoretically range between 1 and n; in general, it can be much larger than p or q. Fermat's little theorem implies that $c_p = m^{d_p} \bmod p$ and $c_q = m^{d_q} \bmod q$, where $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$. The exponents d_p and d_q are smaller than d and can also be precomputed.

Exercise 11. *Find* $85^{2976} \bmod 8453$. *Hint:* $8453 = 79 \cdot 107$.