Number Theory: I

李昂生

Discrete Mathematics
U CAS
10th April, 2018

提纲

1. Background
2. Division （除）
3. Modular （模）Arithmetic （算术）
4. Representation and Algorithms
5. Prime （素数）
6. Exercises

Overviews

- The oldest math
- Foundations of CS
- Engine of algorithms
- New math
- The longest challenges in math

Definition

Definition 1

For integers a, b, $a \neq 0$, we say that a divides b, written $a|b$, if there is an integer c such that

$$b = a \cdot c$$

holds.
In this case, we say that

- a is a factor of b,
- b is a multiple of a, and
- b is divisible by a.

Intuition of division

For a positive integer d, the numbers that are divisible by d, are:

$$\cdots, -4d, -3d, -2d, -d, 0, d, 2d, 3d, 4d, \cdots. \tag{1}$$

Understanding of division

For an integer a,

- $\lfloor \frac{a}{d} \rfloor$: The greatest integer x such that $x \cdot d \leq a$.
- $\lceil \frac{a}{d} \rceil$: The least integer y such that $y \cdot d \geq a$.
- $\lfloor \frac{a}{d} \rfloor \cdot d$ is the integer part of $\frac{a}{d}$
- $a - \lfloor \frac{a}{d} \rfloor \cdot d$ is the fractional part of $\frac{a}{d}$.
- If d is a factor of a and b, then d is a factor of any linear combination of a and b, with integer coefficients.

The Division Theorem

Theorem 2

Let a be an integer and d a positive integer (or a natural number). Then there exist unique integers q and r satisfying:

(1)
$$a = q \cdot d + r,$$

(2)
$$0 \leq r < d.$$

The division theorem - intuition

Consider one-dimensional axis of real numbers with unit d, for a natural number d.
For natural number d and integer a, if both $d|a$ and $-d < a < d$ hold, then

$$a = 0.$$

Uniqueness

Suppose that q, r and q', r' satisfy:

$$a = q \cdot d + r \tag{2}$$
$$0 \leq r < d \tag{3}$$
$$a = q' \cdot d + r' \tag{4}$$
$$0 \leq r' < d. \tag{5}$$

By (3) and (5),

$$(q - q') \cdot d = (r' - r), \text{so that } d|(r' - r). \tag{6}$$

By (2) and (4),

$$-d < r' - r < d. \tag{7}$$

(6) and (7) together give $r' - r = 0$, so that $r' = r$ and $q' = q$.

Existence of q, r

In a one-dimensional real number axis with unit d, there are two cases:

Case 1 a is at some integral point $q \cdot d$.
Then $q = q$, and $r = 0$.

Case 2. $q \cdot d < a < (q+1)d$.
Then $q = q$ and $r = a - q \cdot d$.

Notations

Assume $d > 0$, a, d, q and r satisfy:

(i) $a = q \cdot d + r$, and

(ii) $0 \leq r < d$.

We call:

- d: *divisor* （除数）
- a: *divident* （被除数）
- q: *quotient* （商）, written $q = a \operatorname{div} d$
- r: *remainder* （余数）, written $r = a \mod d$.

For fixed $d > 0$,
$x \operatorname{div} d$
$x \mod d$
are both functions.

Congruence （同余）

Definition 3

Given integers *a*, *b* and natural number *m*, we say that *a is congruent to b modulo m*, if:

$$m|(a-b). \tag{8}$$

In this case, we write

$$a \equiv b \pmod{m}. \tag{9}$$

in which,
m is called *modulus* (moduli, for pl)

Remark

- $a \equiv b \pmod{m}$: *a* and *b* are congruent (or equivalent) modulo *m*
- $a \mod m$: if *m* is fixed and *a* varies, this is a function.

Basic properties - I

Theorem 4

Let a, b be integers and m be natural number. Then

$$a \equiv b \pmod{m} \iff a \bmod m = b \bmod \mathrm{m}.$$

Intuition: $a \equiv b \pmod{m}$ if and only if a and b have the same remainder divided by m.

Thinking of a one-dimensional number axis with unit m.

Basic properties - II

Theorem 5

Let a, b be integers and m be natural number. Then:
a, b are congruent modulo m if and only if there is a k such that

$$a = b + mk.$$

$\mathbb{Z}_m$

Define

$$\mathbb{Z}_m = \{0, 1, 2, \cdots, m-1\}. \tag{10}$$

Here $i \in \mathbb{Z}_m$ represents a *congruence class modulo m* （模m的同余类）, which is the set of all the numbers of the form:

$$i + km$$

for integers k.

Example: For $m = 7$, $\mathbb{Z}_7 = \{0, 1, 2, 3, 4, 5, 6\}$ represents the days in one week.
Pause to think: Why do we need this notion?

Arithmetic in $\mathbb{Z}_m$

For natural number m, in

$$\mathbb{Z}_m = \{0, 1 \cdots, m - 1\}$$

We define *addition* $+$

$$a + b = (a + b) \bmod m,$$

and *multiplication* $\cdot$

$$a \cdot b = (a \times b) \bmod m.$$

Arithmetic Theorem in $\mathbb{Z}_m$

Theorem 6

Both addition $+$ and multiplication $\cdot$ are well-defined.

Proof.

Using Theorem 4. If $a = a'$ and $b = b'$ in $\mathbb{Z}_m$, then $a + b = a' + b'$ and $a \cdot b = a' \cdot b'$ in $\mathbb{Z}_m$. □

Properties of $\mathbb{Z}_m$

For $\langle \mathbb{Z}_m, +, \cdot \rangle$, we have

- Closure（封闭性）: if $a, b \in \mathbb{Z}_m$, then so are $a + b$ and $a \cdot b$
- Associativity （结合律）: $(a + b) + c = a + (b + c)$, $(ab)c = a(bc)$.
- Commutativity （交换律）: $a + b = b + a$, and $ab = ba$.
- Identity elements （单位元）: $0 + a = a$, and $1 \cdot a = a$.
- Additive inverse （加法逆）: $a + (-a) = 0$, $-a = m - a$.
- Distributivity （分配律）: $a(b + c) = ab + ac$.

Question Is there multiplicative inverse?

Group, 群

Definition 7

A group （群）is a set G in which an operation, denoted $*$, is defined, such that the following properties are satisfied:

(1) Closure: If $a, b \in G$, then $a * b \in G$.

(2) Identity element: There is an 1 such that for every $a \in G$, $1 * a = a * 1 = a$.

(3) Inverse: For every $a \in G$, there is a $b \in G$ such that $a * b = b * a = 1$. (The inverse of 2 is $\frac{1}{2}$.)

(4) Associativity: $(a * b) * c = a * (b * c)$.

Furthermore, if $a * b = b * a$ holds for all a, b, then G is called commutative.

Examples of groups

- $\langle \mathbb{Z}, + \rangle$ is a commutative group.
- For every natural number m, $\langle \mathbb{Z}_m, + \rangle$ is a finite, commutative group.

Ring, 环

Definition 8

A ring （环）is a set R with **two operations** $+$ and $\cdot$, satisfying the following properties:

(1) $\langle R, +\rangle$ is a commutative group.

(2) $\langle R, \cdot\rangle$ is associative.

(3) $\langle R, +, \cdot\rangle$ is distributive.

Furthermore, if $\langle R, \cdot\rangle$ is commutative, we say that $\langle R, +, \cdot\rangle$ is a commutative ring.

Examples of rings

- $\langle \mathbb{Z}, +, \cdot \rangle$ is a commutative ring.
- The set of all $n \times n$ matrices with matrix （矩阵）addition and multiplication is a ring, but not commutative.
- For every natural number m, $\langle \mathbb{Z}_m, +, \cdot \rangle$ is a finite, commutative ring.

Representations

- Decimal: base 10
- Binary: base 2
- Octal: base 8
- Hexadecimal: base 16

Theorem 9

Let b be a natural number greater than 1. Then, for every positive integer n, there exists a unique base b representation of n, that is, there is a unique $k+1$-tuple $(a_0, a_1, \cdots, a_k)$ satisfying:

1. *for each j, $0 \leq a_j < b$, and $a_k \neq 0$,*
2. $$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0.$$

Uniqueness

Suppose that $(\alpha_0, \alpha_1, \cdots, \alpha_l)$ and $(\beta_0, \beta_1, \cdots, \beta_r)$ are two representations of n. Therefore,
(1) for each $0 \leq i < l$, $0 \leq \alpha_i < b$, and $0 < \alpha_l < b$,
(2) For each j with $0 \leq j < r$, $0 \leq \beta_j < b$, and $0 < \beta_k < b$,
(3) $n = \alpha_l b^l + \cdots + \alpha_1 b + \alpha_0$, and
(4) $n = \beta_r b^r + \beta_{r-1} b^{r-1} + \cdots + \beta_1 b + \beta_0$.

Uniqueness - Continued

By (3) $-$ (4), $b|(\alpha_0 - \beta_0)$, implying $\alpha_0 = \beta_0$.
Using this, we have:

$$\alpha_l b^l + \cdots + \alpha_1 b = \beta_r b^r + \cdots + \beta_1 b$$

so that

$$\alpha_l b^{l-1} + \cdots + \alpha_2 b + \alpha_1 = \beta_r b^{r-1} + \cdots + \beta_2 b + \beta_1.$$

This shows that $b|(\alpha_1 - \beta_1)$ so that $\alpha_1 = \beta_1$.
Repeating the process, we have that $l = r$, and for each i with $0 \leq i \leq l$, $\alpha_i = \beta_i$.

Existence

Suppose that

$$\begin{array}{rcl} n & = & q_0 b + a_0,\ 0 \leq a_0 < b \\ q_0 & = & q_1 b + a_1,\ 0 \leq a_1 < b \\ \cdots & & \\ q_{k-1} & = & q_k b + a_k,\ 0 < a_k < b \\ q_k & = & 0. \end{array}$$

Then

$$n = a_k b^k + a_{k-1} b^{k-1} + \cdots a_1 b + a_0. \tag{11}$$

Representation Algorithm $\mathcal{R}$

The algorithm $\mathcal{R}$ for finding representation of n with base $b > 1$.

(1) Set $q = n$, and $k = 0$.
Suppose that $a_0, a_1, \cdots, a_{k-1}$ are all defined, and a_k is undefined.

(2) If $q = 0$, then output the representation $(a_{k-1}, \cdots, a_1, a_0)$.

(3) Otherwise. Then:

(3a) define $a_k = q \bmod b$,
(3b) set $q \leftarrow q \operatorname{div} b$
(3c) set $k \leftarrow k + 1$, and
(3d) go back to step (2).

Proof of the Representation Algorithm $\mathcal{R}$ - Exercise 1

Correctness: Prove that the algorithm finds the base b representation of n.

Time complexity: Show that the number k of divisions executed in the algorithm $\mathcal{R}$ is bounded by $O(\log_b n)$.

Binary Representation of Numbers

In computer science, we usually use the binary representation of numbers, that is, the representation of base 2.
In this case, a natural number n is represented by a binary string of the form

$$n = a_k \cdots a_1 a_0,$$

where $a_k = 1$ and for each $j < k$, a_j is 0 or 1.
We always assume the convention （约定）above.

Exercise 2

Design an algorithm to find the binary representation of a natural number. Analyse the time and space complexity of your algorithm.

Time complexity: The number of times of operations (bit addition and bit division, say) used in the execution of the algorithm

Space complexity: The number of cells (each cell contains only one symbol, 0, or 1) used in the execution of the algorithm. The space complexity does not include the storage of the input natural number n, or the output of the algorithm.

空间复杂性不包括输入和输出的存储空间，只包括计算中间过程所用空间。注意，空间可以重用。

Addition algorithm $\mathcal{A}$

```
(1) set $c_{-1} = 0$.
(2) For $j$ with $0 \leq j \leq n-1$,
    (2a) set $c_j = \lfloor (a_j + b_j + c_{j-1})/2 \rfloor$,
     2b) Set $s_j \leftarrow a_j + b_j + c_{j-1} - 2c_j$.
(3) Let $s_n = c_{n-1}$.
(4) Output
```

$$s_n s_{n-1} \cdots s_1 s_0.$$

Time complexity: $O(n)$, where n is the maximal length of a and b.
Space complexity: $O(\log_2 n)$.

Multiplication

Given

$$\begin{aligned} a &= a_l \cdots a_1 a_0 \\ b &= b_r \cdots b_1 b_0. \end{aligned} \tag{14}$$

For every $j = 0, 1 \cdots, l$, set S_j

$$S_j = \begin{cases} b0 \cdots 0 (j \text{ zeros appended}), & \text{if } a_j = 1 \\ 0, & \text{Otherwise} \end{cases} \tag{15}$$

Multiplication algorithm $\mathcal{M}$

(1) Set $S = 0$ and $j = 0$.

(2) If $j = l + 1$, then output S.
Suppose that $j \leq l$.

(3) Otherwise. Then:
– set $S \leftarrow S + S_j$,
– set $j \leftarrow j + 1$, and
– go back to step (2).

The time complexity: $O(l(l + r)) = O(l \cdot r)$, if $l \leq r$.
Space complexity: $O(l + r)$.

Subtraction

Suppose that

$$a = a_l a_{l-1} \cdots a_1 a_0$$

$$b = b_k \cdots b_1 b_0$$

with $l \geq k$ and $a \geq b$.
If $a_0 \geq b_0$, then
- $d_0 = 0$ and
- $s_0 = 2d_o + a_0 - b_0$.
If $a_0 < b_0$, then
- $d_0 = 1$,
- $s_0 = 2d_0 + a_0 - b_0$

Subtraction -continued

Suppose that s_j and d_j are both defined.
Case 1 If $a_{j+1} - d_j \geq b_{j+1}$, then
- $d_{j+1} = 0$, and
- $s_{j+1} = a_{j+1} - d_j - b_{j+1}$.
Case 2. Otherwise, then
- define $d_{j+1} = 1$, and
- set $s_{j+1} = a_{j+1} + 2d_{j+1} - d_j - b_{j+1}$.
The output is

$$a - b = s_l \cdots s_1 s_0.$$

The time complexity is $O(l)$, and the space complexity is $O(\log l)$.

Algorithm for div and mod - idea

Suppose that

$$\begin{aligned} a &= a_l a_{l-1} \cdots a_1 a_0 \\ b &= b_k \cdots b_1 b_0 \end{aligned} \qquad (16)$$

are binary representations with $l \geq k$ and $a \geq b$.
Find q and r such that

$$a = qb + r$$

$$0 \leq r < b$$

Basic idea:
Get an algorithm with time complexity $O(n^3)$.

Algorithm for div and mod

Precisely, the algorithm proceeds as follows:

(1) Determine the highest digit of the quotient by reading the highest k or $k+1$ digits of a that is no less than b.
Let c be the shortest initial binary string that is greater than or equal to b.
Suppose that $a = c\hat{~}d_1 d_2 \cdots d_m$.
Define $q_m = 1$. Let $\alpha = c - b$.

Algorithm for div and mod - continued

Suppose that $q_m, \cdots, q_{m-k}$ are all defined.

(2) Let i be the least j such that $\alpha d_{k+1} \cdots d_j \geq b$.
– For each $j < i$, define $q_{m-k-j} = 0$ and $q_{m-k-i} = 1$,
– Set $\alpha \leftarrow \alpha d_{k+1} \cdots d_i - b$, and
– set $k \leftarrow k + i$.

(3) Let $q = q_m \cdots q_1 q_0$, and $r = \alpha$. Then:

$$a = q \cdot b + r$$
$$0 \leq r < b$$

The time and space complexity

The time complexity of the algorithm $\mathcal{E}$ is:

$$O(l \cdot k),$$

where l and k are the lengths of the binary representations of a and b, respectively.
The space complexity is:

$$O(l + k).$$

Remark

Note that if $a < 0$ and $b > 0$. Then let

$$-a = qb + r,\ 0 \leq r < b.$$

If $r = 0$, then $a = -qb$.
If $0 < r < b$, then

$$a = -(q+1)b + (b-r), 0 < b-r < b.$$

Modular exponentiation

Given integer a, natural numbers m, n, compute

$$a^n \bmod m.$$

Let

$$n = \alpha_l 2^l + \alpha_{l-1} 2^{l-1} + \cdots + \alpha_1 2 + \alpha_0.$$

Then

$$a^n = a^{\alpha_l 2^l} \cdot \cdots \cdot a^{\alpha_1 2} \cdot a^{\alpha_0}.$$

The algorithm

(1) Set $c_0 \leftarrow a \bmod m$,
Suppose that c_j is defined.

(2) Set $c_{j+1} \leftarrow (c_j)^2 \bmod m$.
Suppose that $c_0, c_1 \cdots, c_l$ are all defined. Note that

$$c_j = a^{2^j} \bmod m.$$

(3) Set $s = 1$.

(4) For j from 0 to l, in increasing order, if $\alpha_j = 1$, then
– set $s \leftarrow (s \cdot c_j) \bmod m$.

The time and space complexity

- Step (1): $\log|a|\log m$ (where $|a|$ is the absolute value of a.)
- Step (2):
  $l = \log n$ rounds,
  each round: $\log^2 m$
- Step (3): The same as step (2)

The total time complexity is:

$$O(\log|a|\log m + \log n \log^2 m).$$

The space complexity is the same as that for computing $a \bmod m$, which is $O(\log a + \log m)$ due to the fact that space can be reused.

The fundamental theorem of arithmetic

Definition 10

We say that a natural number n is *prime*, if there are no $a, b < n$ such that $n = a \cdot b$, and *composite*, otherwise.

Intuition: Primes are the “atomic" or “building block" of numbers.

Theorem 11

Every integer greater than 1 *can be uniquely represented by the following form*

$$n = p_1 p_2 \cdots p_k,$$

where p_j's are primes in increasing order.

Existence

We prove by induction on n. It is clear for $n = 2$. For $n > 2$.
Suppose by induction that the theorem holds for all $n' < n$.
Case 1 n is prime.
Done.
Case 2. Otherwise.
In this case, there is a prime p such that $n = p \cdot n_1$.
By the inductive hypothesis, there is a prime factoring $q_1 q_2 \cdot q_l$ of n_1.
Reordering $p, q_1, \cdots, q_l$ in increasing order, gives rise to a prime factoring of n.

Uniqueness

Suppose by induction that the result holds for all $n' < n$.
Suppose that

$$n = p_1 p_2 \cdots p_l$$

$$n = q_1 q_2 \cdots q_r$$

satisfying:

- all p_i, q_j's are primes
- $p_1 \leq p_2 \leq \cdots \leq p_l$
- $q_1 \leq q_2 \leq \cdots \leq q_r$.

Then both p_1 and q_1 are the least prime factor of n, giving $p_1 = q_1 = p$.
Dividing n by p, the same proof shows that $p_2 = q_2$.
Continuing the procedure, we have that $l = r$, and for each j from 1 to l, $p_j = q_j$.

Intuition

Whenever we see a natural number n, we can think of a prime factoring of n of the following form:

$$n = p_1 p_2 \cdots p_k,$$

for primes $p_1 \leq p_2 \leq \cdots \leq p_k$.

A great challenge

The proof above describes an effective mechanism （能行机械）to find the prime factoring of a natural number *n*. However, it is a great challenge to find an efficient algorithm （有效算法）that

- runs in time complexity $\log^{O(1)} n$, and that
- finds the unique prime factoring of *n*.

Significance: The current cryptosystem depends on the hardness of prime factoring.
Good: There exists a quantum algorithm that finds prime factors of *n* in time polynomial of $\log n$.
Bad: Quantum computers are hard to build.

Roles in science

Gödel and Turing used the fundamental theorem of arithmetic to encode symbolic reasoning and the computation of Turing machines to natural numbers, so that reasoning and computation become a theory of natural numbers.
This actually encodes all discrete objects to natural numbers, allowing us to understand the discrete objects by the theory of numbers.
The reason is that both the encoding and decoding based on the prime factoring are computable by mechanisms.
The Theorem plays a fundamental role in both Gödel and Turing's theories, in the last century.

Questions

The fundamental questions about primes are:

1. Is there a polynomial time algorithm that decides, for a given natural number n, whether or not n is prime?
2. Is there a polynomial time algorithm that finds a prime factor of a given natural number?

The two questions are essential to Computer science.

Basic property

Theorem 12

If n is composite, then there exists a prime $p \leq \sqrt{n}$ such that $p|n$.

Towards a contradiction. If $n = q_1 q_2 \cdots q_l$, for $l \geq 2$ and for primes q_j. Then each $q_j > \sqrt{n}$, implying $q_1 q_2 > n$. A contradiction.

Significance: Anyway, the theorem reduces somehow the search space for a prime factor of a natural number, leading to some algorithms.

The sieve

The basic theorem in the last page suggests the method of sieving for finding all the primes not exceeding a fixed natural number n, 100 say.
The sieving proceeds as follows:

1. Find all primes less than or equal to $\sqrt{n}$. Suppose that $p_1, p_2, \cdots, p_l$ is the list of all such primes, in increasing order. Let $L = \{2, 3, \cdots, n\}$.
   Let $j = 1$
2. (Sieving cycle S_j) Cycle S_j:
   - (2a) For each $x \in L$, is p_j is a proper prime factor, i.e., $p_j | x$ and $p_j \neq x$, then sieve x out from L
   - (2b) If $j = l$, then output L and terminate.
   - (2c) If $j < l$, then set $j \leftarrow j + 1$, and go back to step 2 above.

By the theorem, L is the set of all the primes less than or equal to n.

A question

How good is the sieving? What is the limit of sieving?
Chen and other Chinese in 1960 - 1980 achieved significant results (Chen's so called $1 + 2$)

Primes are infinitely many - Exercise 3

Prove the two theorems below

Theorem 13

There are infinitely many primes.

Theorem 14

There are infinitely many primes of the form $3k + 2$*.*

Primes of particular form

(1) (Dirichlet) For a, b with $(a, b) = 1$, there are infinitely many primes of the form $ak + b$.

(2) Erdös conjecture (1930):
For any n, there are a, b with $(a, b) = 1$ such that $ak + b$ for all $k \in \{1, 2, \cdots, n\}$ are primes.
Green, Tao, 2006, proved the conjecture. Tao was awarded the Fields Medal due to this progress.

The Prime Number Theorem

For every natural number x, let π_x be the number of primes less than or equal to x.

Theorem 15

(1896)

$$\lim_{x\to\infty} \frac{\pi_x}{\frac{x}{\ln x}} = 1. \tag{17}$$

This means that

$$\pi_n \approx \frac{n}{\ln n}.$$

要求：记住这个定理。

The Prime Number Theorem -note

- Before 1896, experimental verification
- Gauss conjectured at the age of 15 or 16
- 1896, Hadamard, and de la Valle Poussian independently proved the result using complex analytic properties of Riemann zeta function.
- 1948, Atle Selberg gave a proof without using complex analysis
- Full proof can be found in books of classical number theory

Application of the Prime Number Theorem

(i) The number of primes within m is approximately $\frac{m}{\ln m}$. Therefore, for a number n randomly and uniformly chosen within m, the probability that n is a prime is

$$\approx \frac{1}{\ln m},$$

which is significantly large, since $\ln m << m$.

(ii) For $l = \ln m$, if $n_1, n_2, \cdots, n_l$ is within m, each of which is randomly and uniformly chosen, then with probability greater than some constant α, one of the n_i is a prime.

Primality test

1. M. O. Rabin, Probabilistic algorithm for testing primality. J. Number Theory, 12, pp, 128 - 138, 1980.
2. Manindra Agrawal, Neeraj Kayal, Nitin Saxena, Primality is in P, Annals of Mathematics, 2008
   Time complexity $O(n^6)$, impractical at the moment.

习题 I

1. Page 27
2. Page 29
3. Page 55

习题II

1. Show that if n and k are positive integers, then $\lceil \frac{n}{k} \rceil = \lfloor \frac{n-1}{k} \rfloor + 1$.
2. Design an algorithm that, on binary representations of integers a and b, determines whether $a > b$, $a = b$, or $a < b$. Analyse the time and space complexity of your algorithm.
3. How many zeros are there at the end of 100!?
4. Show that $\log_2 3$ is an irrational number.
5. Show that if $2^n - 1$ is prime, then n is prime.

谢谢！