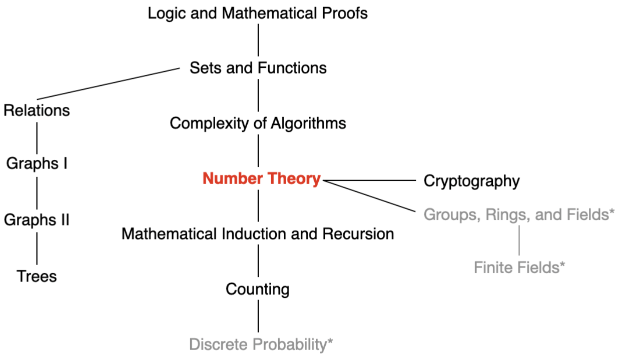
Discrete Mathematics for Computer Science

Lecture 7: Number Theory

Dr. Ming Tang

Department of Computer Science and Engineering
Southern University of Science and Technology (SUSTech)
Email: tangm3@sustech.edu.cn

Number Theory



Number Theory: divisibility and modular arithmetic, integer representations, primes, greatest common divisors, ...

Number Theory

Number theory is a branch of mathematics that explores integers and their properties, is the basis of cryptography, coding theory, computer security, e-commerce, etc.

Division

If a and b are integers with $a \neq 0$,

- we say that a divides b if there is an integer c such that $b = ac$, or equivalently b/a is an integer.
- b is divisible/divided by a

In this case, we say that a is a factor or divisor of b, and b is a multiple of a. (We use the notations $a|b$, $a \nmid b$)

Division

If a and b are integers with $a \neq 0$,

- we say that a divides b if there is an integer c such that $b = ac$, or equivalently b/a is an integer.
- b is divisible/divided by a

In this case, we say that a is a factor or divisor of b, and b is a multiple of a. (We use the notations $a|b$, $a \nmid b$)

Example:

- $4|24$
- $4 \nmid 5$

Divisibility

All integers divisible by $d > 0$ can be enumerated as:

$$\ldots, -kd, \ldots, -2d, -d, 0, d, 2d, \ldots, kd, \ldots$$

Divisibility

All integers divisible by $d > 0$ can be enumerated as:

$$\ldots, -kd, \ldots, -2d, -d, 0, d, 2d, \ldots, kd, \ldots$$

Question: Let n and d be two positive integers. How many positive integers not exceeding n are divisible by d?

Divisibility

All integers divisible by $d > 0$ can be enumerated as:

$$..., -kd, ..., -2d, -d, 0, d, 2d, ..., kd, ...$$

Question: Let n and d be two positive integers. How many positive integers not exceeding n are divisible by d?

Answer: Count the number of integers such that $0 < kd \leq n$. Therefore, there are $\lfloor n/d \rfloor$ such positive integers.

Divisibility: Properties

Let a, b, c be integers. Then the following hold:

(i) if $a|b$ and $a|c$, then $a|(b+c)$

(ii) if $a|b$ then $a|bc$ for all integers c

(iii) if $a|b$ and $b|c$, then $a|c$

Divisibility: Properties

Let a, b, c be integers. Then the following hold:

(i) if $a|b$ and $a|c$, then $a|(b+c)$

(ii) if $a|b$ then $a|bc$ for all integers c

(iii) if $a|b$ and $b|c$, then $a|c$

Proof: Suppose that $a|b$ and $a|c$. Then, from the definition of divisibility, it follows that there are integers s and t with $b = as$ and $c = at$. Hence,

$$b + c = as + at = a(s + t).$$

Therefore, a divides $b + c$.

Divisibility

Corollary: If a, b, c are integers, where $a \neq 0$, such that $a|b$ and $a|c$, then $a|(mb + nc)$ whenever m and n are integers.

Divisibility

Corollary: If a, b, c are integers, where $a \neq 0$, such that $a|b$ and $a|c$, then $a|(mb + nc)$ whenever m and n are integers.

Proof: By part (ii) and part (i) of Properties.

The Division Algorithm

If a is an integer and d a positive integer, then there are unique integers q and r, with $0 \leq r < d$, such that

$$a = dq + r.$$

In this case, d is called the divisor, a is called the dividend, q is called the quotient, and r is called the remainder.

The Division Algorithm

If a is an integer and d a positive integer, then there are unique integers q and r, with $0 \leq r < d$, such that

$$a = dq + r.$$

In this case, d is called the divisor, a is called the dividend, q is called the quotient, and r is called the remainder.

In this case, we use the notations $q = a \textbf{ div } d$ and $r = a \textbf{ mod } d$.

The Division Algorithm

If a is an integer and d a positive integer, then there are unique integers q and r, with $0 \leq r < d$, such that

$$a = dq + r.$$

In this case, d is called the divisor, a is called the dividend, q is called the quotient, and r is called the remainder.

In this case, we use the notations $q = a \textbf{ div } d$ and $r = a \textbf{ mod } d$.

Example: The quotient and remainder when 101 is divided by 11?

$$101 = 11 \times 9 + 2$$

Hence, the quotient is $9 = 101 \textbf{ div } 11$, and the remainder is $2 = 101 \textbf{ mod } 11$.

Congruence Relation

If a and b are integers and m is a positive integer, then a is congruent to b modulo m if m divides $a - b$, denoted by $a \equiv b \pmod{m}$. This is called congruence and m is its modulus.

Congruence Relation

If a and b are integers and m is a positive integer, then a is congruent to b modulo m if m divides $a - b$, denoted by $a \equiv b \pmod{m}$. This is called congruence and m is its modulus.

Example:

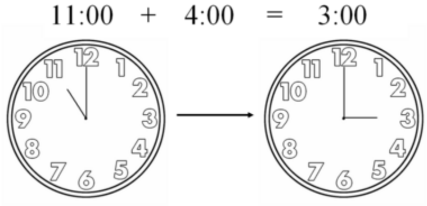
- $15 \equiv 3 \pmod{12}$
- $-1 \equiv 11 \pmod{6}$

Congruence Relation

If a and b are integers and m is a positive integer, then a is congruent to b modulo m if m divides $a - b$, denoted by $a \equiv b \pmod{m}$. This is called congruence and m is its modulus.

Example:

- $15 \equiv 3 \pmod{12}$
- $-1 \equiv 11 \pmod{6}$



Congruence Relation

Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that

$$a = b + km.$$

Congruence Relation

Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that

$$a = b + km.$$

Proof:

- If part:
- Only if part:

Congruence Relation

Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that

$$a = b + km.$$

Proof:

- If part: If there is an integer k such that $a = b + km$, then $km = a - b$. Hence, m divides $a - b$, so that $a \equiv b \pmod{m}$.
- Only if part:

Congruence Relation

Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that

$$a = b + km.$$

Proof:

- If part: If there is an integer k such that $a = b + km$, then $km = a - b$. Hence, m divides $a - b$, so that $a \equiv b \pmod{m}$.
- Only if part: If $a \equiv b \pmod{m}$, by the definition of congruence, we know that $m|(a-b)$. This means that there is an integer k such that $a - b = km$, so that $a = b + km$.

$(\mathbf{mod}\ m)$ and $\mathbf{mod}\ m$ Notations

Notations $a \equiv b \pmod{m}$ and $a \bmod m$ are different.

- $a \equiv b \pmod{m}$ is a relation on the set of integers
- In $a \bmod m$, the notation **mod** denotes a function

$(\mathbf{mod}\ m)$ and $\mathbf{mod}\ m$ Notations

Notations $a \equiv b \pmod{m}$ and $a \bmod m$ are different.

- $a \equiv b \pmod{m}$ is a relation on the set of integers
- In $a \bmod m$, the notation **mod** denotes a function

Let a and b be integers, and let m be a positive integer. Then, $a \equiv b \pmod{m}$ if and only if

$$a \bmod m = b \bmod m.$$

Congruence: Properties

Theorem: Let m be a positive integer. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then

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

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

Proof:

Congruence: Properties

Theorem: Let m be a positive integer. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then

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

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

Proof: We use a direct proof. Since $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, there are integers s and t with $a = b + sm$ and $c = d + tm$. Hence,

$$b + d = (a - sm) + (c - tm) = (a + c) + m(-s - t)$$

$$bd = (a - sm)(c - tm) = ac + m(-at - cs + stm)$$

Hence, $a + c \equiv b + d \pmod{m}$, $ac \equiv bd \pmod{m}$.

Algebraic Manipulation of Congruence

Question: If $ca \equiv cb \pmod{m}$, then $a \equiv b \pmod{m}$?

Algebraic Manipulation of Congruence

Question: If $ca \equiv cb \pmod{m}$, then $a \equiv b \pmod{m}$?

Answer: No. $14 \equiv 8 \pmod{6}$, but $7 \not\equiv 4 \pmod{6}$

Algebraic Manipulation of Congruence

Question: If $ca \equiv cb \pmod{m}$, then $a \equiv b \pmod{m}$?

Answer: No. $14 \equiv 8 \pmod{6}$, but $7 \not\equiv 4 \pmod{6}$

Question: If $a \equiv b \pmod{m}$ and c is an integer, then

- $ca \equiv cb \pmod{m}$?
- $c + a \equiv c + b \pmod{m}$?
- $a/c \equiv b/c \pmod{m}$?

Algebraic Manipulation of Congruence

Question: If $ca \equiv cb \pmod{m}$, then $a \equiv b \pmod{m}$?

Answer: No. $14 \equiv 8 \pmod{6}$, but $7 \not\equiv 4 \pmod{6}$

Question: If $a \equiv b \pmod{m}$ and c is an integer, then

- $ca \equiv cb \pmod{m}$? Yes
- $c + a \equiv c + b \pmod{m}$? Yes
- $a/c \equiv b/c \pmod{m}$? No

Computing the mod Function

Corollary: Let m be a positive integer and let a and b be integers. Then,

$$(a+b) \textbf{ mod } m = ((a \textbf{ mod } m) + (b \textbf{ mod } m)) \textbf{ mod } m$$

$$ab \textbf{ mod } m = ((a \textbf{ mod } m)(b \textbf{ mod } m)) \textbf{ mod } m$$

Computing the mod Function

Corollary: Let m be a positive integer and let a and b be integers. Then,

$$(a+b) \mathbf{\ mod\ } m = ((a \mathbf{\ mod\ } m) + (b \mathbf{\ mod\ } m)) \mathbf{\ mod\ } m$$

$$ab \mathbf{\ mod\ } m = ((a \mathbf{\ mod\ } m)(b \mathbf{\ mod\ } m)) \mathbf{\ mod\ } m$$

Proof: By the definitions of **mod** m and of congruence modulo m, we know that $a \equiv (a \mathbf{\ mod\ } m)(\mathbf{mod\ } m)$ and $b \equiv (b \mathbf{\ mod\ } m)(\mathbf{mod\ } m)$. Hence,

$$a+b \equiv (a \mathbf{\ mod\ } m) + (b \mathbf{\ mod\ } m)(\mathbf{mod\ } m)$$

$$ab \equiv (a \mathbf{\ mod\ } m)(b \mathbf{\ mod\ } m)(\mathbf{mod\ } m).$$

According to the theorem that $a \equiv b \ (\mathbf{mod\ } m)$ if and only if $a \mathbf{\ mod\ } m = b \mathbf{\ mod\ } m$, we obtain the above equalities.

Arithmetic Modulo m

Let $\mathbf{Z}_m$ be the set of nonnegative integers less than m: $\{0, 1, ..., m-1\}$.

- $+_m$: $a +_m b = (a + b) \textbf{ mod } m$
- $\cdot_m$: $a \cdot_m b = ab \textbf{ mod } m$

Arithmetic Modulo m

Let $\mathbf{Z}_m$ be the set of nonnegative integers less than m: $\{0, 1, ..., m-1\}$.

- $+_m$: $a +_m b = (a+b) \ \mathbf{mod} \ m$
- $\cdot_m$: $a \cdot_m b = ab \ \mathbf{mod} \ m$

Example:

- $7 +_{11} 9 = ?$
- $7 \cdot_{11} 9 = ?$

Arithmetic Modulo m

Let $\mathbf{Z}_m$ be the set of nonnegative integers less than m: $\{0, 1, ..., m-1\}$.

- $+_m$: $a +_m b = (a+b)$ **mod** m
- $\cdot_m$: $a \cdot_m b = ab$ **mod** m

Example:

- $7 +_{11} 9 =?$ 5
- $7 \cdot_{11} 9 =?$ 8

Arithmetic Modulo m

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers:

Arithmetic Modulo m

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers:

Closure: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.

Arithmetic Modulo m

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers:

Closure: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.

Associativity: If a, b, and c belong to $\mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.

Arithmetic Modulo m

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers:

Closure: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.

Associativity: If a, b, and c belong to $\mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.

Identity elements: $a +_m 0 = a$ and $a \cdot_m 1 = a$.

Arithmetic Modulo m

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers:

Closure: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.

Associativity: If a, b, and c belong to $\mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.

Identity elements: $a +_m 0 = a$ and $a \cdot_m 1 = a$.

Additive inverses: If $a \neq 0$ and $a \in \mathbf{Z}_m$, then $m - a$ is an additive inverse of a modulo m. That is, $a +_m (m - a) = 0$ and $0 +_m 0 = 0$.

Arithmetic Modulo m

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers:

Closure: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.

Associativity: If a, b, and c belong to $\mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.

Identity elements: $a +_m 0 = a$ and $a \cdot_m 1 = a$.

Additive inverses: If $a \neq 0$ and $a \in \mathbf{Z}_m$, then $m - a$ is an additive inverse of a modulo m. That is, $a +_m (m - a) = 0$ and $0 +_m 0 = 0$.

Commutativity: If $a, b \in \mathbf{Z}_m$, then $a +_m b = b +_m a$.

Arithmetic Modulo m

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers:

Closure: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.

Associativity: If a, b, and c belong to $\mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.

Identity elements: $a +_m 0 = a$ and $a \cdot_m 1 = a$.

Additive inverses: If $a \neq 0$ and $a \in \mathbf{Z}_m$, then $m - a$ is an additive inverse of a modulo m. That is, $a +_m (m - a) = 0$ and $0 +_m 0 = 0$.

Commutativity: If $a, b \in \mathbf{Z}_m$, then $a +_m b = b +_m a$.

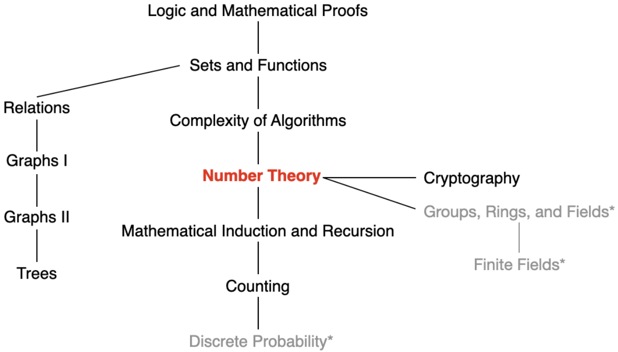
Distributivity: If $a, b, c \in \mathbf{Z}_m$, then

$$a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$$

$$(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$$

SUSTech Southern University of Science and Technology

Number Theory



Number Theory: divisibility and modular arithmetic, integer representations, primes, greatest common divisors, ...

Representations of Integers

We may use decimal (base 10), binary, octal, hexadecimal, or other notations to represent integers.

Representations of Integers

We may use decimal (base 10), binary, octal, hexadecimal, or other notations to represent integers.

Let $b > 1$ be an integer. Then if n is a positive integer, it can be expressed uniquely in the form

$$n = a_k b^k + a_{k-1} b^{k-1} + \quad + a_1 b + a_0,$$

where k is nonnegative, a_k's are nonnegative integers less than b. The representation of n is called the base-b expansion of n and is denoted by $(a_k a_{k-1} ... a_1 a_0)_b$.

Base-b Expansions

From binary, octal, hexadecimal expansions to the decimal expansion:

Base-b Expansions

From binary, octal, hexadecimal expansions to the decimal expansion:

Example

- ◇ $(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 351$
- ◇ $(7016)_8 = 7 \cdot 8^3 + 1 \cdot 8 + 6 = 3598$

Base-b Expansions

From binary, octal, hexadecimal expansions to the decimal expansion:

Example

- $(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 351$
- $(7016)_8 = 7 \cdot 8^3 + 1 \cdot 8 + 6 = 3598$

Conversions between binary and octal (or hexadecimal) expansions:

Base-b Expansions

From binary, octal, hexadecimal expansions to the decimal expansion:

Example

- $(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 351$
- $(7016)_8 = 7 \cdot 8^3 + 1 \cdot 8 + 6 = 3598$

Conversions between binary and octal (or hexadecimal) expansions:

Example

- $(101011111)_2 = (\underline{101}\,\overline{011}\,\underline{111}) = (537)_8$
- $(7016)_8 = (\underline{111}\,\overline{000}\,\underline{001}\,\overline{110})_2$
  $= (\underline{1110}\,\overline{0000}\,\underline{1110})_2 = (E0E)_{16}$

Base-b Expansions

From decimal expansion to the base-b expansion:

Base-b Expansions

From decimal expansion to the base-b expansion:

$$
\begin{aligned}
n &= a_k b^k + a_{k-1} b^{k-1} + a_{k-2} b^{k-2} + \cdots + a_2 b^2 + a_1 b + a_0 \\
&= b(a_k b^{k-1} + a_{k-1} b^{k-2} + a_{k-2} b^{k-3} + \cdots + a_2 b + a_1) + a_0 \\
&= b(b(a_k b^{k-2} + a_{k-1} b^{k-3} + a_{k-2} b^{k-4} + \cdots + a_2) + a_1) + a_0 \\
&= \cdots
\end{aligned}
$$

Base-b Expansions

From decimal expansion to the base-b expansion:

$$\begin{aligned}
n &= a_k b^k + a_{k-1} b^{k-1} + a_{k-2} b^{k-2} + \cdots + a_2 b^2 + a_1 b + a_0 \\
&= b(a_k b^{k-1} + a_{k-1} b^{k-2} + a_{k-2} b^{k-3} + \cdots + a_2 b + a_1) + a_0 \\
&= b(b(a_k b^{k-2} + a_{k-1} b^{k-3} + a_{k-2} b^{k-4} + \cdots + a_2) + a_1) + a_0 \\
&= \cdots
\end{aligned}$$

- Divide n by b to obtain $n = bq_0 + a_0$, with $0 \leq a0 < b$

Base-b Expansions

From decimal expansion to the base-b expansion:

$$\begin{aligned} n &= a_k b^k + a_{k-1} b^{k-1} + a_{k-2} b^{k-2} + \cdots + a_2 b^2 + a_1 b + a_0 \\ &= b(a_k b^{k-1} + a_{k-1} b^{k-2} + a_{k-2} b^{k-3} + \cdots + a_2 b + a_1) + a_0 \\ &= b(b(a_k b^{k-2} + a_{k-1} b^{k-3} + a_{k-2} b^{k-4} + \cdots + a_2) + a_1) + a_0 \\ &= \cdots \end{aligned}$$

- Divide n by b to obtain $n = bq_0 + a_0$, with $0 \leq a0 < b$
- The remainder a_0 is the rightmost digit in the base-b; expansion of n. Then divide q_0 by b to get $q_0 = bq_1 + a_1$ with $0 \leq a1 < b$;

Base-b Expansions

From decimal expansion to the base-b expansion:

$$\begin{aligned} n &= a_k b^k + a_{k-1} b^{k-1} + a_{k-2} b^{k-2} + \cdots + a_2 b^2 + a_1 b + a_0 \\ &= b(a_k b^{k-1} + a_{k-1} b^{k-2} + a_{k-2} b^{k-3} + \cdots + a_2 b + a_1) + a_0 \\ &= b(b(a_k b^{k-2} + a_{k-1} b^{k-3} + a_{k-2} b^{k-4} + \cdots + a_2) + a_1) + a_0 \\ &= \cdots \end{aligned}$$

- Divide n by b to obtain $n = bq_0 + a_0$, with $0 \leq a0 < b$
- The remainder a_0 is the rightmost digit in the base-b; expansion of n. Then divide q_0 by b to get $q_0 = bq_1 + a_1$ with $0 \leq a1 < b$;
- a_1 is the second digit from the right; continue by successively dividing the quotients by b until the quotient is 0

Base-b Expansions

procedure *base b expansion*(n, b: positive integers with $b > 1$)
$q := n$
$k := 0$
while ($q \neq 0$)
    $a_k := q$ **mod** b
    $q := q$ **div** b
    $k := k + 1$
return$(a_{k-1}, \dots, a_1, a_0)$ $\{(a_{k-1} \dots a_1 a_0)_b$ is base b expansion of $n\}$

SUSTech Southern University of Science and Technology

Base-b Expansions

Example: Find the hexadecimal expansion of $(177130)_{10}$.

Base-b Expansions

Example: Find the hexadecimal expansion of $(177130)_{10}$.

Solution: First divide 177130 by 16 to obtain

$$177130 = 16 \cdot 11070 + 10.$$

Base-b Expansions

Example: Find the hexadecimal expansion of $(177130)_{10}$.

Solution: First divide 177130 by 16 to obtain

$$177130 = 16 \cdot 11070 + 10.$$

Successively dividing quotients by 16 gives

$$11070 = 16 \cdot 691 + 14,$$

$$691 = 16 \cdot 43 + 3,$$

$$43 = 16 \cdot 2 + 11,$$

$$2 = 16 \cdot 0 + 2.$$

The successive remainders that we have found, 10, 14, 3, 11, 2. It follows that $(177130)_{10} = (2B3EA)_{16}$.

Binary Addition of Integers

$a = (a_{n-1}a_{n-2}...a_1a_0)_2$, $b = (b_{n-1}b_{n-2}...b_1b_0)_2$

Binary Addition of Integers

$$a = (a_{n-1}a_{n-2}\ldots a_1a_0)_2,\ b = (b_{n-1}b_{n-2}\ldots b_1b_0)_2$$

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1},a_{n-2},\ldots,a_0)_2$ and $(b_{n-1},b_{n-2},\ldots,b_0)_2$, respectively}
$c := 0$
for $j := 0$ to $n-1$
  $d := \lfloor (a_j + b_j + c)/2 \rfloor$
  $s_j := a_j + b_j + c - 2d$
  $c := d$
$s_n := c$
return($s_0, s_1, \ldots, s_n$){the binary expansion of the sum is $(s_n, s_{n-1}, \ldots, s_0)_2$}

Binary Addition of Integers

$a = (a_{n-1}a_{n-2}...a_1a_0)_2$, $b = (b_{n-1}b_{n-2}...b_1b_0)_2$

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1},a_{n-2},...,a_0)_2$ and $(b_{n-1},b_{n-2},...,b_0)_2$, respectively}
$c := 0$
for $j := 0$ to $n-1$
    $d := \lfloor (a_j + b_j + c)/2 \rfloor$
    $s_j := a_j + b_j + c - 2d$
    $c := d$
$s_n := c$
return$(s_0, s_1, ..., s_n)${the binary expansion of the sum is $(s_n, s_{n-1}, ..., s_0)_2$}

$O(n)$ bit additions

Algorithm: Binary Multiplication of Integers

$a = (a_{n-1}a_{n-2}...a_1 a_0)_2$, $b = (b_{n-1}b_{n-2}...b_1 b_0)_2$

$ab = a(b_0 2^0 + b_1 2^1 + \ + b_{n-1}2^{n-1}) = a(b_0 2^0) + a(b_1 2^1) + \ + a(b_{n-1}2^{n-1})$

Algorithm: Binary Multiplication of Integers

$a = (a_{n-1}a_{n-2}...a_1a_0)_2,\ b = (b_{n-1}b_{n-2}...b_1b_0)_2$

$ab = a(b_0 2^0 + b_1 2^1 + \ + b_{n-1}2^{n-1}) = a(b_0 2^0) + a(b_1 2^1) + \ + a(b_{n-1}2^{n-1})$

procedure *multiply*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}, a_{n-2}, ..., a_0)_2$ and $(b_{n-1}, b_{n-2}, ..., b_0)_2$, respectively}
for $j := 0$ to $n-1$
    if $b_j = 1$ **then** $c_j = a$ shifted j places
    else $c_j := 0$
{$c_0, c_1, ..., c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ to $n-1$
  $p := p + c_j$
return p {p is the value of ab}

Algorithm: Binary Multiplication of Integers

$a = (a_{n-1}a_{n-2}...a_1a_0)_2, \; b = (b_{n-1}b_{n-2}...b_1b_0)_2$

$ab = a(b_0 2^0 + b_1 2^1 + \; + b_{n-1}2^{n-1}) = a(b_0 2^0) + a(b_1 2^1) + \; + a(b_{n-1}2^{n-1})$

procedure *multiply*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}, a_{n-2}, ..., a_0)_2$ and $(b_{n-1}, b_{n-2}, ..., b_0)_2$, respectively}
for $j := 0$ to $n - 1$
  if $b_j = 1$ **then** $c_j = a$ shifted j places
  else $c_j := 0$
{$c_0, c_1, ..., c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ to $n - 1$
  $p := p + c_j$
return p {p is the value of ab}

$O(n^2)$ shifts and $O(n^2)$ bit additions

Algorithm: Computing div and mod

Compute $q = a$ **div** d and $r = a$ **mod** d:

procedure *division algorithm* (a: integer, d: positive integer)
$q := 0$
$r := |a|$
while $r \geq d$
    $r := r - d$
    $q := q + 1$
if $a < 0$ and $r > 0$ **then**
    $r := d - r$
    $q := -(q+1)$
return (q, r) {$q = a$ **div** d *is the quotient*, $r = a$ **mod** d *is the remainder* }

Southern University of Science and Technology

Algorithm: Computing div and mod

Compute $q = a$ **div** d and $r = a$ **mod** d:

procedure *division algorithm* (a: integer, d: positive integer)
$q := 0$
$r := |a|$
while $r \geq d$
    $r := r - d$
    $q := q + 1$
if $a < 0$ and $r > 0$ **then**
    $r := d - r$
    $q := -(q+1)$
return (q, r) {$q = a$ **div** d *is the quotient*, $r = a$ **mod** d *is the remainder* }

$O(q \log a)$ bit operations. But there exist more efficient algorithms with complextiy $O(n^2)$, where $n = \max(\log a, \log d)$

Algorithm: Binary Modular Exponentiation

Compute $b^n \ \mathbf{mod}\ m$:

Algorithm: Binary Modular Exponentiation

Compute $b^n \textbf{ mod } m$: Let $n = (a_{k-1} \ldots a_1 a_0)_2$.

$$b^n = b^{a_{k-1}\cdot 2^{k-1}+\cdots+a_1\cdot 2+a_0} = b^{a_{k-1}\cdot 2^{k-1}} \cdots b^{a_1\cdot 2} \cdot b^{a_0}$$

Recall that

$$ab \textbf{ mod } m = ((a \textbf{ mod } m)(b \textbf{ mod } m)) \textbf{mod } m.$$

Algorithm: Binary Modular Exponentiation

Compute b^n **mod** m: Let $n = (a_{k-1} \ldots a_1 a_0)_2$.

$$b^n = b^{a_{k-1} \cdot 2^{k-1} + \cdots + a_1 \cdot 2 + a_0} = b^{a_{k-1} \cdot 2^{k-1}} \cdots b^{a_1 \cdot 2} \cdot b^{a_0}$$

Recall that

$$ab \textbf{ mod } m = ((a \textbf{ mod } m)(b \textbf{ mod } m)) \textbf{mod } m.$$

Successively finds b **mod** m, b^2 **mod** m, b^4 **mod** m, . . . , $b^{2^{k-1}}$ **mod** m, and multiplies together the terms b^{2^j}, where $a_j = 1$.

Algorithm: Binary Modular Exponentiation

Compute b^n **mod** m: Let $n = (a_{k-1} \ldots a_1 a_0)_2$.

$$b^n = b^{a_{k-1} \cdot 2^{k-1} + \cdots + a_1 \cdot 2 + a_0} = b^{a_{k-1} \cdot 2^{k-1}} \cdots b^{a_1 \cdot 2} \cdot b^{a_0}$$

Recall that

$$ab \textbf{ mod } m = ((a \textbf{ mod } m)(b \textbf{ mod } m)) \textbf{ mod } m.$$

Successively finds b **mod** m, b^2 **mod** m, b^4 **mod** m, . . . , $b^{2^{k-1}}$ **mod** m, and multiplies together the terms b^{2^j}, where $a_j = 1$.

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2} \ldots a_1 a_0)_2$, m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ to $k - 1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return x {x equals b^n **mod** m }

Algorithm: Binary Modular Exponentiation

Use the algorithm to find 3^{644} mod 645:

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2}\ldots a_1a_0)_2$, m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ to $k - 1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return x {x equals b^n **mod** m }

Algorithm: Binary Modular Exponentiation

Use the algorithm to find 3^{644} mod 645:

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2}\ldots a_1a_0)_2$, m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ to $k-1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return x {x equals b^n **mod** m }

The algorithm initially sets $x = 1$ and $power = 3 \textbf{ mod } 645 = 3$. The binary expansion of 644 is $(1010000100)_2$. Here are the steps used:

Algorithm: Binary Modular Exponentiation

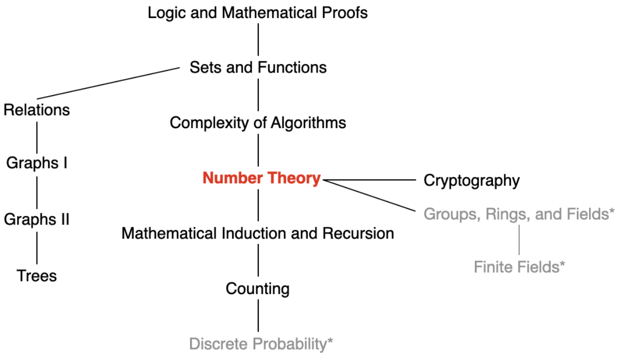
Use the algorithm to find 3^{644} mod 645:

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2}...a_1a_0)_2$, m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ to $k - 1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return x {x equals b^n **mod** m }

The algorithm initially sets $x = 1$ and $power = 3$ **mod** $645 = 3$. The binary expansion of 644 is $(1010000100)_2$. Here are the steps used:

$i = 0$: Because $a_0 = 0$, we have $x = 1$ and $power = 3^2 \textbf{ mod } 645 = 9 \textbf{ mod } 645 = 9$;
$i = 1$: Because $a_1 = 0$, we have $x = 1$ and $power = 9^2 \textbf{ mod } 645 = 81 \textbf{ mod } 645 = 81$;
$i = 2$: Because $a_2 = 1$, we have $x = 1 \cdot 81 \textbf{ mod } 645 = 81$ and $power = 81^2 \textbf{ mod } 645 = 6561 \textbf{ mod } 645 = 111$;
$i = 3$: Because $a_3 = 0$, we have $x = 81$ and $power = 111^2 \textbf{ mod } 645 = 12{,}321 \textbf{ mod } 645 = 66$;
$i = 4$: Because $a_4 = 0$, we have $x = 81$ and $power = 66^2 \textbf{ mod } 645 = 4356 \textbf{ mod } 645 = 486$;
$i = 5$: Because $a_5 = 0$, we have $x = 81$ and $power = 486^2 \textbf{ mod } 645 = 236{,}196 \textbf{ mod } 645 = 126$;
$i = 6$: Because $a_6 = 0$, we have $x = 81$ and $power = 126^2 \textbf{ mod } 645 = 15{,}876 \textbf{ mod } 645 = 396$;
$i = 7$: Because $a_7 = 1$, we find that $x = (81 \cdot 396) \textbf{ mod } 645 = 471$ and $power = 396^2 \textbf{ mod } 645 = 156{,}816 \textbf{ mod } 645 = 81$;
$i = 8$: Because $a_8 = 0$, we have $x = 471$ and $power = 81^2 \textbf{ mod } 645 = 6561 \textbf{ mod } 645 = 111$;
$i = 9$: Because $a_9 = 1$, we find that $x = (471 \cdot 111) \textbf{ mod } 645 = 36$.

Next Lecture



Number Theory: divisibility and modular arithmetic, integer representations, primes, greatest common divisors, ...