Distribution of Primes

Definition. For positive real numbers x, let $\pi(x)$ be the number of prime numbers less than or equal to x.

For example, $\pi(1) = 0$, $\pi(10) = 4$ and $\pi(100) = 25$. To use some ciphers, we will have to choose some large primes, say, 100-digit primes. The growth rate of $\pi(x)$ has a strong effect on the difficulty of finding a large prime. Fortunately for cryptography, $\pi(x)$ grows nearly as rapidly as x. This matters in SSL and HTTPS setup times.

Suppose, for example, that $\pi(x) \approx \sqrt{x}$ when x in large. Then there would be about 10^{50} 100-digit primes, that is, one out of 10^{50} 100-digit numbers would be prime. If we tried to choose a random 100-digit prime by choosing random 100-digit numbers and testing whether each is prime, we would have to try about 10^{50} 100-digit numbers to get one prime!

The Prime Number Theorem:

The ratio of $\pi(x)$ to $x/\ln x$ tends to 1 as x goes to infinity. In symbols,

$$\lim_{x\to\infty} \frac{\pi(x)}{x/\ln x} = 1.$$

Thus, $\pi(x) \approx x/\ln x$ when x in large. There are about $10^{100}/\ln 10^{100}$ 100-digit primes, that is, one out of $\ln 10^{100} \approx 230$ 100-digit numbers is prime. If we try to choose a random 100-digit prime by choosing a random 100-digit number and testing whether it is prime, we would have to try about 230 100-digit numbers to get one prime. And, since no even 100-digit number is prime, we can skip the even numbers and just try about $230/2 = 115$ odd 100-digit numbers to get one prime.

When we study congruences, we will learn a fast way to test whether a 100-digit odd number is prime, so that we can test 115 of them in a millisecond.

Identifying and Finding Primes

Now that we know there are plenty of large primes, how do we distinguish them from composite numbers? The next theorem tells how to tell in O($\sqrt{n}$) steps whether n is prime or composite. This is really slow, but one must learn to walk before one learns to run.

Theorem. If the integer $n > 1$ is composite, then n has a prime divisor $p \le \sqrt{n}$. In other words, if the integer $n > 1$ has no prime divisor $p \le \sqrt{n}$, then n is prime.

Proof: Suppose n is composite. Then we can write $n = ab$, where a and b are integers > 1. Swap a and b, if necessary, to make $1 < a \le b < n$. Then $a \le \sqrt{n}$, for if $a > \sqrt{n}$, then $b \ge a > \sqrt{n}$ and $n = ab > \sqrt{n}\sqrt{n} = n$, which is impossible. By the fundamental theorem of arithmetic a must have a prime divisor $p \le a \le \sqrt{n}$. Then p divides n.

The theorem suggests a simple algorithm for testing a small number for primality and for factoring it if it is composite.

[Factoring and Prime Testing by Trial Division]
Input: A positive integer n to factor or to test for primeness.
Output: Whether n is prime, or one or more prime factors of n.

```
$m = n$
$p = 2$
while ($p \leq \sqrt{m}$) {
     if ($m$ mod $p = 0$) {
          Print "$n$ is composite with factor $p$"
          $m = m/p$
          }
     else { $p = p + 1$ }
     }
if ($m = n$) { Print "$n$ is prime" }
else if ($m > 1$)
{ Print "The last prime factor of $n$ is $m$" }
```

Congruences

A congruence is a statement about divisibility. It is a notation that simplifies reasoning about divisibility. It suggests proofs by its analogy to equations.

Congruences are familiar to us as “clock arithmetic.” Four hours after 10 AM it will be 2 PM. How do we get the 2 from the 10 and the 4? We add four to ten and then subtract 12. We have used a congruence modulo 12.

Definition: Suppose a and b are integers and m is a positive integer. If m divides $a-b$, then we say a *is congruent to* b *modulo* m and write $a \equiv b \pmod{m}$. If m does not divide $a-b$, we say a *is not congruent to* b *modulo* m and write $a \not\equiv b \pmod{m}$. The formula $a \equiv b \pmod{m}$ is called a *congruence*. The integer m is called the *modulus* (plural *moduli*) of the congruence.

Do not confuse the binary operator "mod" in $a \bmod b$, which means the remainder when a is divided by b, with the "mod" enclosed in parentheses together with the modulus of a congruence. These concepts are related as follows. If m is a positive integer and a and b are integers, then $a \equiv b \pmod{m}$ if and only if $(a \bmod m) = (b \bmod m)$.

We will often use the fact that $a \equiv b \pmod{m}$ if and only if there is an integer k so that $a = b + km$. This fact follows immediately from the definitions of congruence and divide.

The congruence relation has many similarities to equality. The following theorem says that congruence, like equality, is an equivalence relation.

Theorem. Let m be a positive integer. Let a, b and c be integers. Then:

1. $a \equiv a \pmod{m}$.

2. If $a \equiv b \pmod{m}$, then $b \equiv a \pmod{m}$.

3. If $a \equiv b \pmod{m}$and $b \equiv c \pmod{m}$, then $a \equiv c \pmod{m}$.

Let $m > 0$ be fixed. For each integer a, the set of all integers $b \equiv a \pmod{m}$ is called the *congruence class* or *residue class* of a modulo m. The congruence class of a modulo m consists of all integers in the arithmetic progression $a + dm$, where d runs through all integers. Each integer in a congruence class is a *representative* of it. If the modulus m is understood and a and b are in the same congruence class, then each is called a *residue* of the other. The smallest nonnegative representative of a congruence class is often used as the standard representative of it. For example, the standard representative of the congruence class of 27 (mod 5) is 2.

Theorem. Let a, b, c and d be integers. Let m be a positive integer. Suppose $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$. Then

1. $a + c \equiv b + d \pmod{m}$.

2. $a - c \equiv b - d \pmod{m}$.

3. $ac \equiv bd \pmod{m}$.

Let a and b be integers. Let m be a positive integer. Let f be a polynomial with integer coefficients. If $a \equiv b \pmod{m}$, then $f(a) \equiv f(b) \pmod{m}$.

Let a and b be integers. Let m and d be positive integers with $d|m$. If $a \equiv b \pmod{m}$, then $a \equiv b \pmod{d}$.

Theorem. Let a, b and $m > 1$ be integers. If $a \equiv b \pmod{m}$ and $0 \leq a < m$ and $0 \leq b < m$, then $a = b$.

Proof. If $a = b$, we are done. If $a < b$, interchange a and b. With no loss of generality, we can assume $a > b$.

Since $a > b$ and $a < m$ and $0 \leq b$, we have $0 < a - b < m$.

The statement $a \equiv b \pmod{m}$ means that m divides $a - b$. But m does not divide any of the numbers 1, 2, ..., $m - 1$. Therefore m cannot divide $a - b$. The only other possibility is that $a = b$ (so that $a - b = 0$, which is a multiple of m).

Let $f(x)$ be the polynomial

$$1234x^3 + 5678x^2 + 9012x + 3456$$

and let $m = 7890$. Suppose we wish to evaluate $f(1000) \bmod m$ using Horner's rule:

```
ans = poly[k];
for (i=k-1; i>=0; i--)
        ans = ans*x+ poly[i];
```

to evaluate a polynomial

$$\mathtt{poly[k]}x^k + \mathtt{poly[k-1]}x^{k-1} + \cdots + \mathtt{poly[0]}$$

at x. At the end we will compute `ans%m` to form the remainder.

If we do this, the integer stored in `ans` will overflow because

$$f(x) = 1234567890123456 > 2^{31}$$

so the remainder will be wrong.

Instead we should reduce the numbers modulo m at each step. The line of code inside the `for` loop should be

```
ans = (ans*x+ poly[i])%m;
```

Although the arithmetic operations of addition, subtraction and multiplication for congruences obey the usual rules for the same operations with integers, division does not always work as for integers. For example, $2 \cdot 3 = 6 \equiv 18 = 2 \cdot 9 \pmod{12}$, but $3 \not\equiv 9 \pmod{12}$.

In general $ac \equiv bc \pmod{m}$ does not always imply $a \equiv b \pmod{m}$. We now investigate when this implication will be true.

Theorem. If gcd$(a, m) = 1$, then there is a unique x in $0 < x < m$ such that $ax \equiv 1 \pmod{m}$.

Proof: The function $f(i) = (ai \bmod m)$ for $1 \leq i \leq m-1$ is one-to-one, and so the set

$$\{ai \bmod m; i = 1, \ldots, m-1\}$$

is a permutation of $\{1, \ldots, m-1\}$. Therefore 1 appears exactly once in the first set, that is, there is exactly one x in $0 < x < m$ such that $ax \equiv 1 \pmod{m}$.

Note that the x in this theorem is like "a^{-1}," the reciprocal of a modulo m. Sometimes we even use the notation "$a^{-1} \pmod{m}$" to mean the x of this theorem.

Warning. Do not write $a^{-1} \pmod{m}$ unless you already know that $\gcd(a,m) = 1$!

When you do know that $\gcd(a,m) = 1$, a good way to compute $a^{-1} \pmod{m}$ is with the Extended Euclidean algorithm.

The Extended Euclidean algorithm gives us integers x and y with

$$ax + my = \gcd(a,m) = 1.$$

This equation implies the congruence

$$ax \equiv 1 \pmod{m}$$

so that x (or $m + x$ if $x < 0$) is $a^{-1} \pmod{m}$.

Theorem. If $m > 1$, a, b, c are integers, ($c \neq 0$), gcd(c, m) = 1, then $ac \equiv bc \pmod{m}$ implies $a \equiv b \pmod{m}$.

Proof: By the previous theorem, there is an x such that $cx \equiv 1 \pmod{m}$. Then $ac \equiv bc \pmod{m}$ implies $acx \equiv bcx \pmod{m}$, which implies $a1 \equiv b1 \pmod{m}$, which implies $a \equiv b \pmod{m}$.

Definition: A set of m integers $r_1, \ldots, r_m$ is a *complete set of residues (CSR) modulo* m if every integer is congruent modulo m to exactly one of the r_i's.

The set $\{1, \ldots, m\}$ is called the standard CSR modulo m.

Examples of Congruences

1. A computer job starts at 3 PM and runs for 100 hours. At what time of day does it finish?

This is a job for a congruence modulo 24, as there are 24 hours in a day. Now $100 \equiv 4 \pmod{24}$, and $3+4=7$, so the job finishes at 7 PM (four days later).

2. Suppose today is Tuesday. What day of the week will it be 100 days from now?

Use a congruence modulo 7, as there are 7 days in a week. We have $100 \equiv 2 \pmod{7}$ (since $7 \times 14 = 98$), so the answer is two week days after Tuesday, or Thursday.

Caesar Cipher

Let $n \bmod m$ denote the remainder when n is divided by m, i.e., mod means % in C or Java.

Use the numbers 0 to 25 to code the English alphabet: 0 = A, 1 = B, 2 = C, ..., 25 = Z.

With this code, we can encipher a message by computing with the numbers corresponding to the letters of the message.

Encipher the numbers with a formula.

Then we use the code to change numbers back to letters.

For example, we can “rotate the alphabet” by k letters.

In terms of the numbers, we encipher x by adding k to it modulo 26:
$E(x) = (x + k) \bmod 26$.

Julias Caesar used this cipher with $k = 3$. Under this cipher, the message

```
RENAISSANCE
```

is enciphered as

```
UHQDLVVDQFH
```

One deciphers this cipher either by rotating the alphabet backwards by k letters:
$D(x) = (x - k) \bmod 26$ or by rotating the alphabet forward by $26 - k$ letters: $D(x) = (x + (26 - k)) \bmod 26$. These two formulas for $D(x)$ are equivalent because

$$(x - k) \equiv (x + (26 - k)) \pmod{26}.$$

The case $k = 13$ gives the rotate cipher used in some newsgroups. It has the nice property that the deciphering formula is the same as the enciphering formula:
$D(x) = (x + 13) \bmod 26 = E(x)$. This works because

$$x - 13 \equiv x + 13 \pmod{26}.$$

Another possibility is to multiply the numbers that represent the letters by a constant: $E(x) = kx \bmod 26$.

Under this cipher with $k = 9$, the message

`RENAISSANCE`

is enciphered as

`XKNAUGGANSK`

In order for deciphering to be possible, k and 26 must be relatively prime. When this is so, let $jk \bmod 26 = 1$

The deciphering function is $D(x) = jx \bmod 26$.

How do you compute j from k and 26? Hint: Extended Euclid.

Linear Congruences

We now tell how to solve congruences like $ax \equiv b \pmod{m}$, where a, b and $m > 1$ are given integers and x is an unknown integer. The solution to an equation $ax = b$, where $a \neq 0$, is the single number $x = b/a$. In contrast, if the congruence $ax \equiv b \pmod{m}$ has any solution, then infinitely many integers x satisfy it.

For example, the solution to the congruence $2x \equiv 1 \pmod{5}$ is all integers of the form $x = 5k + 3$, where k may be any integer, that is, x lies in the arithmetic progression

$$\ldots, -12, -7, -2, 3, 8, 13, 18, \ldots.$$

This set of integers may be described compactly as $x \equiv 3 \pmod{5}$. We could have written this solution as $x \equiv 28 \pmod{5}$, but we generally use the least nonnegative residue as the standard representative of its congruence class.

Theorem. Let $m > 1$, a and b be integers. Then $ax \equiv b \pmod{m}$ has a solution if and only if gcd(a, m) divides b.

Theorem. Let $m > 1$, a and b be integers. Suppose gcd$(a, m) = 1$. Then $ax \equiv b \pmod{m}$ has exactly 1 solution modulo m. It is

$$x \equiv bx_0 \pmod{m},$$

where x_0 is any solution of $ax_0 \equiv 1 \pmod{m}$. This means that

$$x = bx_0 + tm, \quad t = 0, 1, \ldots,$$

are all integer solutions x.

Example: Solve $7x \equiv 3 \pmod{12}$.

We find $g = \gcd(7, 12) = 1$, so there is a solution. Since $7 \cdot 7 \equiv 1 \pmod{12}$, we have $x_0 = 7$, and the solution to $7x \equiv 3 \pmod{12}$ is $x \equiv 3 \cdot 7 = 21 \equiv 9 \pmod{12}$.

Example: Solve $59x \equiv 23 \pmod{103}$.

We have $\gcd(59, 103) = 1$ since both are prime.

The Extended Euclidean Algorithm gives

$$(103)(-4) + (59)(7) = 1,$$

so $(59)(7) \equiv 1 \pmod{103}$.

This means $59^{-1} \bmod 103 = 7$.

The solution to $59x \equiv 23 \pmod{103}$ is

$x \equiv (23)(59^{-1}) \equiv (23)(7) = 161 \equiv 58 \pmod{103}$.

This may also be written $x = 58 + 103t$, where t is any integer.