Applied Cryptography and Computer Security
CSE 664 Spring 2020

Lecture 11: Introduction to Number Theory

Department of Computer Science and Engineering
University at Buffalo

Lecture Outline

- What we've covered so far:
  - symmetric encryption
  - hash functions
- Where we are heading:
  - number theory
  - public-key encryption
  - digital signatures

Lecture Outline

- Introduction to number theory
  - divisibility
  - GCD and Euclidean algorithm
  - prime and composite numbers
  - Chinese remainder theorem
  - Euler ϕ function
  - Fermat's theorem

Divisibility

- Divisibility
  - given integers a and b, we say that a divides b (denoted by $a|b$) if $b = ac$ for integer c
  - a is called a divisor of b
- Transitivity theorem
  - we are given integers a, b, and c, all of which > 1
  - if $a|b$ and $b|c$, then $a|c$
- Linear combination theorem
  - let a, b, c, x, and y be integers > 1
  - if $a|b$ and $a|c$, then $a|(bx + cy)$

Divisibility

- Division algorithm (theorem)
  - let $a > 0$ and b be two integers
  - then there exist two unique integers q and r such that $0 \leq r < a$ and $b = aq + r$
- Notation
  - the integer q is called the quotient
  - the integer r is called the remainder
  - $\lfloor x \rfloor$ is the floor of x (largest integer $\leq x$)
  - $\lceil x \rceil$ is the ceiling of x (smallest integer $\geq x$)
  - then $q = \lfloor b/a \rfloor$ and $r = b \bmod a$

Greatest Common Divisor

- Greatest common divisor (GCD)
  - suppose we are given integers a and b which are not both 0
  - their greatest common divisor $gcd(a, b) = c$ is the greatest number that divides both a and b
  - example: $gcd(128, 100) = 4$
  - it is clear that $gcd(a, b) = gcd(b, a)$
- GCD and multiplication
  - we are given integers a, b, and $m > 1$
  - if $gcd(a, m) = gcd(b, m) = 1$, then $gcd(ab, m) = 1$
  - example: $gcd(25, 7) = gcd(3, 7) = 1 \Rightarrow gcd(75, 7) = 1$

Greatest Common Divisor

- GCD and division
  - Theorem 1
    - we are given integers a and b
    - if $g = gcd(a, b)$, then $gcd(\frac{a}{g}, \frac{b}{g}) = 1$
    - example: $gcd(25, 45) = 5 \Rightarrow gcd(\frac{25}{5}, \frac{45}{5}) = gcd(5, 9) = 1$
  - Theorem 2
    - if a is a positive integer and b, q, and r are integers with $b = aq + r$, then $gcd(b, a) = gcd(a, r)$
    - we can use this theorem to find GCD

Euclidean Algorithm

- Fact: given integers $a > 0$, b, q, and r such that $b = aq + r$, $gcd(a, b) = gcd(a, r)$
- Euclidean algorithm for finding $gcd(a, b)$
  - apply the division algorithm iteratively to compute the remainder
  - the last non-zero remainder is the answer
  - while $a \neq 0$ do

    $r \leftarrow b \bmod a$

    $b \leftarrow a$

    $a \leftarrow r$

    return b

Euclidean Algorithm

- Example:
  - compute GCD of 165 and 285
  - steps of Euclidean algorithm:
  - the answer is $gcd(165, 285) =$

Towards Extended Euclidean Algorithm

- Theorem:
  - if integers a and b are not both 0, then there are integers x and y so that $ax + by = gcd(a, b)$
  - we can find x and y using the extended Euclidean algorithm
- Example:
  - find x and y such that $285x + 165y = gcd(285, 165) = 15$
  - we start with the next to last equation in our example and work backwards

Extended Euclidean Algorithm

- Example (cont.)
  - algorithm steps:
  - thus, we get
- Also, if $gcd(a, b) = 1$, then $ax + by = 1$, i.e., $ax \bmod b = 1$

Extended Euclidean Algorithm

- Input: integers $a \geq b > 0$
- Output: $g = gcd(a, b)$ and x and y with $ax + by = gcd(a, b)$
- The algorithm itself:

  $x = 1; y = 0; g = a; r = 0; s = 1; t = b$
  while $(t > 0)$ $\{$
  $\quad q = \lfloor g/t \rfloor$
  $\quad u = x - qr; v = y - qs; w = g - qt$
  $\quad x = r; y = s; g = t$
  $\quad r = u; s = v; t = w$
  $\}$

- Algorithm invariants: $ax + by = g$ and $ar + bs = t$

Extended Euclidean Algorithm

- Complexity of the algorithm (theorem)
  - this result is due to Lamé, 1845
  - the number of steps (division operations) needed by the Euclidean algorithm is no more than five times of decimal digits in the smaller of the two numbers
- Corollary
  - the number of bit operations needed by the Euclidean algorithm is $O((\log_2 a)^3)$, where a is the larger of the two numbers

Prime and Composite Numbers

- Prime numbers
  - a prime number is an integer greater than 1 which is divisible by 1 and itself
  - the first prime numbers are 2, 3, 5, 7, 11, 13, 17, etc.
- Composite numbers
  - a composite number is an integer greater than 1 which is not prime
  - the composite numbers are 4, 6, 8, 9, 10, 12, 14, etc.
- Relatively prime numbers
  - integers a and b are relatively prime is $gcd(a, b) = 1$
  - relatively prime numbers don't have common divisors other than 1

Decomposition of Numbers

- Fundamental Theorem of Arithmetics:
  - every integer $n > 1$ can be written as a product of prime numbers
  - and this product is unique if the primes are written in non-decreasing order

$$n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k} = \prod_{i=1}^{k} p_i^{e_i}$$

  - here $p_1, \ldots, p_k$ are the primes that divide n and $e_i \geq 1$ is the number of factors of p_i dividing n
  - this decomposition is called the standard representation

- Example: $84 = 2 \cdot 2 \cdot 3 \cdot 7 = 2^2 \cdot 3^1 \cdot 7^1$

Using Standard Representation

- GCD and LCM
  - we are given $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ and $m = p_1^{f_1} p_2^{f_2} \cdots p_k^{f_k}$, where p_i are prime numbers and $e_i, f_i \geq 0$
  - $gcd(n, m) = p_1^{\min(e_1, f_1)} p_2^{\min(e_2, f_2)} \ldots p_k^{\min(e_k, f_k)}$
  - the least common multiple of integers a and b is the smaller positive integer divisible by both a and b
  - $lcm(n, m) = p_1^{\max(e_1, f_1)} p_2^{\max(e_2, f_2)} \ldots p_k^{\max(e_k, f_k)}$
  - also, $gcd(a, b) \cdot lcm(a, b) = ab$

Using Standard Representation

- Examples:
  - $n = 84 = 2^2 \cdot 3 \cdot 7$
  - $m = 63 = 3^2 \cdot 7$
  - $gcd(84, 63) =$
  - $lcm(84, 63) =$
  - $gcd(84, 63) \cdot lcm(84, 63) =$

Distribution of Prime Numbers

- In cryptography, we'll need to use large primes and would like to know how prime numbers are distributed
- (Theorem) The number of prime numbers is infinite
- (Theorem) Gaps between primes
  - for every positive integer n, there are n or more consecutive composite numbers
- For a positive real number x, let $\pi(x)$ be the number of prime numbers $\leq x$

Distribution of Prime Numbers

- The Prime Number Theorem
  - $\pi(x)$ tends to $x/\ln x$ as x goes to infinity. In symbols,

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

  - this tells us that there are plenty of large primes
- The question now is how we find prime numbers
- Theorem
  - if integer $n > 1$ is composite, it has a prime divisor $p \leq \sqrt{n}$
  - in other words, if $n > 1$ has no prime divisor $p \leq \sqrt{n}$, then it is prime

Finding Primes

- This suggests a simple algorithm for testing a small number for primality (and factoring if it is composite)
  - Input: a positive integer n
  - Output: whether n is prime, or one or more factors of n

$m = n; p = 2$
while ($p \leq \sqrt{m}$) {
    if ($m \bmod p = 0$) {
        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 factor of n is m"}

Summary

- Today we've learned:
  - divisibility theorems
  - how to use Euclidean algorithm to compute GCD and more
  - the number of prime numbers is large and they are well distributed
- More on number theory is still ahead