

# Discrete

## Mathematics - P46

二进制除法算法

```

procedure division algorithm ( $a \in \mathbb{Z}$ ,  $d \in \mathbb{Z}^+$ )
     $q := 0$ 
     $r := |a|$  得到  $q \in \mathbb{N}$ ,  $r \in \mathbb{N}$  且  $0 \leq r < d$ 
    while  $r \geq d$  do
         $r := r - d$  使得  $|a| = dq + r$ 
         $q := q + 1$  在  $a > 0$  时,  $q = a \text{ div } d$ 
    if  $a < 0$  且  $r > 0$  then
         $r := d - r$  如果  $a < 0$  且  $r > 0$ , 即有余数,  $a/d$  不是整数且  $a/d < 0$ 
         $q := -q + 1$  于是有  $\lfloor a/d \rfloor = -(\lfloor a/d \rfloor + 1)$ 
    return  $(q, r)$  { $q = a \text{ div } d$ ,  $r = a \text{ mod } d$ }

```

(注意: 这个算法要求除数为正整数, 对 0 的检查可以放在算法开始处)

- 对负整数的处理可在开始处取  $d' = |d|$ , 并替换循环中的变量  $d$

然后在最后调整时修改条件  $a < 0$  为  $(a < 0 \wedge d > 0) \vee (a > 0 \wedge d < 0)$ , 即  $a, d$  异号

且 if  $d < 0$  then  $r := -r$ , 即  $d < 0$  时余数取相反数

特别注意: 这个算法遵循数学意义上对于  $a \text{ div } d = \lfloor a/d \rfloor$ ,  $a \text{ mod } d = a - d \cdot \lfloor a/d \rfloor$  的定义

另外这种算使用  $O(q \log|a|)$  次位运算, 更有效的方式为  $O(\log|a| \cdot \log|d|)$ : 2=进制位运算

模指数运算

指形如  $b^n \text{ mod } m$  的运算, 其中  $b, n, m \in \mathbb{Z}^+$  且都是大整数, 则先求  $b^n$  再求模是不现实的

但是注意到  $n$  有二进制展开式  $n = a_k 2^k + a_{k-1} 2^{k-1} + \dots + a_1 \cdot 2^1 + a_0 \cdot 2^0 = (a_k a_{k-1} \dots a_0)_2$

于是有  $b^n = b^{a_k 2^k + a_{k-1} 2^{k-1} + \dots + a_0 \cdot 2^0} = b^{a_k 2^k} \cdot b^{a_{k-1} 2^{k-1}} \cdots b^{a_0 2^0} = \prod_{i=0}^k b^{a_i 2^i} = \prod_{i=0}^k (b^{2^i})^{a_i}$

$\forall a_i \in \{0, 1\}$ , 则当  $a_i = 0$  时,  $b^{a_i 2^i} = 1$

则  $b^n \text{ mod } m = (\prod_{i=0}^k b^{a_i 2^i}) \text{ mod } m = (\prod_{i=0}^k ((b^{2^i})^{a_i} \text{ mod } m)) \text{ mod } m$   $O(c \log m^2 \log n)$  次

$\forall b^{2^{i+1}} \text{ mod } m = (b^{2^i})^2 \text{ mod } m = (b^{2^i} \text{ mod } m)^2 \text{ mod } m$  二进制位运算

```

procedure modular exponentiation ( $b \in \mathbb{Z}$ ,  $m \in \mathbb{Z}^+$ ,  $n = (a_{k-1} a_{k-2} \dots a_0)_2$ )

```

$x := 1$  初始化  $x = 1$   $i$   $b^{2^i} (\text{power})$   $a_i$   $x = 1$

$\text{power} := b \text{ mod } m$   $\text{power} = b \text{ mod } m$   $0$   $\text{power} \rightarrow 0$   $\longrightarrow x \times 1$

for  $i := 0$  to  $k-1$

$\quad$  if  $a_i = 1$  then  $x := (x \cdot \text{power}) \text{ mod } m$   $= (b \text{ mod } m)^2 \text{ mod } m \rightarrow 0$   $\longrightarrow x \times 1$

$\quad$   $\text{power} := (\text{power} \cdot \text{power}) \text{ mod } m$   $b^{2^i} \text{ mod } m \rightarrow 1$   $\longrightarrow (x \cdot b^{2^i} \text{ mod } m) \text{ mod } m$

$\quad$   $\text{return } x$

$\quad$   $(b^{2^{k-1}} \text{ mod } m) \rightarrow 1$   $\longrightarrow (x \cdot b^{2^{k-1}} \text{ mod } m) \text{ mod } m$

循环不变量

初始化: 第 1 次迭代前,  $x$  为  $\prod_{i=0}^{k-1} (b^{2^i})^{a_i}$  前 0 项的积, 即  $x = 1$ ,  $\text{power}$  为  $b^{2^0} \text{ mod } m = b \text{ mod } m$

保持: 第  $j$  次迭代前,  $x$  为  $\prod_{i=0}^{j-1} (b^{2^i})^{a_i}$  前  $j-1$  项积,  $\text{power}$  为  $b^{2^{j-1}} \text{ mod } m$  即第  $j$  项, 若  $a_{j-1} = 1$  则至  $x$ , 否则  $x$  保持

第  $j$  次迭代后,  $x$  为前  $j$  项积,  $\text{power}$  为  $b^{2^j} \text{ mod } m$  即第  $j+1$  项

终止: 第  $k+1$  次迭代前,  $x$  为  $\prod_{i=0}^{k-1} (b^{2^i})^{a_i} \text{ mod } m = b^n \text{ mod } m$ ,  $\text{power}$  为  $b^{2^k} \text{ mod } m$

# Discrete

## Mathematics - P47

原码 (signed magnitude representation), 这种表示法中数字的正负号由一位 sign bit 表示，其余位表示数字的 magnitude (或绝对值 absolute value)。  
如一个 8-bit sign-magnitude 的二进制数的表示范围是  $-127(1111111) \sim +127(0111111)$   
注意在原码表示中，0 有两种表示，分别是 positive zero to (00000000), negative zero -0 (10000000)

1的补码 (ones' complement) 或称反码，与原码相同，用一位 sign bit 表示数字的正负号。

但是对于 magnitude，在取负数时，对 magnitude 部分按位取反，即取补码  
ones' complement form of a negative binary number is the bitwise NOT applied to it  
i.e. the "complement" of positive counterpart.

与原码相比，原码取相反数对 sign bit 按位取反，而反码对整个二进制数之按位取反

但 1 的补码的运算要比原码简单，但是要注意反码中的 end-around borrow

end-around borrow 指在进行整数减法后，超出字外的借位，需要在运算后从结果中减去

the borrow extends past the end of the word

end-around carry 指在进行整数加法后，超出字外的进位，需要在运算后加到结果中去。

the carry extends past the end of the word

wrapped around 指这种在运算后将超出的进位和借位加入结果或从结果中减去

$$\text{如: } 19(00010011) - (-3)(11111100) = (1) \underset{\substack{\text{borrow} \\ \downarrow}}{23(00010111)} - 1 = 22(000010110)$$

$$22(00010110) + (-0)(11111111) = (1) \underset{\substack{\text{carry} \\ \downarrow}}{22(00010101)} + 1 = 22(00010110)$$

2的补码 (two's complement)，是常用于计算机算术的整数二进制表示法。

n-bit 2's complement 的表示范围是  $[-2^{n-1}, 2^{n-1}-1]$ ，且不存在 negative zero (-0)

2的补码取反有两种方法，一是对二进制正数按位取反后 +1，即  $-2: 1101 + 1 = 1110$

另一种方法是，先找到二进制从右向左的第 1 位 1，并将其左侧的部分按位取反

即  $44(00101100) \xrightarrow{\text{negation}} 00101100 \xrightarrow{\text{左侧按位 NOT}} 11010100 (-44)$

注意由于  $0(00000000)$  中没有 1，所以不用变化， $-128(10000000)$  的第 1 位 1 左侧为空，无变化  
另外，虽然 2 的补码的最左一位也用于指示正负号，但其值为 0

但注意与原码、反码不同，最左一位 signbit 并非指定

而是由 2 的补码自行形成。

在无符号下 n-bit binary 表示  $\{0, 1, \dots, 2^n - 1\}$

而在 2 的补码下则表示  $\{0, 1, \dots, 2^{n-1} - 1, -2^{n-1}, -2^{n-1} + 1, \dots, -1\}$

注意这两种表示是用一个完全剩余系来替代了一个最小剩余系，即有一个一一对应的映射。

于是定义在最小剩余系上的模加法与模乘法可以直接应用在 2 的补码上，而无需像反码一样

$$\text{即 } 4(0100) + 7(0111) = 11(1011) \rightarrow -5(1011)$$

$$3(0011) \times 7(0111) = 21(10101) \rightarrow 5(0101)$$

考虑 end-around  
carry / borrow

# Discrete

## Mathematics - P48

二进制编码的十进制形式，指通过 4-bit 二进制编码每个十进制数字，从而为十进制整数之编码。

如:  $(791)_0 \rightarrow (0111\ 1001\ 0001)_2$  (二进制编码的十进制形式)

康托尔展开 (Cantor expansion) 指对于正整数  $m$ , 可以分解为  $m = a_n \cdot n! + \dots + a_2 \cdot 2! + a_1 \cdot 1!$ , 其中  $a_i \in \mathbb{Z}$  且  $0 \leq a_i \leq i$   
即  $\forall m \in \mathbb{Z}^+ \exists a_1, a_2, \dots, a_n \in \mathbb{N} (\bigwedge_{i=1}^n 0 \leq a_i \leq i) \wedge (a_n \neq 0) \wedge (\sum_{i=1}^n a_i \cdot i! = m)$

即对于每个正整数  $m$ , 其康托尔展开是唯一的。

procedure Cantor expansion (正整数  $m$ ) 这种算法利用康托尔展开的另一种形式

$q := m$

$k := 1$

while  $q \neq 0$

$a_k := q \bmod (k+1)$

$q := q \text{ div } (k+1)$

$k := k+1$

return  $(a_{k-1}, a_{k-2}, \dots, a_1)$

等价于第  $k$  位

按照  $k+1$  进制

计算

$a_k$  为  $k+1$  进制数

CantorExpan :: Int  $\rightarrow$  [Int]  $\rightarrow$  [Int]

CantorExpan 0 acc = acc

CantorExpan m acc = acc 为  $k+1$  进制

let i = length(acc) + 2 即第  $k+1$  次迭代

in CantorExpan (m `div` i) ((m `mod` i) : acc)

procedure Cantor add (正整数  $A = (a_0 a_1 \dots a_n)$ ,  $B = (b_0 b_1 \dots b_n)$ , 均为  $n$  位康托尔展开)

$c := 0$

for  $j := 1$  to  $n$

for  $j := 1$  to  $n$

$d := L(a_j + b_j + c) / (j+1)$

$s_{j+1} := L(a_j + b_j + s_j) / (j+1)$

$s_j := a_j + b_j + c - (j+1) \cdot d$

$s_j := a_j + b_j + s_j - s_{j+1} \cdot (j+1)$

$c := d$

return  $(s_{n+1}, s_n, \dots, s_1)$

$s_{n+1} := c$  但是注意这种方法会让加法进位变得模糊

return  $(s_{n+1}, s_n, \dots, s_1)$

### 素数

(prime), 对于大于 1 的整数  $P$ , 如果  $P$  的正因子只有  $1$  和  $P$ , 则称  $P$  为素数

即有  $P$  为素数  $\leftrightarrow (P \in \mathbb{Z}^+ \wedge P > 1 \wedge \forall n \in \mathbb{Z}^+ (1 < n < P \rightarrow n \nmid P))$

### 合数

(composite), 对于大于 1 的整数  $P$ , 若  $P$  不是素数, 则  $P$  为合数

即有  $P$  为素数  $\leftrightarrow (P \in \mathbb{Z}^+ \wedge P > 1 \wedge \exists n \in \mathbb{Z}^+ (1 < n < P \wedge n \mid P))$

算术基本定理 (fundamental theorem of arithmetic) 或称唯一分解定理 (unique factorization theorem)

对于每个大于 1 的正整数  $P$ ,  $P$  要么是素数, 要么可以唯一地写成素数的乘积

即有  $\forall P \in \mathbb{Z}^+ (P > 1 \rightarrow (P \text{ 是素数} \vee \exists! (p_1^{n_1} \cdots p_k^{n_k}) (\bigwedge_{i=1}^k p_i \text{ 为素数} \wedge \bigwedge_{i=1}^k n_i \in \mathbb{Z}^+ \wedge P = p_1^{n_1} \cdot p_2^{n_2} \cdots p_k^{n_k}))$

注意一般地有素数因子以非递减序排列, 即有  $n = p_1 p_2 \cdots p_n$ , 其中  $p_1 \leq p_2 \leq \cdots \leq p_n$  且皆为素数

# Discrete Mathematics - P49

如果  $n$  是一个合数，则  $n$  必有一个素因子小于或等于  $\sqrt{n}$  其中  $a$  为素数  
 即有，对于  $n \in \mathbb{Z}^+$  且  $n > 1$ ,  $\exists a \in \mathbb{Z}^+ (a < n \wedge a | n) \rightarrow (\exists a \in \mathbb{Z}^+ (a \leq \sqrt{n} \wedge a | n))$   
 证明过程有，如果  $n$  为合数，则存在  $a \in \mathbb{Z}^+$  且  $1 < a < n$  有  $a | n$ .  
 即存在  $b \in \mathbb{Z}^+$ , 使得  $n = ab$ .  
 假设  $a > \sqrt{n} \wedge b > \sqrt{n}$ , 则有  $ab > \sqrt{n} \cdot \sqrt{n} = n$ . 与  $n = ab$  矛盾  
 于是有  $a \leq \sqrt{n} \vee b \leq \sqrt{n}$   
 由此可以得到更有效的素数检验方法.

即对于正整数  $p > 1$ ,  $p$  是素数  $\leftrightarrow \forall n \in \mathbb{Z}^+ (1 < n \leq p \rightarrow n \nmid p)$

注意以模运算的次数度量这个算法的时间复杂度(最坏情形)为  $\Theta(\sqrt{n})$   
 而依素数定义构造的算法时间复杂度为  $\Theta(n)$

埃拉托斯特尼筛法 (sieve of Eratosthenes) 用于寻找不超过一个给定整数的所有素数

procedure Eratosthenes (正整数  $n$ )

集合  $N = \{x \in \mathbb{Z}^+ \mid 1 < x \leq n\}$  ] 初始化  $N$  为  $\mathbb{Z}^+$  的正整数集合  
 $P = \emptyset$  ]  $P$  为空集

while  $N \neq \emptyset$  ] 只要  $N$  非空, 就从  $N$  中取最小元素  $q$

$q = N$  中最小的元素 ]  $q$  为  $N$  中最小元素且  $q$  为素数

$P = P \cup \{q\}$  ] 将  $q$  加入集合  $P$

$N = \{x \in N \mid q \nmid x\}$  ] 从  $N$  中删除可被  $q$  整除的元素

return  $P$  {  $P$  为不大于  $n$  的素数的集合 }

循环不变量:  $\forall k, m \in \mathbb{Z}^+ (k \in P \wedge m \in N \rightarrow k$  是素数  $\wedge k < m \wedge k \nmid m)$

初始化: 在第1次迭代开始前,  $P$  为空集, 于是有循环不变量平凡地为真, 此时  $N$  中最小元素 2 为素数

保持: 在第  $j$  次迭代开始前,  $P$  中已提取的  $j-1$  个素数,  $N$  中元素皆不可为  $P$  中元素整除.

则在第  $j+1$  次迭代开始前,  $P$  中已提取的  $j$  个素数, 提取该元素加入  $P$ , 并从  $N$  中删除可被该元素整除的元素

则在第  $j+1$  次迭代开始前,  $P$  中已提取的  $j$  个素数,  $\forall k, m \in \mathbb{Z}^+ (k \in P \wedge m \in N \rightarrow k$  是素数  $\wedge k < m \wedge k \nmid m)$

终止: 在第  $|P| + 1$  次迭代开始前,  $N$  为空集, 循环不变量平凡地为真,  $P$  为从  $[2, n]$  中提取的  $|P|$  个素数

list Prime :: Int → [Int]

list Prime n =

let check =  $\lambda m \rightarrow (m \bmod k) \neq 0$

prime l acc

| null l = reverse acc

| otherwise = prime (filter (check (head l)) l)

| l = filter (check (chead l)) (acc : acc)

in prime [2..n] [

def listPrime(n):

l = list(range(2, n+1))

p = []

while len(l) > 0:

k = l[0]

p.append(k)

l = [i for i in l if i % k != 0]

return p

# Discrete

## Mathematics - P50

素数的无限性 指存在无限多个素数，或者说对于n个最小的素数，总能找到一个更大的素数不在其中。 $(n \in \mathbb{Z}^+)$

证明过程 假设只有有限多个素数  $P_1, P_2, \dots, P_n$

则令  $Q = P_1 P_2 \dots P_n + 1$ , 可知  $Q \in \mathbb{Z}^+$  且  $Q > 1$

根据算术基本定理,  $Q$  或者是素数, 或者可以写为两个或多个素数的乘积

如果  $Q$  为素数, 又  $\forall i \leq n, P_i < Q$ , 则  $Q$  不在有限个素数  $\{P_1, P_2, \dots, P_n\}$  之中

如果  $Q$  不是素数, 则  $\exists p \in \mathbb{Z}^+ (P \text{ 是素数} \wedge P \mid Q)$

又  $\forall i \leq n (Q \bmod P_i = 1)$  即  $\forall i \leq n P_i \nmid Q$ . 于是有  $\forall i \leq n P_i \neq P$

则素数  $P$  不在有限个素数  $\{P_1, P_2, \dots, P_n\}$  之中

是可知与假设只有有限个素数矛盾, 即存在无限多个素数

注意: 这是一个非构造性的存在性证明, 即并没有显式地给出一个不在  $\{P_1, P_2, \dots, P_n\}$  之中的素数  
非构造性的存在性证明通常以下形式呈现.

得到某一个分情形讨论的所有情形均可独立地蕴含存在性为真的结论.

即有一个分情形讨论如  $\bigvee_{i=1}^n P_i \equiv T$ , 又有  $\bigwedge_{i=1}^n (P_i \rightarrow q) \equiv T$

又  $\bigwedge_{i=1}^n (P_i \rightarrow q) \equiv \bigwedge_{i=1}^n (\neg P_i \vee q) \equiv (\bigwedge_{i=1}^n \neg P_i) \vee q \equiv \neg (\bigvee_{i=1}^n P_i) \vee q$

则有  $FVq \equiv T$ , 于是有  $q \equiv T$ . 即P存在性为真

梅森素数 (Mersenne prime). 指形如  $2^P - 1$  的素数, 其中  $P$  为素数

但是注意: 形如  $2^P - 1$  ( $P$  为素数) 的正整并非全是素数, 如  $2^{11} - 1 = 2047 = 23 \times 89$

卢卡斯-莱默尔测试 (Lucas-Lehmer primality test), 专门用于梅森素数的素性测试,  $M_p = 2^p - 1$ , 其中  $p$  大于2的素数

测试 定义序列  $\{S_n\}$  有  $S_0 = 4$ ,  $\forall n \in \mathbb{Z}^+, S_n = S_{n-1}^2 - 2$

则对于  $M_p = 2^p - 1$ , 其中  $p$  为素数, 有  $M_p$  是素数当且仅当  $S_{p-2} \equiv 0 \pmod{M_p}$

即有对于  $M_p = 2^p - 1$  且  $p$  为素数,  $M_p$  是素数  $\longleftrightarrow M_p \mid S_{p-2}$  ( $S_{p-2}$  为序列  $\{S_n\}$  中的项)

lucasLehmer :: Int  $\rightarrow$  Bool

lucasLehmer p =

let mp =  $2^p - 1$  ]  $M_p$  为待检验的梅森数

$S_0' = S_0 \bmod M_p$

test s<sub>i</sub> k | k == 0 = si

$S_1' = S_1 \bmod M_p$

| otherwise = test ((si  $\wedge 2 - 2) \bmod mp) (k-1)$

$= (S_0'^2 - 2) \bmod M_p$

in (test . 4 . (p-2)) == 0 ] 判断  $S_{p-2} \equiv 0 \pmod{M_p}$

$= (S_0'^2 - 2) \bmod M_p$

特别地有: 如果  $n$  为合数, 则  $2^n - 1$  也是合数, 其中  $n \in \mathbb{Z}^+$  且  $n > 1$

证明过程有, 令  $n = ab$ , 其中  $a, b \in \mathbb{Z}^+$  且  $a, b > 1$ ,

则有  $2^{ab} - 1 = 2^{ab} + 2^{ab-1} - 2^{ab-2} - \dots + 2^a - 2^a - 1 = 2^a (2^{ab-a} + \dots + 1) - (2^{ab-a} + \dots + 2^a + 1)$

$= (2^a - 1)(2^{ab-a} + 2^{ab-2} + \dots + 2^a + 1)$ , 即有  $2^n - 1$  为合数

注意其逆否命题为 如果  $2^n - 1$  是素数, 则  $n$  也是素数, 其中  $n \in \mathbb{Z}^+$  且  $n > 1$

# Discrete

## Mathematics - P51

素数分布

当  $x$  无限增长时, 不超过  $x$  的素数个数与  $x/\ln x$  之比趋近于 1

或者有对于  $x \in \mathbb{Z}^+$ , 不超过  $x$  的素数个数  $N(x) \sim (x/\ln x)$

又有当选择一个足够大的正整数  $n$ ,  $n$  是素数的概率大约是  $1/\ln n$ , ( $n$  为随机选择)

算术级数 (arithmetic series), 指的是形如  $\{an+b\}$  的序列, 其中  $a, b \in \mathbb{Z}$

则对于算术级数  $\{an+b\}$ , 如果有  $a, b \in \mathbb{Z}^+$  且  $\forall n \in \mathbb{Z}^+ (n > 1 \rightarrow n \nmid a \vee n \nmid b)$

则这个算术级数包含有无限多个素数

对于任意大于 2 的整数  $n$ , 存在完全由素数构成的长度为  $n$  的算术级数

即有  $\forall n \in \mathbb{Z}^+ (n > 2 \rightarrow \exists a, b \in \mathbb{Z}^+ \forall i \in \mathbb{Z}^+ (1 \leq i \leq n \rightarrow ai+b \text{ 是素数}))$

如  $f(x) = x^2 - x + 41$ , 有性质如  $\forall x \in \mathbb{Z}^+ (1 \leq x \leq 40 \rightarrow f(x) \text{ 是素数})$

则考察多项式是否可能有性质.  $\forall x \in \mathbb{Z}^+ P(x)$  是素数. ( $P(x)$  为  $x$  的整系数多项式)

证伪过程有, 假设存在一个整系数多项式  $f(n)$ , 使得  $(\forall x \in \mathbb{Z}^+ f(x) \text{ 是素数})$  为真

则有对于  $x_0 \in \mathbb{Z}^+$ ,  $f(x_0) = P$  是素数, 其中  $f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_0$ .  $n \in \mathbb{Z}^+$

对对于  $k \in \mathbb{N}$ ,  $x_0 + kp \in \mathbb{Z}^+$ ,

$$\text{则有 } f(x_0 + kp) = a_n (x_0 + kp)^n + a_{n-1} (x_0 + kp)^{n-1} + \dots + a_1 (x_0 + kp) + a_0$$

$$= a_n x_0^n + a_{n-1} x_0^{n-1} + \dots + a_1 x_0 + a_0 + M p$$

$$= f(x_0) + M p = (M+1)p$$

其中  $M$  为由  $x_0, k, p$  组成的  $x_0$  的  $n$  次多项式, 且有  $M \in \mathbb{Z}$

于是可知  $\forall k \in \mathbb{N} (P \mid f(x_0 + kp) \wedge f(x_0 + kp) \text{ 是素数})$

即  $\forall k \in \mathbb{N} f(x_0 + kp) = P$

但是  $n$  次多项式 ( $n > 1$ ) 在同一个值最多取  $n$  次,

即至多有  $n$  个不同的  $k$  值, 使得  $f(x_0 + kp) = P$

由此产生矛盾, 进而可知假设不成立

即不存在一个非常量整系数多项式  $f(n)$ , 使得  $\forall n \in \mathbb{Z}^+ f(n)$  是素数

哥德巴赫猜想 (Goldbach's conjecture), every even integer greater than 2 can be expressed as sum of two primes

即有  $\forall n \in \mathbb{Z}^+ (12 \mid n \wedge n > 2 \rightarrow \exists a, b \in \mathbb{Z}^+ (a \text{ 是素数} \wedge b \text{ 是素数} \wedge n = a + b))$

目前弱一点的结论有: 每个大于 2 的偶数都可以表示成至多 6 个素数之和

以及对每个充分大的偶数, 可写成一个素数与另一个数之和, 这个数或者是素数, 或者是两个素数之积

猜想存在无限多个正整数  $n$ , 使得  $n^2 + 1$  为素数

目前弱一点的结论有: 存在无限多个  $n \in \mathbb{Z}^+$ , 使得  $n^2 + 1$  或者是素数, 或者是两个素数之积

# Discrete

## Mathematics - P52

孪生素数 (twin prime), 即对于形如  $P$  和  $P+2$  的一对正整数,  $P$  和  $P+2$  都是素数

有猜想为: 有无限多个正整数  $P$ , 使得  $P$  和  $P+2$  均为素数

目前弱一点的结论为: 有无限多个正整数  $P$ , 使得  $P$  为素数,  $P+2$  或者是素数, 或者是两个素数的乘积

最大公约数 (greatest common divisor), 用  $\gcd(a, b)$  表示对于不全为 0 的整数  $a, b$ , 能整除  $a$  和  $b$  的最大整数

即对于  $a, b \in \mathbb{Z}$  且  $a \neq 0 \vee b \neq 0$ , 有整数  $d = \gcd(a, b)$  使得  $d | a \wedge d | b \wedge \forall n \in \mathbb{Z} (n > d \rightarrow (n \nmid a \vee n \nmid b))$

互素整数 (relatively prime integers), 如果对于整数  $a, b$ , 有最大公约数为 1, 则称  $a$  和  $b$  是互素的

即有对于  $a, b \in \mathbb{Z}$  且  $a \neq 0 \vee b \neq 0$ ,  $a, b$  互素  $\leftrightarrow \gcd(a, b) = 1$

另外当  $a=0$ ,  $a, b$  互素  $\leftrightarrow |b|=1$

当  $|a|=1$ ,  $a, b \in \mathbb{Z}$   $a, b$  是互素的

特别地有, 通常讨论  $a, b$  是否互素, 是在  $|a| \neq |b|$  的情形下

两两互素 (pairwise relatively prime integers), 指任何两个整数都是互素的一组整数

即有对于  $a_1, a_2, \dots, a_n \in \mathbb{Z}$ ,  $a_1, \dots, a_n$  两两互素  $\leftrightarrow \forall i, j \in \mathbb{Z} (1 \leq i < j \leq n \rightarrow \gcd(a_i, a_j) = 1)$

可以通过素因子分解式求两个整数的最大公约数.

假如对  $n$  个不同的素数  $p_1, p_2, \dots, p_n$  正整数  $a, b$  有素因子分解式

其中  $a = p_1^{a_1} p_2^{a_2} \dots p_n^{a_n}$ ,  $b = p_1^{b_1} p_2^{b_2} \dots p_n^{b_n}$ , 其中  $a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_n \in \mathbb{N}$

则  $\gcd(a, b) = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \dots p_n^{\min(a_n, b_n)}$

证明过程有, 令  $d = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \dots p_n^{\min(a_n, b_n)}$

则  $a/d = p_1^{a_1 - \min(a_1, b_1)} p_2^{a_2 - \min(a_2, b_2)} \dots p_n^{a_n - \min(a_n, b_n)}$

可知  $\forall 1 \leq i \leq n a_i - \min(a_i, b_i) \geq 0$ , 即  $p_i^{a_i - \min(a_i, b_i)}$  为整数

于是  $a/d = \prod_{i=1}^n p_i^{a_i - \min(a_i, b_i)}$  为整数, 即  $d | a$ , 同理可知  $d | b$ .

而  $\forall m > d$  或者存在一个不在  $p_1, \dots, p_n$  中的素数  $p_{n+1}$ , 有  $p_{n+1} | m$ .

或者存在一个  $i \in \mathbb{Z}^+$ , 有  $a_i - m_i < 0 \vee b_i - m_i < 0$ .

所以对于  $a, b \in \mathbb{Z}^+$ , 有  $\gcd(a, b) = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \dots p_n^{\min(a_n, b_n)}$

最小公倍数 (least common multiple), 用  $\text{lcm}(a, b)$  表示对于非零整数  $a, b$ , 可被  $a, b$  整除的最小正整数

即对于  $a, b \in \mathbb{Z}$  且  $a \neq 0 \wedge b \neq 0$ , 有正整数  $d = \text{lcm}(a, b)$

使得  $a | d \wedge b | d \wedge \forall n \in \mathbb{Z} (n < d \rightarrow (a \nmid n \vee b \nmid n))$

且有对  $n$  个不同素数  $p_1, p_2, \dots, p_n$ , 和素因子分解式的正整数  $a, b$  有  $\text{lcm}(a, b) = p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \dots p_n^{\max(a_n, b_n)}$

有  $\text{lcm}(a, b) = p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \dots p_n^{\max(a_n, b_n)}$

# Discrete

mathematics

## Mathematics - P53

对于正整数  $a, b$ , 有  $ab = \gcd(a, b) \cdot (\text{lcm}(a, b))$

证明过程有, 对于正整数  $a, b$ , 可知存在  $n$  个不同的素数  $p_1, p_2, \dots, p_n$  使得  $a, b$  可以表示为由  $p_1, p_2, \dots, p_n$  组成的素因子分解式

即有  $a = p_1^{a_1} p_2^{a_2} \dots p_n^{a_n}$ , 其中  $a_1, a_2, \dots, a_n \in \mathbb{N}$

$b = p_1^{b_1} p_2^{b_2} \dots p_n^{b_n}$ , 其中  $b_1, b_2, \dots, b_n \in \mathbb{N}$

则  $ab = p_1^{a_1+b_1} p_2^{a_2+b_2} \dots p_n^{a_n+b_n}$

对于  $1 \leq i \leq n$ , 有  $a_i + b_i = \min(a_i, b_i) + \max(a_i, b_i)$

即  $\forall 1 \leq i \leq n, p_i^{a_i+b_i} = p_i^{\min(a_i, b_i) + \max(a_i, b_i)}$

则  $ab = p_1^{\min(a_1, b_1) + \max(a_1, b_1)} p_2^{\min(a_2, b_2) + \max(a_2, b_2)} \dots p_n^{\min(a_n, b_n) + \max(a_n, b_n)}$

$= p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \dots p_n^{\min(a_n, b_n)} \cdot p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \dots p_n^{\max(a_n, b_n)}$

$= \gcd(a, b) \cdot (\text{lcm}(a, b))$

辗转相除法 又称欧几里得算法 (Euclidean algorithm), 用于求整数间的最大公约数

其过程为通过连续使用除法, 直至某次除法余数为0, 以求最大公约数

引理 有整数  $a, b, q, r$  使得  $a = bq + r$ , 则有  $\gcd(a, b) = \gcd(b, r)$

注意有  $a, b$  不同时为0, 否则  $\gcd(a, b)$  无意义

而当  $a, b$  不同时为0时,  $b, r$  也不同时为0,

当  $b \neq 0$  时, 该论平凡地为真,

当  $a \neq 0, b=0$  时, 有  $a = 0 \cdot q + r = r$ , 于是  $r = a \neq 0$ .

引理 证明有: 对于  $a, b, q, r \in \mathbb{Z}$ , 且  $a \neq 0 \vee b \neq 0$ , 有  $a = bq + r$

则对于任意整数  $d \neq 0$ , 如果有  $d \mid a \wedge d \mid b$ , 即  $d$  为  $a, b$  的公约数

有  $\exists m \in \mathbb{Z}, a = md, \exists n \in \mathbb{Z}, b = nd$ .

于是有  $r = a - bq = md - nqd = cm - nqd \mid d$ .

又  $q \in \mathbb{Z}$ , 则有  $d \mid r$ , 即  $d$  为  $b, r$  的公约数.

所以  $\forall d \in \mathbb{Z} \wedge d \neq 0$  ( $d$  是  $a, b$  的公约数  $\rightarrow d$  是  $b, r$  的公约数)

另有 对于任意整数  $c \neq 0$ , 如果有  $c \mid b \wedge c \mid r$ , 即  $c$  为  $b, r$  的公约数

有  $\exists m \in \mathbb{Z}, b = mc, \exists n \in \mathbb{Z}, r = nc$ .

于是有  $a = bq + r = mq + nc = (mq + nc)c$

又  $q \in \mathbb{Z}$ , 则有  $c \mid a$ , 即  $c$  为  $a, b$  的公约数

所以  $\forall c \in \mathbb{Z} \wedge c \neq 0$  ( $c$  是  $b, r$  的公约数  $\rightarrow c$  是  $a, b$  的公约数)

合并可知  $\forall d \in \mathbb{Z} \wedge d \neq 0$  ( $d$  是  $a, b$  的公约数  $\leftrightarrow d$  是  $b, r$  的公约数)

即有  $\gcd(a, b) = \gcd(b, r)$

# Discrete

## Mathematics - P54

欧几里得算法 已知有对于整数  $a, b, q, r$  且  $a \neq 0 \vee b \neq 0$ , 使得有  $a = bq + r$ , 则有  $\gcd(a, b) = \gcd(b, r)$

又当  $a=0$  或  $b=0$  时,  $\gcd(a, b)$  为非零的数

则假定  $a, b \in \mathbb{Z}^+$ , 且  $a \geq b$ , 则如果取  $0 \leq r < a, b$ , 则有  $r = a - bq = a \bmod b$

于是令  $r_0 = a, r_1 = b$ , 则有迭代关系

$$r_0 = r_1 q_1 + r_2, 0 \leq r_2 < r_1, \text{ 即 } r_2 = r_0 \bmod r_1$$

$$r_1 = r_2 q_2 + r_3, 0 \leq r_3 < r_2, \text{ 即 } r_3 = r_1 \bmod r_2$$

直到  $r_n = 0$  为止

$$r_{n-2} = r_{n-1} q_{n-1} + r_n, 0 \leq r_n < r_{n-1}, \text{ 即 } r_n = r_{n-2} \bmod r_{n-1}$$

$$r_{n-1} = r_n q_n, r_{n+1} = 0, \text{ 即 } r_{n+1} = r_{n-1} \bmod r_n = 0$$

$$\begin{aligned} \text{于是有 } \gcd(a, b) &= \gcd(r_0, r_1) = \gcd(r_1, r_2) = \dots \\ &= \gcd(r_{n-2}, r_{n-1}) = \gcd(r_{n-1}, r_n) = \gcd(r_n, 0) = r_n \end{aligned}$$

特别注意: 由于迭代终止于某个  $r_{n+1}=0$ , 所以应有  $\forall 0 \leq i \leq n \quad r_i > 0$

procedure gcd( $a, b \in \mathbb{Z}^+$ )

$x := a$  [ 初始化参数  $x, y$ , 并且  $y \neq 0$  ]

$y := b$  [ 注意这里不需要进行  $a > b$  的判断 ]

while  $y \neq 0$  do [ 当  $y \neq 0$  时, 应用  $\gcd(a, b) = \gcd(b, a \bmod b)$  ]

$r := x \bmod y$  [ 应用  $\gcd(a, b) = \gcd(b, a \bmod b)$  ]

$x := y$  [ 并更新  $x, y$  的值 ]

$y := r$  [ 注意如果  $x < y$ , 则仅交换  $x, y$  的值 ]

return  $x$  [  $\gcd(a, b) = x$  ] [ 当  $y=0$  时,  $\gcd(a, b) = \gcd(x, 0) = x$  ]

注意: 辗转相除法的循环不变量为  $\gcd(a, b) = \gcd(x, y)$

初始化: 在第1次迭代前,  $x=a \wedge y=b$ , 所以  $\gcd(a, b) = \gcd(x, y)$  显然成立

保持: 在第  $k$  次迭代前, 若有  $\gcd(x_k, y_k) = \gcd(a, b)$  成立

则在第  $k+1$  次迭代中,  $x$  更新为  $y$ ,  $y$  更新为  $x \bmod y$

于是 在第  $k+1$  次迭代前,  $\gcd(x_{k+1}, y_{k+1}) = \gcd(y_k, x_k \bmod y_k)$

$= \gcd(x_k, y_k) = \gcd(a, b)$

终止: 迭代终止于  $y=0$  时, 假设此时为第  $n+1$  次迭代前

则有  $\gcd(a, b) = \gcd(x_n, y_n) = \gcd(x_n, 0) = x_n$

欧几里得算法源于欧几里得(Euclid)的《几何原本》(The Elements)

辗转相除法源于《九章算术》, 正式见于秦九韶的《数书九章》

# Discrete

## Mathematics - P55

贝祖定理 (Bézout's theorem) 指对于正整数  $a, b$ , 存在整数  $s, t$ , 使得  $\gcd(a, b) = sa + tb$

即有  $\forall a, b \in \mathbb{Z}^+ \exists s, t \in \mathbb{Z}$   $\gcd(a, b) = sa + tb$

贝祖恒等式 (Bézout identity), 指  $\gcd(a, b) = sa + tb$ , 其中  $a, b \in \mathbb{Z}^+, s, t \in \mathbb{Z}$

贝祖系数 (Bézout coefficient), 指对于  $a, b \in \mathbb{Z}^+$ , 使  $\gcd(a, b) = sa + tb$  成立的整数  $s, t$

通过正整数集合子集的良序性证明 贝祖定理:

假设对于正整数  $a, b$ , 定义集合  $S = \{sa + tb > 0 \mid s, t \in \mathbb{Z}\}$

首先有集合  $S \neq \emptyset$ , 因为集合  $S$  至少包含元素  $a+b$  和  $a$

然后由  $a, b \in \mathbb{Z}^+, s, t \in \mathbb{Z}$ , 可知 当  $sa + tb > 0$  时, 有  $sa + tb \in \mathbb{Z}^+$

即集合  $S$  为正整数集合  $\mathbb{Z}^+$  的一个子集

于是根据良序性公理 集合  $S$  必有最小元素  $c = s_0a + t_0b$

令整数  $d$  为  $a, b$  的公因子, 即有  $d | a \wedge d | b$ , 且  $d \neq 0$

于是有对于整数  $s_0, t_0$ ,  $d | (s_0a + t_0b)$  即  $d | c$

即有  $\forall d \in \mathbb{Z}$  ( $d | a \wedge d | b \rightarrow d | c$ )

考虑  $c$  与  $a, b$ . 假设  $c | a, c | b, a, b \in \mathbb{Z}^+$ ,

则存在  $d \in \mathbb{Z}, r \in \mathbb{Z}^+$  且  $0 < r < c$ , 使得  $a = dc + r$

又  $c = s_0a + t_0b$ , 于是有  $r = (1 - ds_0)a - dt_0b$ .

又  $r \in \mathbb{Z}^+$ , 则根据集合  $S$  的定义应有  $r \in S$ ,

但  $r < c$ , 于是与  $c$  为  $S$  的最小元素矛盾.

于是有  $c | a$ , 同理可知  $c | b$

如果有整数  $k$  使得  $k > c \wedge k | a \wedge k | b$ , 即如果存在大于  $c$  的  $a, b$  公因子

则根据  $\forall d \in \mathbb{Z}$  ( $d | a \wedge d | b \rightarrow d | c$ ), 又  $k > c > 0$ , 则  $k | c$ , 产生矛盾

于是不存在与  $c$  大于的  $a, b$  的公因子,  $c = \gcd(a, b)$

所以对于两个正整数, 有唯一的最大公因数

且存在整数  $s, t$ , 使得  $\gcd(a, b) = sa + tb$

求解 通常求贝祖系数的过程为在执行欧几里得算法后反向执行另一个迭代.

如:  $r_2 = a - q_1b \rightarrow r_n = s_0a + t_0b$  其中  $s_0 = -t_2$   
 $r_3 = b - q_2r_2 \rightarrow r_n = s_1b + t_2r_2$  其中  $t_1 = s_1 - t_2q_1$  且  $s_0, t_0 \in \mathbb{Z}$

$r_{n-1} = r_{n-3} - q_{n-2}r_{n-2} \rightarrow r_n = s_{n-3}r_{n-3} + t_{n-2}r_{n-2}$  其中  $s_{n-3} = -q_{n-1}$   
 $t_{n-2} = 1 + q_{n-1}q_{n-2}$

$(r_n | r_{n-1}) \quad r_n = r_{n-2} - q_{n-1}r_{n-1}$

$\therefore r_{n+1} = q_n r_n$  于是有  $\gcd(252, 198) = 18$

如  $54 = 252 - 1 \times 198 \rightarrow 18 = 4 \times 252 - 5 \times 198$  且  $18 = 4 \times 252 - 5 \times 198$

$36 = 198 - 3 \times 54 \rightarrow 18 = 4 \times 54 - 198$

且  $18 = 54 - 1 \times 36$

并且  $18 = 54 - 1 \times 36$

# Discrete

## Mathematics - P56

扩展欧几里得算法(Extended Euclidean algorithm) 用于对整数  $a, b$ , 在求  $\gcd(a, b)$  的同时求出 贝祖系数

即对于整数  $a, b$  且  $a \neq 0 \vee b \neq 0$ , 求出整数  $d, s, t$ , 使得  $d = \gcd(a, b) = sa + tb$

注意当  $a=0$  时  $\gcd(a, b)=b$ , 当  $b=0$  时  $\gcd(a, b)=a$

又  $\gcd(a, b) = \gcd(|a|, |b|)$

于是简起见仅讨论  $a, b$  为正整数的情形.

基本步骤为, 令  $s_0=1, t_0=0, s_1=0, t_1=1$ . 令  $r_0=a, r_1=b$ ,

在第  $k$  步, 如果  $r_k \neq 0$ , 则取  $r_{k+1} = r_{k-1} \bmod r_k, q_k = r_{k-1} \text{ div } r_k$

$s_{k+1} = s_{k-1} - q_k s_k, t_{k+1} = t_{k-1} - q_k t_k$

如果  $r_k = 0$ , 则返回  $\gcd(a, b) = r_{k-1} = s_{k-1}a + t_{k-1}b$

以矩阵形式表示. 使用矩阵初等变换, 直到第  $n$  次迭代后  $r_{n+1}=0$ .

即  $\begin{pmatrix} r_{k-1} & s_{k-1} & t_{k-1} \end{pmatrix} \xrightarrow{q_k = r_{k-1} \text{ div } r_k} \begin{pmatrix} r_{k-1} - q_k r_k & s_{k-1} - q_k s_k & t_{k-1} - q_k t_k \end{pmatrix} \rightarrow \begin{pmatrix} r_k & s_k & t_k \end{pmatrix}$   
 在第  $k$  次迭代  $\begin{pmatrix} r_k & s_k & t_k \end{pmatrix} \xrightarrow{r_{k-1} \text{ div } r_k} \begin{pmatrix} r_k & s_k & t_k \end{pmatrix}$

procedure ExtendedEuclidean (正整数  $a, b$ )

$[r_0 = a, s_0 = 1, t_0 = 0]$  ] 初始化  $\begin{pmatrix} a & 1 & 0 \end{pmatrix}$   
 $[r_1 = b, s_1 = 0, t_1 = 1]$  ] 矩阵  $\begin{pmatrix} b & 0 & 1 \end{pmatrix}$

while  $r_i \neq 0$  ] 当  $r_i \neq 0$  时, 则进行下一次迭代

$q = r_0 \text{ div } r_i$  ] 更新矩阵为, 其中  $q = r_0 \text{ div } r_i$

$[r_0 = r_i, s_0 = s_i, t_0 = t_i]$  ]  $\begin{pmatrix} r_i & s_i & t_i \end{pmatrix}$   
 $[r_1 = r_0 \bmod r_i, s_1 = s_0 - q s_i, t_1 = t_0 - q t_i]$  ]  $\begin{pmatrix} r_0 \bmod r_i & s_0 - q s_i & t_0 - q t_i \end{pmatrix}$

return  $(r_0, s_0, t_0)$   $\{r_0 = \gcd(a, b) = s_0a + t_0b\}$

循环不变量为  $(\gcd(r_0, r_1) = \gcd(a, b)) \wedge (r_0 = s_0a + t_0b) \wedge (r_1 = s_1a + t_1b)$

初始化: 由  $r_0 = a, r_1 = b$  可知  $\gcd(r_0, r_1) = \gcd(a, b)$  平凡地真,  $r_0 = s_0a + t_0b$  且  $r_1 = s_1a + t_1b$  也为真

保持: 第  $k$  次迭代开始前有  $(\gcd(r_0, r_1) = \gcd(a, b)) \wedge (r_0 = s_0a + t_0b) \wedge (r_1 = s_1a + t_1b)$

由于迭代完成后  $r'_0 = r_1, s'_0 = s_1, t'_0 = t_1, r'_1 = r_0 \bmod r_1$ , 即有  $r'_0 = s'_0a + t'_0b$  为真

而  $\gcd(r'_0, r'_1) = \gcd(r_1, r_0 \bmod r_1) = \gcd(r_0, r_1) = \gcd(a, b)$

而  $q = r_0 \text{ div } r_1$ , 则有  $r'_1 = r_0 - q r_1 = (s_0a + t_0b) - q(s_1a + t_1b)$

$= (s_0 - q s_1)a + (t_0 - q t_1)b = s'_0a + t'_0b$

终止: 当  $r_{n+1}=0$  时循环终止, 有  $\gcd(a, b) = \gcd(r_n, 0) = r_n$ , 且  $r_n = s_n a + t_n b$

extendEuclid :: Int → Int → (Int, Int, Int) (Haskell 实现)

extendEuclid a b = let iter r0 s0 t0 r1 s1 t1 in

$| r_1 == 0 \quad = (r_0, s_0, t_0)$

$| \text{otherwise} \quad = \text{let } q = r_0 \text{ div } r_1$

in iter r1 s1 t1 (r0 `mod` r1) (s0 - q \* s1) (t0 - q \* t1)

in iter a1 b1 r1 s1 t1 in iter a1 b1 r1 s1 t1