Discrete Mathematics
Number theory

Saad Mneimneh

1 Divisibility and primes

The focus of this entire note is on positive integers. I will start by the basic notion of divisibility. We say that a divides b, or a is a divisor of b, or b is a multiple of a, if there exists an integer m such that

$$b = am$$

We also denote this by $a \mid b$ (a divides b). If a is not a divisor of b, the we write $a \not| b$, but we can still talk about division by a with a remainder r. Given any two integers a and b, there is a unique way to write (q and r are integers)

$$b = aq + r$$

where $0 \leq r < a$. The proof is by contradiction. Assume $b = aq_1 + r_1 = aq_2 + r_2$ and, without loss of generality, let $r_2 > r_1$. Then $0 < r_2 - r_1 < a$. But $r_2 - r_1 = b - aq_2 - (b - aq_1) = a(q_1 - q_2)$. Therefore,

$$0 < a(q_1 - q_2) < a$$

which is impossible since $q_1 - q_2$ is an integer.

The following relation between divisibility and remainders will prove to be useful later on. Let r be the remainder of the division of b by a i.e. $b = aq + r$ ($0 \leq r < a$), then

$$d \mid b \text{ and } d \mid a \Leftrightarrow d \mid a \text{ and } d \mid r$$

We have to prove both directions. If $d \mid b$ and $d \mid a$, then $b = dm_1$ and $a = dm_2$. Therefore, $r = b - aq = dm_1 - dm_2q = d(m_1 - m_2q) \Rightarrow d \mid r$. Conversely if $d \mid r$ and $d \mid a$, then $r = dm_3$ (and $a = dm_2$). Therefore, $b = aq + r = dm_2q + dm_3 = d(m_2q + m_3) \Rightarrow d \mid b$.

An integer $p > 1$ is called prime if it is not divisible by any integer d such that $1 < d < p$. In other words, p is prime if it is divisible by only 1 and p (we are excluding negative integers here). An integer $n > 1$ that is not prime is composite.

2 The Euclidean algorithm

Consider two positive integers $a_0 \geq a_1$. The greatest common divisor of a_0 and a_1, denoted $\gcd(a_0, a_1)$ is the largest integer g such that $g \mid a_0$ and $g \mid a_1$, i.e. g is the largest integer that divides both a_0 and a_1.

The first observation we can make is that $\gcd(a_0, a_1)$ always exists because $1 \mid a_0$ and $1 \mid a_1$. The second observation we can make is that if $a_0 = a_1 q_1 + r_1$, then (from above)

$$g \mid a_0 \text{ and } g \mid a_1 \Leftrightarrow g \mid a_1 \text{ and } g \mid r_1$$

It follows that

$$\gcd(a_0, a_1) = \gcd(a_1, r_1)$$

This suggests a recursive algorithm for finding the greatest common divisor, the Euclidean algorithm. We compute the sequence

$$a_0 \quad a_1 \quad \ldots \quad a_k \quad a_{k+1}$$

where

$$a_i = r_{i-1}, i > 1$$

$$a_{i-2} = a_{i-1} q_{i-1} + r_{i-1}, i > 1$$

$$a_{k+1} = 0$$

The sequence is strictly decreasing and, therefore, $a_{k+1} = 0$ is guaranteed for some k. We can easily show that $\gcd(a_0, a_1) = a_k$, as follows:

$$\gcd(a_0, a_1) = \gcd(a_1, a_2) = \ldots = \gcd(a_{k-1}, a_k)$$

Since $a_{k+1} = r_k = 0$, a_{k-1} is a multiple of a_k and hence $\gcd(a_{k-1}, a_k) = a_k$.

As an example, to compute the $\gcd(300, 18)$, we successively find the remainder of the division of a_i by a_{i+1} to obtain the following sequence.

$$300 \quad 18 \quad 12 \quad 6 \quad 0$$

which reveals that $\gcd(300, 18) = 6$.

3 Why not brute force?

In principle, to find $\gcd(a_0, a_1)$ we can try all integers from a_1 down to 1. The first one that divides both a_0 and a_1 must be the greatest common divisor. So why don't we do it this way and save ourselves the more complicated theory? The reason is efficiency. In the worst case, one would have to try all integers in $\{1, 2, \ldots, a_1\}$. Since a_1 can be represented using $\log a_1$ bits, the running time of such a brute force algorithm is exponential in the size of its input. In comparison, the Euclidean algorithm requires k steps. Let's discover how large k is.

Observe that since $a_{i-2} > a_{i-1}$, $q_{i-1} \geq 1$. Therefore,

$$a_{i-2} = a_i + a_{i-1}q_{i-1} \geq a_i + a_{i-1}$$

In addition, we have

$$a_{k-1} \geq 2 \text{ (otherwise } a_k = 0)$$

$$a_k \geq 1 \text{ (the gcd)}$$

Comparing this to the Fibonacci sequence

$$F_n = F_{n-1} + F_{n-2}$$

$$F_3 = 2$$

$$F_2 = 1$$

we see that a_i is always an upper bound on some Fibonacci number.

F_0	F_1	F_2	F_3	$\ldots$	F_{k+2-i}	$\ldots$	F_{k+2}
0	1	1	2				
	0	a_k	a_{k-1}	$\ldots$	a_i	$\ldots$	a_0

$$F_{k+2-i} \leq a_i$$

$$F_{k+2} \leq a_0$$

We can prove by induction that $F_n \geq c\phi^{n-1}$ for some constant c and, therefore,

$$c\phi^{k+1} \leq a_0$$

$$k \leq \log_\phi \frac{a_0}{c} - 1$$

This shows that the number of steps in the Euclidean algorithm is logarithmic in a_0, which means linear in the number of bits required to represent a_0. The above table also shows that the worst case occurs when $a_i = F_{k+2-i}$ for every $i = 0 \ldots k$, i.e. when a_0 and a_1 are consecutive Fibonacci numbers. Here's an example of the sequence when $a_0 = 13$ and $a_1 = 8$.

$$13 \quad 8 \quad 5 \quad 3 \quad 2 \quad 1 \quad 0$$

4 The extended Euclidean algorithm

We can extend the Euclidean algorithm to compute x_i and y_i such that

$$a_i = a_0 x_i + a_1 y_i$$

In other words, every a_i can be expressed as a linear combination of a_0 and a_1. To see this, we set $x_0 = 1, y_0 = 0$ and $x_1 = 0, y_1 = 1$. This will satisfy the property for a_0 and a_1, the base case. We can then proceed by induction.

$$a_i = a_{i-2} - a_{i-1}q_{i-1} = a_0 x_{i-2} + a_1 y_{i-2} - q_{i-1}(a_0 x_{i-1} + a_1 y_{i-1})$$

$$= a_0(x_{i-2} - q_{i-1}x_{i-1}) + a_1(y_{i-2} - q_{i-1}y_{i-1})$$

which gives the following recurrences for x_i and y_i

$$x_i = x_{i-2} - \frac{a_{i-2} - a_i}{a_{i-1}} x_{i-1}$$

$$y_i - y_{i-2} - \frac{a_{i-2} - a_i}{a_{i-1}} y_{i-1}$$

Now $\gcd(a_0, a_1) = a_k = a_0x_k + a_1y_k$, so the greatest common divisor of a_0 and a_1 can also be expressed as a linear combination of a_0 and a_1.

Given two positive integers a and b, $\gcd(a, b) = ar + bs$ for some two integers r and s. Observe that since $0 < \gcd(a, b) \leq \min(a, b)$ either $r > 0, s \leq 0$ or $r \leq 0, s > 0$. Furthermore, we can increase r and decrease s using the following trick:

$$\gcd(a, b) = ar + bs = a(r + b) + b(s - a)$$

until $r > 0, s \leq 0$. Therefore, we can make

$$\gcd(a, b) = ar - bs$$

where $r > 0, s \geq 0$. Henceforth, we can decrease r and decrease s using a similar trick:

$$ar - bs = a(r - b) - b(s - a)$$

until $r \leq b$. When this happen $\gcd(a, b) \leq ab - bs = b(a - s)$ so $a - s > 0$ and $s < a$. Therefore, we can state the following:

Given two positive integers a and b,

$$\gcd(a, b) = ar - bs = b(a - s) - a(b - r)$$

where $0 < r \leq b$ and $0 \leq s < a$. The Euclidean algorithm finds one of the above two combinations.

5 Co-primes (relatively prime)

Two positive integers are said to be co-primes or relatively prime if their greatest common divisor is 1. For co-primes, we can reverse the direction of the implication of the Euclidean algorithm:

$$\gcd(a, b) = 1 \Leftrightarrow ar - bs = 1 \text{ for some integers } r, s$$

To prove this equivalence, we need to prove both direction of the implication. The first direction is trivial, by the Euclidean algorithm, $\gcd(a, b) = 1 \Rightarrow ar - bs = 1$ for some integers r and s. To prove the second direction, let d be a common divisor of a and b, thus $a = dn$ and $b = dm$ for some integers n, m. If $ar - bs = 1$, then $dnr - dms = d(nr - ms) = 1$. Therefore, d must be 1.

An interesting application is that two co-primes a and b can be combined to generate any integer value. Obviously, since $ar - bs = 1$, then $a(nr) - b(ns) = n$ for every n. This is the trick behind the water juggling puzzle that we talked about before. Which numbers are co-primes? Any two prime numbers are co-primes (obvious). Any two consecutive integers are co-primes (try to prove). Similarly, any two consecutive odd numbers are co-primes (try to prove).

Here's a more interesting property of co-primes. If $\gcd(a, b) = 1$, then every integer $n \geq (a-1)(b-1)$ can be expressed as $ax+by$, where $x, y \geq 0$. Obviously we can set x and y to obtain arbitrarily large numbers. So we will prove the lower bound by considering a number $n = ax + by$ and showing that $n - 1$ can be expressed similarly as long as it greater than or equal to $(a-1)(b-1)$. Note that this is not a proof by induction. It is just a way to figure out the bound beyond which we cannot guarantee that the statement holds.

Let $n = ax + by$. Since $\gcd(a, b) = 1$, we know that $ar - bs = 1$ and $b(a - s) - a(b - r) = 1$ where $0 < r \leq b$ and $0 \leq s < a$. Then

$$n - 1 = a(x - r) + b(y + s)$$

$$n - 1 = a(x + b - r) + b(y - a + s)$$

The statement for $n - 1$ will fail if $x - r < 0$ and $y - a + s < 0$, i.e. if $x \leq r - 1$ and $y \leq a - s - 1$. So the smallest bound that we can guarantee is

$$a(r - 1) + b(a - s - 1) = ar - a + ba - bs - b = 1 - a - b + ba = (a - 1)(b - 1)$$

6 Fundamental theorem of arithmetic

Every integer can be written as a product of primes, we call it prime factorization. This can be easily proved by induction. Our base case is $n = 2$, which is itself prime. Assume that the property holds up to $n - 1$ and let's prove that it remains true for n. Now for every $n > 2$, we make the following case analysis. If n is prime, we are done. If n is composite, then we can find two integers $a < n$ and $b < n$ such that $n = ab$ (otherwise n would be prime). Both a and b can be written as a product of primes, so we are done.

The fact that every integer can be factored into primes is not so deep. What is fascinating is that this factorization is unique (up to the order of the prime factors). This is called the fundamental theorem of arithmetic. The proof of this theorem can be done by contradiction.

Let n be an integer such that:

$$n = p_1 p_2 \dots p_k = q_1 q_2 \dots q_t$$

where p's and $q's$ are prime numbers with $p_1 \neq q_i$ for all $i = 1 \dots t$. Since $p_1 \neq q_1$ and they are both prime, $\gcd(p_1, q_1) = 1$. By the Euclidean algorithm we can write

$$p_1 r - q_1 s = 1$$

If we multiply both sides by $q_2 \dots q_t$ we obtain:

$$(p_1 r - q_1 s) q_2 \dots q_t = q_2 \dots q_t$$

$$p_1 r q_2 \dots q_t - s q_1 q_2 \dots q_t = q_2 \dots q_t$$

$$p_1 r q_2 \dots q_t - s p_1 \dots p_k = q_2 \dots q_t$$

$$p_1 (r q_2 \dots q_t - s p_2 \dots p_k) = q_2 \dots q_t$$

By factoring $(r q_2 \dots q_t - s p_2 \dots p_k)$ into primes, we obtain two prime factorizations for $q_2 \dots q_t < n$, a contradiction because we could have started with the smallest n that admits two (or more) prime factorizations.

7 Some nice properties of primes

Let p be prime.

- $p \mid ab \Rightarrow p \mid a$ or $p \mid b$
- $a \mid b$, $p \mid b$, and $p \not\mid a \Rightarrow p \mid \frac{b}{a}$

Let b/a be a reduced fraction, then

- $\frac{b}{a}$ has a finite decimal representation $\Leftrightarrow a = 2^x 5^y$ where $x, y \geq 0$

All of these properties can be proved using the fundamental theorem of arithmetic. While some of them may seem trivial, none of them holds without the uniqueness of prime factorization. Let's start with the first one. If a prime p divides a product, then it must divide one of the factors. This can be seen by writing:

$$p \mid ab \Leftrightarrow ab = mp$$

By factoring a, b, and m into primes, we obtain two products of primes that are equal. But since prime factorization is unique, p must appear on the left hand side of the equation. Therefore, p must be a prime factor of a or b. Note that this is not true if p is not prime; for instance, $10 \mid 4 \times 5$ but $10 \not\mid 4$ and $10 \not\mid 5$.

On to the second property. Let $b/a = k$, i.e. $b = ak$, then since $p \mid ak$, p must divide a or k. But we know that $p \not\mid a$; therefore, $p \mid k$.

Consider the term $\left(\begin{array}{c} p \\ k \end{array} \right)$, where p is prime and $0 < k < p$. Can you show that $p \mid \left(\begin{array}{c} p \\ k \end{array} \right)$?

Finally, given a fraction b/a we want to show that it has a finite decimal representation if and only if a has exactly two prime factors, 2 and 5.

If b/a has a finite decimal representation, then there exists an $x \geq 0$ such that $b10^x/a = c$ is an integer. Therefore,

$$b10^x = ac$$

By the fundamental theorem of arithmetic, when we factor both sides of the equation, all prime factors of a must appear on the left hand side. But a and b share no prime factors because the fraction is reduced. Therefore, all prime factors of a must be those of $10 = 5 \times 2$. Conversely, if $a = 2^x 5^y$, then b/a can be expressed as

$$\frac{b}{a} = b \underbrace{\frac{1}{2}\frac{1}{2}\cdots\frac{1}{2}}_{x}\underbrace{\frac{1}{5}\frac{1}{5}\cdots\frac{1}{5}}_{y}$$

Each of the terms above has a finite decimal representation, so their product has a finite decimal representation.

8 Distribution of primes

We previously proved that primes are infinite. In this section, we study a little bit their distribution. Primes are quite frequent, although it is possible to find arbitrarily large stretches with no primes. For instance, consider the following k integers:

$$(k+1)!+2\ ,\ (k+1)!+3\ ,\ \ldots\ ,\ (k+1)!+(k+1)$$

None of them can be prime (why?) But if we let $\pi(n)$ be the number of primes less or equal to n, then

$$\pi(n) \sim \frac{n}{\ln n}$$

The approximation is better for large values of n. This is known as the prime number theorem and a proof of it is beyond the scope of the course. The prime number theorem has important implications in computer science. For example, if we pick a random integer in $\{n, n+1, n+2, \ldots, m\}$, what is the probability that it is prime? This is approximately

$$\frac{\pi(m) - \pi(n)}{m-n} \sim \frac{m/\ln m - n/\ln n}{m-n}$$

and if $m = 2n$, this is

$$\frac{2}{\ln 2n} - \frac{1}{\ln n} = \frac{2}{\ln n + \ln 2} - \frac{1}{\ln n} \sim \frac{1}{\ln n}$$

One might argue that this is pretty small, especially if n is large. But what if we repeat our experiment? If the probability of getting a prime is $1/\ln n$, then we expect to repeat $\ln n$ times to get a prime. But $\ln n$ is a reasonable number of repetitions, which is only linear in the size of the representation of n. Therefore, we can find a prime quickly. The ability to pick a large prime number is an essential part of modern cryptography.

9 Congruence as an equivalence relation

The mathematician Gauss introduced the following notation, knows as congruence:

$$a \equiv b \pmod{n} \Leftrightarrow n \mid a - b$$

We say that a is congruent to b modulo m, equivalently a and b have the same remainder in the division by n, or $a - b$ is divisible by n. Sometimes $\mod n$ is used as an operator like in:

$$x \mod n$$

which means the remainder of x in the division by n.

The congruence notation is useful because it defines an equivalence relation on integers. An equivalence relation is reflexive, symmetric, and transitive.

- reflexive: $a \equiv a \pmod{n}$
- symmetric: $a \equiv b \pmod{n} \Leftrightarrow b \equiv a \pmod{n}$
- transitive: $a \equiv b \pmod{n}$, $b \equiv c \pmod{n} \Rightarrow a \equiv c \pmod{n}$

Congruence behaves quite similar to equality when it comes to addition, subtraction, and multiplication (and sometimes division). Assume that $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$. Then we can show the following: $a+c \equiv b+d \pmod{n}$. To see this, $n \mid a-b$ and $n \mid c-d$; therefore, $n \mid (a-b)+(c-d) = (a+c)-(b+d)$, equivalently $a + c \equiv b + d \pmod{n}$. Similarly, we can show that $a - c \equiv b - d \pmod{n}$ and $ac \equiv bd \pmod{n}$ (division will have to wait a little bit). This means that we can move variables from one side of the $\equiv$ symbol to the other just like we do with equality. For example:

$$a \equiv b \pmod{n}$$

$$b \equiv b \pmod{n}$$

$$a + b \equiv 2b \pmod{n}$$

Similarly,

$$a \equiv b \pmod{n}$$

$$b \equiv b \pmod{n}$$

$$a - b \equiv 0 \pmod{n}$$

The same applies for multiplication and division, but this has to wait until we define division properly (recall that we are only dealing with integers here).

Every equivalence relation defines equivalence classes. An equivalence class is a set of elements that are equivalent. For instance, let $n = 7$. Then since the remainder in division by 7 can take the values 0, 1, 2, 3, 4, 5, and 6, every integer must be congruent to one of these. This virtually classifies the integers into 7 classes:

$$\{\ldots, -14, -7, 0, 7, 14, \ldots\}$$
$$\{\ldots, -13, -6, 1, 8, 15, \ldots\}$$
$$\{\ldots, -12 - 5, 2, 9, 16, \ldots\}$$
$$\{\ldots, -11, -4, 3, 10, 17, \ldots\}$$
$$\{\ldots, -10, -3, 4, 11, 18, \ldots\}$$
$$\{\ldots, -9, -2, 5, 12, 19, \ldots\}$$
$$\{\ldots, -8, -1, 6, 13, 20, \ldots\}$$

Now if we replace every integer by the representative of its class, say the smallest non-negative integer in the class, then we can define a new world of arithmetic using only the set $\{0, 1, 2, 3, 4, 5, 6\}$. In fact, this is what we do with the days of the week (say 0 is Sunday)! For instance, $4 + 5 = 2$ (think of it as Thursday plus five days is Tuesday). What we really mean is $4+5 \equiv 2 \pmod{7}$. This will definitely work for addition, subtraction, and multiplication (simply replace every integer with its class representative), and we call it modular arithmetic.

But what about division in modular arithmetic? It is quite reasonable to say, for instance, that $6/3 = 2$. But what is $2/3$? It is not an integer! or is it? Let's say that $2/3 = x$, where x is an integer. This means $3x = 2$ (yes, we are defining division as the inverse of multiplication). Is there such an x? There is in our new system! Try $x = 3$. What about $3/2$? If $3/2 = y$, then $2y = 3$ and, therefore, $y = 5$ (by trial and error). But this means that 3×5 should be 1 (it is, verity!). Does division always behave nicely? The heart of the matter is to find inverses. Say I want $1/a = x \mod n$, i.e. to find an x such that $ax \equiv 1 \pmod{n}$. Does x exist, and if it does, is it unique? The answer is YES and YES if n is prime. More generally, if $a \neq 0$, we call a^{-1} the multiplicative inverse of a, i.e.

$$aa^{-1} \equiv 1 \pmod{n}$$

then a^{-1} uniquely exists if and only if n and a are co-primes. If $aa^{-1} \equiv 1 \pmod{n}$, then $aa^{-1} = kn + 1$ so $a(a^{-1}) - n(k) = 1$, thus a and n are co-primes. I will present two proofs for the other direction, one based on the Euclidean algorithm, and one on the fundamental theorem of arithmetic.

Euclidean algorithm: If n and a are co-primes, then there exists $r, s \geq 0$ such that $ar - ns = 1$. Therefore, $ar = ns + 1$, i.e. $ar \equiv 1 \pmod{n}$. As a result $a^{-1} = r \mod n$. Now assume $ax \equiv 1 \pmod{n}$ for $x < n$, then $a^{-1}ax \equiv a^{-1} \pmod{n}$. Therefore, $x \equiv a^{-1} \pmod{n}$. Since $x < n$ and $a^{-1} < n$, they must be equal.

Properties of primes: Assume $ax \equiv ay \pmod{n}$ with $x > y$, then $a(x-y) \equiv 0 \pmod{n}$. This means $n \mid a(x - y)$. But since a and n are co-primes, all the prime factors of n must be prime factors of $(x - y)$ (fundamental theorem of arithmetic). Therefore, $n|(x - y)$ which is a contradiction since $x - y < n$. This proves that ax is congruent to different values for different x's. Since x can take

n different values, we must have n different values for ax, one of them (and the only) will be 1.

While the first proof is a constructive proof, i.e. it gives a way to find a^{-1} using the Euclidean algorithm, the second proof is existential, i.e. it only proves the existence and uniqueness of a^{-1} without providing means of obtaining it.

Example: Find 1/13 mod 21 (note: 13 and 21 are co-primes). Here's a run of the Euclidean algorithm:

a	21	13	8	5	3	2	1	0
x	1	0	1	-1	2	-3	5	-13
y	0	1	-1	2	-3	5	-8	21

Therefore, 21(5)-13(8)=1. So $1/13 \equiv -8 \pmod{21}$, which is 13. Another way is to rewrite the above combination as 13(8+5)-21(12-5)=13(13)-21(8), which yields the answer 13.

Another example: Find 1/2 mod 234527 (note: 234527 is prime). Here's a run of the Euclidean algorithm:

a	234527	2	1	0
x	1	0	1	-2
y	0	1	-117263	234527

Therefore, 234527(1)-2(117263)=2(234527-117263)-234527(2-1)=1. So $1/2 \equiv 117264 \pmod{234527}$.

10 The Chinese remainder theorem

Assume we have equations of the form

$$x \equiv a_1 \pmod{n_1}$$

$$\vdots$$

$$x \equiv a_k \pmod{n_k}$$

where the n_i's are pairwise co-prime. Then x has a solution, and all solutions are congruent modulo $n = n_1 n_2 \dots n_k$.

Let $e_i = sn/n_i$, where $rn_i + s(n/n_i) = 1$ (extended Euclidean algorithm). Thus

$$e_i \equiv 1 \pmod{n_i}$$

$$e_i \equiv 0 \pmod{n_j}, i \neq j$$

It is easy to see that $x = \sum_{i=1}^{k} e_i a_i$ satisfies $x \equiv a_i \pmod{n_i}$ for every $i = 1 \ldots k$. Moreover, if x and y are solutions, then $x - y \equiv 0 \pmod{n_i}$ for every $i = 1 \ldots k$. Since n_i's are pairwise co-primes, it follows that $x - y \equiv 0 \pmod{n}$ (why?)

11 Fermat's little theorem and primality testing

Let p be prime. Fermat's little theorem is the following:

$$p \not| a \Rightarrow a^{p-1} \equiv 1 \pmod{p}$$

First note that a and p are co-primes (p is prime and $p \not| a$). As we have seen before ax for $x = 0 \ldots p-1$ must have different values modulo p. Therefore, those values must be $0, 1, 2, \ldots, p-1$ permuted (0 when $x = 0$). So,

$$a \times 2a \times 3a \times \ldots \times (p-1)a \equiv 1 \times 2 \times 3 \times \ldots \times (p-1) \pmod{p}$$

$$(p-1)!a^{p-1} \equiv (p-1)! \pmod{p}$$

$$p \mid (p-1)!a^{p-1} - (p-1)!$$

$$p \mid (p-1)!(a^{p-1} - 1)$$

Since p divides the above product, it must divide one of the factors. But p cannot divide any integer $< p$; therefore, $p \mid a^{p-1} - 1$ and this proves the theorem.

Another way to prove Fermat's little theorem is to prove that $p \mid a(a^{p-1} - 1)$ for any a; this will give the above result when $p \not| a$ because p will have to divide $a^{p-1} - 1$. Since there is now no restriction on a, we can now proceed by induction. $a = 1$ provides the base case and assume that for $a > 1$, $p \mid (a-1)^p - (a-1)$. Then

$$a^p - a = [1 + (a-1)]^p - (a-1) - 1$$

Using the binomial theorem, this is

$$\binom{p}{0} + \binom{p}{1}(a-1) + \ldots + \binom{p}{p-1}(a-1)^{p-1} + \binom{p}{p}(a-1)^p - (a-1) - 1$$

$$= \binom{p}{1}(a-1) + \ldots + \binom{p}{p-1}(a-1)^{p-1} + [(a-1)^p - (a-1)]$$

Since $p \mid \binom{p}{k}$ when $0 < k < p$ (you were asked to prove this earlier), then p divides every term in the above expression. Done.

Fermat's little theorem can be strengthened in the following way:

$$p \text{ is prime} \Leftrightarrow a^{p-1} \equiv 1 \pmod{p} \text{ for all } 1 \leq a < p$$

Since when $a < p$, $p \nmid a$, the first direction is trivial from the previous version of the theorem. For the second direction, I will use the contrapositive to prove that if p is composite, then there exists $a < p$ such that $a^{p-1} \not\equiv 1 \pmod{p}$. If p is composite, then $p = ab$ where $a < p$ and p and a must share a prime factor (why?), say q (could be a itself). Then $q \mid a \Rightarrow q \mid a^{p-1} \Rightarrow q \nmid a^{p-1} - 1$. This means $p \nmid a^{p-1} - 1$; otherwise, q would because $q \mid p$. Therefore, $a^{p-1} \not\equiv 1 \pmod{p}$.

Fermat's little theorem can be used to test whether a number is prime or not. Of course the brute force algorithm of checking if some number $a < n$ divides n is highly inefficient (one has to consider all numbers less than $\sqrt{n}$ in the worst case). But consider the following algorithm:

```
//to check if n is prime
repeat k times
  pick a random a in {1,...,n-1}
  if a^(n-1) mod n is not 1
    return composite
return prime
```

Fermat's little theorem tells us that if n is composite, there must be an $a < n$ that will fail the Fermat test. We repeat k times with the hope to find it, and if we don't we declare n as prime. Therefore, if n is prime we have nothing to worry about, but if n is composite, are we going to be lucky in finding that a? Assume an $a < n$ exists such that $\gcd(a, n) = 1$ and $a^{n-1} \not\equiv 1 \pmod{n}$. For every b such that $b^{n-1} \equiv 1 \pmod{n}$, we have

$$(ab)^{n-1} \equiv a^{n-1}b^{n-1} \equiv a^{n-1} \not\equiv 1 \pmod{n}$$

And since a and n are co-primes, a has a multiplicative inverse modulo n, then $ab \equiv ac \pmod{n} \Rightarrow b = c$ (simply multiply by a^{-1} on both sides). Therefore, every $b < n$ that passes the Fermat test is associated with a unique integer $ab \mod n$ that fails it. This means at least half of the integers will fail the test. The probability of not finding one is, therefore, at most $1/2^k$ which is tiny if $k = 100$.

But what if n is composite and every a such that $\gcd(a, n) = 1$ passes the Fermat test? Such n is called a Charmichael number. Charmichael numbers are rare, the smallest is $561 = 3 \times 11 \times 17$. The Fermat test above can be modified to handle Charmichael numbers, e.g. the Miller-Rabin algorithm, but this is beyond the scope of the course.

One final note is that for the algorithm to be efficient, a^{n-1} is always computed modulo n (so numbers are always less than n) and using a technique called repeated squaring (so no more than $O(\log n)$ multiplications are needed). Repeated squaring relies on the fact that $a^b = (a^{b/2})^2$.

$$a^b \mod n = f(a, b, n) = \begin{cases} 1 & b = 0 \\ f^2(a, b/2, n) \mod n & b \text{ even} \\ af(a, b-1, n) \mod n & b \text{ odd} \end{cases}$$

Here's an example for $a = 2$ and $n = 30$ (so we need to compute $2^{29} \mod 30$):

b	29	28	14	7	6	3	2	1	0
	odd	even	even	odd	even	odd	even	odd	
	$\times a$	$(\,)^2$	$(\,)^2$	$\times a$	$(\,)^2$	$\times a$	$(\,)^2$	$\times a$	
a^b	536870912	268435456	16384	128	64	8	4	2	1
$a^b \mod n$	2	16	4	8	4	8	4	2	1

12 Cryptography

Let n be prime and choose e such that $\gcd(e, n-1) = 1$ (e and $n-1$ are co-primes). Declare (e, n) as public information and an encoding scheme that encodes $x < n$ as follows:

$$y = x^e \mod n$$

Given y, it is generally hard to obtain x. However, since e and $n-1$ are co-primes, there exists a d such that

$$ed \equiv 1 \pmod{n-1}$$

Now

$$y^d \equiv (x^e)^d = x^{ed} = x^{k(n-1)+1} = xx^{k(n-1)} = x(x^{n-1})^k \equiv x \pmod{n}$$

The congruence follows from Fermat's little theorem because n is prime and $n \not| x$ ($x < n$), hence $x^{n-1} \equiv 1 \pmod{n}$. Since $x < n$, we can decode y back into x

$$x = y^d \mod n$$

This is the basic idea behind public/private key encryption/decryption, where (e, n) is the public key used to encode, and (d, n) is the private key required to decode. But it is extremely easy to obtain d using the Euclidean algorithm once (e, n) is known; it is simply the multiplicative inverse of e modulo $n-1$. We now modify the algorithm slightly to make it almost impossible to obtain d from (e, n).

- pick two large primes p and q, and let $n = pq$
- pick e such that e and $(p-1)(q-1)$ are co-primes
- (e, n) is public
- d is such that $ed \equiv 1 \pmod{(p-1)(q-1)}$
- (d, n) is private
- encode $x < n$: $y = x^e \mod n$
- decode $x = y^d \mod n$

Since it is generally hard to factor numbers into primes, it is almost impossible to obtain p and q. This makes it hard to compute d. Now let's show that the decoding still works. Observe that $y \equiv x^e \pmod{p}$ (because $y = x^e \mod pq$).

$$y^d \equiv (x^e)^d = x^{ed} = x^{k(p-1)(q-1)+1} = xx^{k(p-1)(q-1)} = x(x^{p-1})^{k(q-1)}$$

If $p \mid x$, then $x \equiv 0 \pmod{p}$. This means $y^d \equiv 0 \equiv x \pmod{p}$. If $p \nmid x$, then $x^{p-1} \equiv 1 \pmod{p}$ by Fermat's little theorem, hence $y^d \equiv x \pmod{p}$. Therefore,

$$y^d \equiv x \pmod{p}$$

and similarly,

$$y^d \equiv x \pmod{q}$$

This means $(y^d - x)$ is a multiple of p and q, thus a multiple of pq (uniqueness of prime factorization). Therefore, $y^d \equiv x \pmod{pq}$, i.e.

$$y^d \equiv x \pmod{n}$$

and since $x < n$,

$$x = y^d \mod n$$

13 Breaking encryption by Chinese remaindering

Let's say k people share the same value for e (e.g. 3) but they have different p's and $q's$, thus different n's. Now if a message x is encrypted using each one's public key (e, n_i) and broadcast to everyone, we have the following system of equations:

$$x^e \equiv a_1 \pmod{n_1}$$

$$\vdots$$

$$x^e \equiv a_k \pmod{n_k}$$

where $a_1, \ldots, a_k$ are the encrypted transmissions.

Since the p's and q's are different, all n's are pairwise co-prime. By the Chinese remainder theorem, we can solve for $x^e \mod n$, where $n = n_1 n_2 \ldots n_k$. But if $k \geq e$, then we know $x^e \leq n_1 n_2 \ldots n_k$ since $x < n_i$ for every $i = 1 \ldots k$. Therefore, $x^e < n$ we can determine x^e exactly and hence x. This is why a small value of e is not a good thing!