ELEN 162
A. I. Zecevic

Lecture Notes for Week 6

Number Theory and Encryption

One of the most important applications of quantum computing has to do with the encryption and decryption of messages. Most methods that are currently used for this purpose rely on "keys" which allow the sender and receiver to exchange information in a way that is both reliable and secure. A key can be thought of as an invertible mapping F, which transforms messages that are represented in numerical form. If two individuals share this mapping, one of them can send message x to the other by encoding it as $y = F(x)$, and the receiver can then easily decode it as $x = F^{-1}(y)$. Since the key is known only to the sender and the receiver, this type of encryption is commonly referred to as *non-public-key encryption*.

Although function F can be chosen in many different ways, non-public-key encryption is quite straightforward (at least, in theory). In practice, however, it faces certain difficulties that can not be easily resolved. One of them stems from the fact that keys need to be changed periodically, since there is always a chance that they can be deduced by a third party given a sufficient amount of time.

An even more critical problem is how to manage keys when a large number of people is involved. To see why this poses a challenge, consider a scenario in which n individuals need to exchange encrypted messages on a regular basis. If each pair has their own "private" key, then the number of different keys that are needed grows rapidly as n increases.

We can show this explicitly if we number the individuals as $p_1, p_2, \ldots, p_n$, and represent the key sharing scheme in the form of a matrix

$$K = \begin{bmatrix} 0 & k_{12} & k_{13} & \ldots & k_{1,n-1} & k_{1n} \\ 0 & 0 & k_{23} & \ldots & k_{2,n-1} & k_{2n} \\ 0 & 0 & 0 & \ldots & k_{3,n-1} & k_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 & k_{n-1,n} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{1}$$

in which the key shared by individuals p_i and p_j is denoted by k_{ij}. Since we are interested only in establishing the number of *different* keys that are needed, it will suffice to record only *one* number for each pair (which explains why matrix K is upper triangular). We should also set all the diagonal elements to zero, since individuals don't need to exchange keys with themselves.

Given that an $n \times n$ matrix consists of n^2 elements and that n of these elements are on the diagonal, it is not difficult to see that in this case

$$\frac{n^2 - n}{2} = \frac{n(n-1)}{2} \tag{2}$$

of them are nonzero. Note that this number can be quite large in practical situations - if $n = 100$, for example, it becomes necessary to maintain (and regularly update) as many as

4,950 separate keys. It is therefore fair to say that this process is often cumbersome, and that managing it involves significant overhead.

The difficulties associated with non-public-key encryption motivated the development of a different paradigm, which is considerably more efficient. This paradigm (which is known as *public-key encryption*) is based on the idea that the keys for encrypting and decrypting a message needn't necessarily be the same. The first such algorithm was the so-called RSA encryption scheme, which was named after the three mathematicians who developed it (Ronald Rivest, Adi Shamir and Leonard Adelman). In this scheme, the receiver of the message produces two different keys - a *public* one that is shared, and a *private* one that is not. The generation of both keys relies on *prime factorization*, and the fact that this mathematical operation is virtually impossible to execute when the number is large.

A Brief Overview of Number Theory

In order to understand how public-key encryption works, it will be necessary to review some fundamental results from number theory (particularly those that pertain to prime factorization and modular arithmetic). In this section we provide a brief summary of these results, starting with the following definition.

Definition 1. The term "integer" refers to any whole number, positive or negative (including zero). A positive integer p is said to be a *prime number* if it is divisible only by 1 and itself. A *composite number* n, on the other hand, is an integer that has at least one divisor other than 1 and n.

The so-called "Fundamental Theorem of Arithmetic" claims that any integer $a > 1$ can be *uniquely* represented as

$$a = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \ldots \cdot p_m^{\alpha_m} \tag{3}$$

where $p_1 < p_2 < \ldots < p_m$ are distinct prime numbers greater than 1, and $\alpha_i \geq 1$ $(i = 1, 2, \ldots, m)$. Expression (3) is known as the *prime factorization of* a, and finding integers $\{p_1, p_2, \ldots, p_m\}$ efficiently is one of the most important problems in number theory. Until very recently, it was believed that this problem cannot be solved when a is a large number. It turns out, however, that quantum computers can actually accomplish this task in a reasonable amount of time. This remarkable development could have major repercussions, since it makes standard encryption techniques potentially unsafe.

To explain how quantum algorithms handle prime factorization, we first need to briefly describe how modular arithmetic works, and how it relates to RSA encryption.

Modular Arithmetic

Modular arithmetic is concerned with the remainders that are obtained when we divide two integers. The following lemma provides the basis for many of the results that will be described in this section (for the sake of clarity, we will be using the same numbering for lemmas and theorems as in the textbook).

Lemma 5.4. Let a and N be integers, and assume that $N > 0$. Then, a has a *unique* representation of the form

$$a = kN + r \tag{4}$$

where k is an integer (which could be positive, negative or zero), and $0 \leq r < N$.

Lemma 5.4 allows us to precisely define the remainder that is obtained when a is divided by N. This number, which is usually described as

$$r = a \bmod N \tag{5}$$

can be specified uniquely because the remainder is always assumed to be a non-negative number that is *smaller* than N.

To illustrate why this assumption is important, consider how one might interpret a number like 12 mod 7. In principle, we could express 12 as

$$12 = 2 \cdot 7 - 2 \tag{6}$$

$$12 = 1 \cdot 7 + 5 \tag{7}$$

$$12 = 0 \cdot 7 + 12 \tag{8}$$

$$12 = -1 \cdot 7 + 19 \tag{9}$$

and so on, but only *one* of these representations has a remainder that satisfies $0 \leq r < 7$. As a result, we can say that $12 \bmod 7 = 5$ without any ambiguity.

In modular arithmetic, it is common to encounter expressions such as

$$a \bmod N = b \bmod N \tag{10}$$

and

$$a = b \pmod N \tag{11}$$

These two expressions are actually equivalent, and both of them tell us that dividing a and b by N produces the *same* remainder. The following lemma provides an alternative way to describe this relationship.

Lemma 5.5. Let a and b be two arbitrary integers. Then, $a - b$ is divisible by N if and only if dividing a and b by N produces the same remainder.

Remark 1. Note that using (mod N) in parenthesis implies that mod N applies to *both* sides of the equality. If we are just referring to the remainder that is obtained when a is divided by N, we do *not* use parenthesis (we express this number as $a \bmod N$).

Example 1. Let $a = 32$ and $b = -10$. Since

$$32 = 5 \cdot 6 + 2 \tag{12}$$

and

$$-10 = -2 \cdot 6 + 2 \tag{13}$$

it follows that that these two numbers have the *same* remainder when divided by 6. We can therefore conclude that

$$32 \bmod 6 = -10 \bmod 6 = 2 \tag{14}$$

Given the convention that we introduced in Remark 1, we can equivalently represent expression (14) as

$$32 = -10 \pmod 6 \tag{15}$$

It is easily verified that Lemma 5.5 applies in this case - all we need to do is recognize that

$$a - b = 42 \tag{16}$$

is divisible by 6.

Modular arithmetic is quite straightforward when it comes to addition, subtraction and multiplication. It is not difficult to see, for example, that

$$(2 + 7) \bmod 4 = 1 \tag{17}$$

since

$$9 = 2 \cdot 4 + 1 \tag{18}$$

We could equivalently express this relationship as

$$2 + 7 = 1 \pmod 4 \tag{19}$$

since 9 and 1 have the same remainder when divided by 4.

We can easily extend this logic to subtraction and multiplication. Typical examples would be identities such as

$$(1 - 8) \bmod 5 = 3 \tag{20}$$

and

$$(5 \cdot 7) \bmod 3 = 2 \tag{21}$$

which follow from the fact that

$$-7 = -2 \cdot 5 + 3 \tag{22}$$

and

$$35 = 11 \cdot 3 + 2 \tag{23}$$

respectively.

Division is considerably more complicated, however, because b/a needn't be an integer (in which case the expression $(b/a) \bmod N$ makes no sense). In order to resolve this problem, it will be necessary to introduce the notion of a *modular multiplicative inverse*.

To explain what this term means, suppose that a and N are a pair of integers, and that there exists an integer ω such that

$$\omega \cdot a = 1 \pmod N \tag{24}$$

This is equivalent to saying that $\omega \cdot a$ and 1 have the *same* remainder when divided by N.

What is this remainder? Given that

$$1 = 0 \cdot N + 1 \tag{25}$$

when $N > 1$, it follows that $1 \bmod N = 1$ in all such cases. As a result, we can say that ω satisfies

$$\omega \cdot a = kN + 1 \tag{26}$$

where k is an integer whenever $N > 1$. In the following, we will refer to ω as the multiplicative inverse of a modulo N, and will denote this number by a_N^{-1}.

Remark 2. Although the multiplicative inverse is often denoted as a^{-1} in the literature, it should not be confused with the ordinary inverse, since a_N^{-1} is an *integer* by definition. It also satisfies

$$a_N^{-1}\, a = kN + 1 \tag{27}$$

which means that $a_N^{-1}\, a \neq 1$ whenever $k \neq 0$.

How does introducing a_N^{-1} help us define modular division? To see this a bit more clearly, let us assume that expression

$$x = (b/a) \bmod N \tag{28}$$

is interpreted as

$$x = a_N^{-1}\, b \bmod N \tag{29}$$

It is not difficult to show that if x is computed in this manner, it will also satisfy

$$ax = b \pmod N \tag{30}$$

Because this looks very much like standard division (where $x = b/a$ implies $ax = b$ and vice versa), it makes sense to think of expressions (28) and (29) as equivalent statements.

Does a_N^{-1} always exist, and is it unique? We will first show that our definition of a_N^{-1} *does not specify this number uniquely*, and will deal with the existence problem after that.

To see why a_N^{-1} is not unique, let us consider the multiplicative inverse of 2 modulo 5. Since

$$3 \cdot 2 = 1 \cdot 5 + 1 \tag{31}$$

we can legitimately claim that $2_5^{-1} = 3$. However, -7 and 13 can be identified as 2_5^{-1} as well, because

$$-7 \cdot 2 = -3 \cdot 5 + 1 \tag{32}$$

and

$$13 \cdot 2 = 5 \cdot 5 + 1 \tag{33}$$

It turns out that there are actually infinitely many such numbers, which suggests that a_N^{-1} should be viewed as a *class of integers*. If we denote the members of this class as $\{\omega_1, \omega_2, \ldots\}$, it is easy to show that any two of them must satisfy

$$\omega_i = \omega_j \pmod N \tag{34}$$

The following example shows that the nonuniqueness of the multiplicative inverse does not cause problems with modular division.

Example 2. Suppose that we want to compute (6/5) mod 7. In order to do that, we should first observe that 3 is one of the possible values for 5_7^{-1}, since

$$3 \cdot 5 = 2 \cdot 7 + 1 \tag{35}$$

Using expression (29), we can now calculate (6/5) mod 7 as

$$(6/5) \bmod 7 = (5_7^{-1} \cdot 6) \bmod 7 = 18 \bmod 7 = 4 \tag{36}$$

since

$$18 = 2 \cdot 7 + 4 \tag{37}$$

Note that the result wouldn't change if we picked a different value for 5_7^{-1}. We could replace 5_7^{-1} in equation (36) with -4, for example, since

$$-4 \cdot 5 = -3 \cdot 7 + 1 \tag{38}$$

in which case we obtain

$$(6/5) \bmod 7 = (5_7^{-1} \cdot 6) \bmod 7 = -24 \bmod 7 = 4 \tag{39}$$

This tells us that *modular division is uniquely defined*, despite the fact that a_N^{-1} can take an unlimited number of values.

It is important to keep in mind in this context that a multiplicative inverse needn't exist for every choice of N, so modular division must be handled with care. In order to explain why this is so, we first need to introduce the notion of the *greatest common divisor*, and discuss some of its properties.

Definition 2. Given two positive integers a and b, their greatest common divisor (denoted $\gcd(a, b)$) represents the *largest* integer that divides both of them. If $\gcd(a, b) = 1$, we say that these two numbers are *co-prime*.

Remark 3. Note that $\gcd(a, b)$ satisfies $\gcd(a, b) \geq 1$, regardless of how a and b are chosen. This is because 1 is a common divisor for *any* pair of integers.

The following two theorems provide necessary and sufficient conditions for the existence of a_N^{-1}, and will help us develop a method for computing this number.

Theorem 5.4. Let a and N be integers, and suppose that $N > 1$. Then, a_N^{-1} exists if and only if $\gcd(a, N) = 1$.

Theorem 5.3. Suppose that a and b are two arbitrary integers, and let $m = \gcd(a, b)$. Then, m represents the *smallest positive integer* that can be expressed as

$$m = ax + by \tag{40}$$

where x and y are integers (not necessarily positive).

Euclid's Algorithm

Since finding the greatest common divisor of a pair of integers is an important step in determining the prime factors of a composite number, we need to examine how this can be done. A simple and efficient procedure for computing the gcd is known as Euclid's algorithm (which we will use extensively in the next few lectures). To understand how this algorithm works, we will need the following theorem.

Theorem 5.5. Let a and b be two arbitrary integers, and let r be the remainder when a is divided by b. If $r > 0$, then

$$\gcd(a, b) = \gcd(b, r) \tag{41}$$

To see why this result is helpful, let us assume (without any loss of generality) that $a > b$. If we divide a by b, we will obtain an expression of the form

$$a = k_1 b + r_1 \tag{42}$$

where $0 \leq r_1 < b$. If $r_1 > 0$, Theorem 5.5 ensures that

$$\gcd(a, b) = \gcd(b, r_1) \tag{43}$$

Since $r_1 > 0$, our next step will be to divide b by r_1, which produces

$$b = k_2 r_1 + r_2 \tag{44}$$

where $0 \leq r_2 < r_1$. If $r_2 > 0$, we have that

$$\gcd(b, r_1) = \gcd(r_1, r_2) \tag{45}$$

and we can divide r_1 by r_2. Once we do so, we can rewrite r_1 as

$$r_1 = k_3 r_2 + r_3 \tag{46}$$

where $0 \leq r_3 < r_2$.

How long should this process continue? For our purposes, it makes sense to end it when the remainder becomes zero. This is bound to happen after a finite number of steps, since $\{r_1, r_2, \ldots\}$ is a strictly decreasing sequence and $r_1 < b$ by definition.

To see why such a termination criterion is desirable, let us assume that we obtained $r_m = 0$ after m steps. This implies that:

1. $0 < r_{m-1} < r_{m-2} < \ldots < r_1 < b$ after step $m - 1$
2. In step m, we have
$$r_{m-2} = k_m r_{m-1} + r_m = k_m r_{m-1} \tag{47}$$

Since remainders $\{r_1, r_2, \ldots, r_{m-1}\}$ are all positive, we know that

$$\gcd(a, b) = \gcd(b, r_1) = \gcd(r_1, r_2) = \ldots = \gcd(r_{m-2}, r_{m-1}) \tag{48}$$

(by virtue of Theorem 5.5). Equation (47) additionally implies that

$$\gcd(r_{m-2}, r_{m-1}) = r_{m-1} \tag{49}$$

since r_{m-2} is divisible by r_{m-1}, and $r_{m-2} > r_{m-1}$. Combining (48) and (49), we can now conclude that

$$\gcd(a, b) = r_{m-1} \tag{50}$$

which means that the *smallest nonzero remainder* produced by this procedure represents the greatest common divisor of a and b.

The result that we just obtained represents the theoretical basis for Euclid's algorithm. The following example illustrates how this algorithm works in practice.

Example 3. Suppose that we want to find the greatest common divisor of $a = 18,445$ and $b = 2,805$. In order to do that, we need to perform the following sequence of operations:

$$a = k_1 b + r_1 \implies 18,445 = 6 \times 2,805 + 1,615 \implies k_1 = 6 \quad r_1 = 1,615 \tag{51}$$

$$b = k_2 r_1 + r_2 \implies 2,805 = 1 \times 1,615 + 1,190 \implies k_2 = 1 \quad r_2 = 1,190 \tag{52}$$

$$r_1 = k_3 r_2 + r_3 \implies 1,615 = 1 \times 1,190 + 425 \implies k_3 = 1 \quad r_3 = 425 \tag{53}$$

$$r_2 = k_4 r_3 + r_4 \implies 1,190 = 2 \times 425 + 340 \implies k_4 = 2 \quad r_4 = 340 \tag{54}$$

$$r_3 = k_5 r_4 + r_5 \implies 425 = 1 \times 340 + 85 \implies k_5 = 1 \quad r_5 = 85 \tag{55}$$

$$r_4 = k_6 r_5 + r_6 \implies 340 = 4 \times 85 + 0 \implies k_6 = 4 \quad r_6 = 0 \tag{56}$$

Since the smallest nonzero remainder is $r_5 = 85$, it follows that the greatest common divisor of $18,445$ and $2,805$ is 85.

Computing a Multiplicative Inverse

Euclid's algorithm also provides us with a systematic way to compute multiplicative inverses. This approach makes use of the fact that a_b^{-1} exists if only if a and b are co-prime, in which case the algorithm produces $\gcd(a, b) = 1$. Under such circumstances, Theorem 5.3 ensures that a and b are related as

$$1 = ax + by \tag{57}$$

where x and y are integers. Since expression (57) is equivalent to

$$xa = (-y)b + 1 \tag{58}$$

it follows that x is one of the integers that correspond to a_b^{-1}.

Example 4 illustrates how this idea can be combined with Euclid's algorithm to produce multiplicative inverses. In order to identify all such numbers, we will make use of the following lemma.

Lemma 5.7. If w is a multiplicative inverse of a modulo N, then all other numbers with this property must have the form

$$z = w + mN \tag{59}$$

where m is an integer.

Example 4. Suppose that we want to compute $1,964_{115}^{-1}$. In order to do that, we must first establish whether such a number actually exists. According to Theorem 5.4, this requires checking if $\gcd(1,964, 115) = 1$.

If we apply Euclid's algorithm to $a = 1,964$ and $b = 115$, we obtain

$$a = k_1 b + r_1 \implies 1,964 = 17 \cdot 115 + 9 \implies k_1 = 17 \quad r_1 = 9 \tag{60}$$

$$b = k_2 r_1 + r_2 \implies 115 = 12 \cdot 9 + 7 \implies k_2 = 12 \quad r_2 = 7 \tag{61}$$

$$r_1 = k_3 r_2 + r_3 \implies 9 = 1 \cdot 7 + 2 \implies k_3 = 1 \quad r_3 = 2 \tag{62}$$

$$r_2 = k_4 r_3 + r_4 \implies 7 = 3 \cdot 2 + 1 \implies k_4 = 3 \quad r_4 = 1 \tag{63}$$

$$r_3 = k_5 r_4 + r_5 \quad \Longrightarrow \quad 2 = 2 \cdot 1 + 0 \quad \Longrightarrow \quad k_5 = 2 \quad r_5 = 0 \tag{64}$$

Since the smallest nonzero remainder equals 1, we can conclude that $\gcd(1,964,115) = 1$, and that $1,964^{-1}_{115}$ exists.

To compute $1,964^{-1}_{115}$, we now need to apply Euclid's algorithm "in reverse". This entails the following sequence of steps:

$$\begin{aligned} 1 &= r_4 = r_2 - k_4 r_3 = r_2 - 3r_3 = r_2 - 3(r_1 - k_3 r_2) = \\ &= r_2 - 3r_1 + 3r_2 = 4r_2 - 3r_1 = 4(b - k_2 r_1) - 3r_1 = \\ &= 4(b - 12r_1) - 3r_1 = 4b - 51r_1 = 4b - 51(a - k_1 b) = \\ &= 4b - 51(a - 17b) = -51a + 871b \end{aligned} \tag{65}$$

Recalling that $a = 1,964$ and $b = 115$, (65) can be rewritten as

$$1 = -51 \times 1,964 + 871 \times 115 \tag{66}$$

which is equivalent to

$$-51 \times 1,964 = -871 \times 115 + 1 \tag{67}$$

This implies that -51 is one of the possible values for $1,964^{-1}_{115}$.

Lemma 5.7 allows us to identify *all* such numbers, since we know that they satisfy

$$y = -51 + m \cdot 115 \tag{68}$$

(where m is an integer). We can therefore conclude that the multiplicative inverse of 1,964 modulo 115 corresponds to set

$$\{\ldots - 166, -51, 64, 179, \ldots\} \tag{69}$$

As noted in (34), any two numbers y_i and y_j that belong to this set satisfy $y_i = y_j \pmod{115}$.

The RSA Encryption Algorithm

The basic idea behind the RSA algorithm is for the receiver (who is traditionally referred to as Alice in the literature) to choose two large prime numbers, p and q, and form their product $n = pq$. She then computes

$$\phi(n) = (p-1)(q-1) \tag{70}$$

and picks a number b such that $1 < b < \phi(n)$, and $\gcd(b, \phi(n)) = 1$.

Both n and b are made available to anyone who wants to send Alice a message, which is why these two numbers are referred to as Alice's "public key". Her "private key" consists of p and q, as well as a number a which satisfies

$$ab = 1 \pmod{\phi(n)} \tag{71}$$

This number is not difficult to compute, since it corresponds to the multiplicative inverse of b modulo $\phi(n)$ (we saw that Euclid's algorithm can do this very efficiently).

Suppose now that Bob wants to send a message to Alice, which is represented by number x (this number is always chosen so that $1 < x < n$). Since n and b are publicly available, Bob can encrypt the message as

$$y = x^b \bmod n \tag{72}$$

and send it to her. It is not difficult to show that Alice can then recover x by computing

$$x = y^a \bmod n \tag{73}$$

once she receives y (see Theorem 5.9 in the textbook).

What makes this exchange secure is the fact that integer a is a part of Alice's *private key*. No one else has access to this information, since computing a requires knowing $\phi(n)$, and therefore p and q as well. As we already noted, these two numbers are practically impossible to deduce from n, since they represent its prime factors. If this were not the case, virtually anyone who intercepted the encoded message y could determine the original message x.

Although the RSA algorithm appears to be quite straightforward, it poses some nontrivial computational challenges. Perhaps the most critical one relates to the calculation of integers $x^b \bmod n$ and $y^a \bmod n$, since both x^b and y^a can be very large numbers in general. The following two lemmas will help us simplify this task.

Lemma 5.10. Let k, m and n be positive integers. Then,

$$a^{km} \bmod n = [\, a^k \bmod n \,]^m \bmod n \tag{74}$$

Lemma 5.11. Suppose that

$$w = w_1 \cdot w_2 \cdot \ldots \cdot w_m \tag{75}$$

where w_i $(i = 1, 2, \ldots, m)$ are positive integers. Then,

$$w \bmod n = [(w_1 \bmod n) \cdot (w_2 \bmod n) \cdot \ldots \cdot (w_m \bmod n)] \bmod n \tag{76}$$

We will now illustrate how these two results can be used for encryption and/or decryption purposes.

Example 5. Suppose that we want to compute

$$18^{37} \bmod 77 \tag{77}$$

as part of the RSA encryption process. In order to do that, we will first rewrite the exponent using powers of 2, which produces

$$18^{37} = (18)^{32} \cdot (18)^4 \cdot (18)^1 \tag{78}$$

A good starting point for the computation is 18^4, since this number is not too large. Given that

$$18^4 = 104,976 = 1,363 \times 77 + 25 \tag{79}$$

it follows that

$$18^4 \bmod 77 = 25 \tag{80}$$

We now make use of Lemma 5.10 to compute 18^{32}. This procedure will require two steps - we will first find 18^{16}, and then 18^{32}.

STEP 1. By virtue of Lemma 5.10, we know that

$$18^{16} \bmod 77 = [\, 18^4 \bmod 77 \,]^4 \bmod 77 = 25^4 \bmod 77 \tag{81}$$

Observing that

$$25^4 = 390,625 = 5,073 \times 77 + 4 \tag{82}$$

it follows that

$$18^{16} \bmod 77 = 4 \tag{83}$$

STEP 2. Using expression (83) in conjunction with Lemma 5.10, we have

$$18^{32} \bmod 77 = [\, 18^{16} \bmod 77 \,]^2 \bmod 77 = 4^2 \bmod 77 \tag{84}$$

Since

$$16 = 0 \cdot 77 + 16 \tag{85}$$

it follows that

$$18^{32} \bmod 77 = 16 \tag{86}$$

Now that we have all the powers of 18 that we need, we can invoke Lemma 5.11, which allows us to express $18^{37} \bmod 77$ as

$$\begin{aligned} &18^{37} \bmod 77 = [\, (18^{32} \bmod 77) \cdot (18^4 \bmod 77) \cdot (18^1 \bmod 77) \,] \bmod 77 = \\ &(16 \cdot 25 \cdot 18) \bmod 77 = 7,200 \bmod 77 \end{aligned} \tag{87}$$

Observing that

$$7,200 = 93 \times 77 + 39 \tag{88}$$

we finally obtain

$$18^{37} \bmod 77 = 39 \tag{89}$$

Example 6. Suppose that Alice forms her private key by choosing $p = 7$ and $q = 11$, which produces $n = pq = 77$ and $\phi(n) = (p-1)(q-1) = 60$. She then needs to pick a number b such that $1 < b < 60$ and $\gcd(b, 60) = 1$. The easiest way to do this would be to consider the set of prime numbers that are smaller than 60. Since any number x in this set has only two divisors, 1 and x, we can guarantee that $\gcd(x, 60) = 1$ if x is not a divisor of 60.

In the following, we will assume that Alice picked $b = 37$ (for no particular reason). Since 60 is not divisible by 37, she can adopt $(n, b) = (77, 37)$ as her public key.

Because a needs to be chosen so that it satisfies

$$ab = 1 \pmod{\phi(n)} \tag{90}$$

Alice must now compute the multiplicative inverse of b modulo $\phi(n)$. She can do so by applying Euclid's algorithm, which produces $a = 13$ (this is one of the infinitely many values that correspond to 37_{60}^{-1}). With this final piece of information in hand, Alice's private key becomes $(p, q, a) = (7, 11, 13)$.

Let us now assume that Bob wants to send message $x = 18$ to Alice. According to the RSA algorithm, he needs to encrypt this message as

$$y = x^b \bmod n = 18^{37} \bmod 77 = 39 \tag{91}$$

(as shown in Example 5). Since $1 < x < n$, Alice can decrypt the message using the fact that

$$x = y^a \bmod n = 39^{13} \bmod 77 = 18 \tag{92}$$

Remark 4. We won't go through the calculation of x at this point, since it follows the same procedure as the one described in Example 5. However, a derivation is available in the textbook (see Example 5.6).