离散数学

--数论和密码学

数论基础、数论算法、密码算法

2025年9月16日 星期二

总体概要

除法和模运算

质数和最大公约数

求解同余式

同余式的应用

密码学

整数除法和模算术运算

整除

定义:如果 a 和 b 是整数，且 $a \neq 0$，那么如果存在一个整数 c，使得 b=ac，则称 a 整除 b，记为$a \mid b$

① 当a整除 b时，我们称a是b的因数或除数，并且称 b是a的倍数

② 如果 $a \mid b$，那么 b/a 是一个整数

③ 如果 a 不整除 b，我们记作 $a \nmid b$

- **例题：**判断 3|7 和 3|12 是否成立。

整除性质

定理 1: *a*, *b*, *c* 都是整数, 且 *a* ≠0.

i. 【线性组合性】如果 *a* | *b* 并且 *a* | *c*, 那么 *a* | (*b* + *c*);

ii. 如果 *a* | *b*，那么对于所有的整数 *c*, *a* | b*c* ;

iii. 【传递性】如果 *a* | *b* 并且 *b* | *c*, 那么 *a* | *c*.

iv. 【大小关系】如果*a* | *b* , b ≠0,那么 |a|≤|b|

证明： (i) 假设 a | b 且 a | c ，则存在整数 s 和 t ，使得 b = as 和 c = at 。所以有 b + c = as + at = a(s + t) 。因此， a | (b + c)

推论：若 a, b, c 是整数，且 a≠0，且 $a|b$ 和 $a|c$，则对任意整数 m和 n，$a|mb+nc$。

练习: 请证明，如果c|a-b, c|a'-b', 那么 c|aa'-bb'

除法算法

当一个整数被一个正整数除时，会得到一个商和一个余数。这通常被称为“除法算法”（Division Algorithm），但实际上它是一个定理。

除法算法:如果 a是一个整数，b 是一个正整数，那么存在唯一的整数 q 和 r，使得 $0 \leq r < b$，并且 $a = bq + r$.

- b 称为除数.
- a 称为被除数.
- q 称为商.
- r 称为余数.

Definitions of Functions
div and **mod**

$q = a$ **div** b

$r = a$ **mod** b

例题:

- 当 101 被 11 除时，商和余数是多少？
  - 解答：当 101 被 11 除时，商为 9=101 div 11，余数为 2=101 mod 11.
- 当 −11 被 3 除时，商和余数是多少？
  - 解答：当 −11被 3 除时，商为 −4=−11 div 3−，余数为 1=−11 mod 3.

存在性的证明

良序原理(Well-Ordering Principle): 非负整数集合的任意非空子集都有一个最小元素

思考：良序原理和良序有什么联系？

Existence of q and r. Let

$$S = \{a - bk : k \in \mathbb{Z} \text{ and } a - bk \geq 0\}.$$

If $0 \in S$, then b divides a, and we can let $q = a/b$ and $r = 0$. If $0 \notin S$, we can use the Well-Ordering Principle. We must first show that S is nonempty. If $a > 0$, then $a - b \cdot 0 \in S$. If $a < 0$, then $a - b(2a) = a(1 - 2b) \in S$. In either case $S \neq \emptyset$. By the Well-Ordering Principle, S must have a smallest member, say $r = a - bq$. Therefore, $a = bq + r$, $r \geq 0$. We now show that $r < b$. Suppose that $r > b$. Then

$$a - b(q + 1) = a - bq - b = r - b > 0.$$

In this case we would have $a - b(q+1)$ in the set S. But then $a - b(q+1) < a - bq$, which would contradict the fact that $r = a - bq$ is the smallest member of S. So $r \leq b$. Since $0 \notin S$, $r \neq b$ and so $r < b$.

唯一性的证明

Uniqueness of q and r. Suppose there exist integers r, r', q, and q' such that

$$a = bq + r,\ 0 \leq r < b$$

and

$$a = bq' + r',\ 0 \leq r' < b.$$

Then $bq + r = bq' + r'$. Assume that $r' \geq r$. From the last equation we have $b(q - q') = r' - r$; therefore, b must divide $r' - r$ and $0 \leq r' - r \leq r' < b$. This is possible only if $r' - r = 0$. Hence, $r = r'$ and $q = q'$. □

Congruence Relation(同余关系)

定义:如果 a 和 b 是整数，且 m 是正整数，那么当 m 整除 $a-b$ 时，称 a 与 b 在模 m 下同余，记作 $a \equiv b \pmod{m}$.

- 符号 $a \equiv b \pmod{m}$ 表示 a 与 b 在模 m 下同余.
- 我们称 $a \equiv b \pmod{m}$ 为一个同余关系，且 m 是其模数.
- 两个整数在模 m 意义下同余当且仅当它们除以 m 后的余数相同.
- 如果 a 不与 b 在模 m 意义下同余，我们记作 $a \not\equiv b \pmod{m}$.

例题:判断 17 是否与 5 在模 6 意义下同余，24 和 14 是否在模 6 意义下同余.

解答:

- $17 \equiv 5 \pmod{6}$ 因为 6 整除 $17 - 5 = 12$.
- $24 \not\equiv 14 \pmod{6}$ 因为 $24 - 14 = 10$ 不被 6整除.

More on Congruences

定理 2:设 m 为正整数。当且仅当存在一个整数 k 使得 $a=b+km$，整数 a 和 b 在模 m 意义下同余。

证明:

- 如果 $a \equiv b \pmod{m}$，则根据同余的定义，m 整除 $a-b$。因此，存在一个整数 k，使得 $a-b=km$，即等价于 $a=b+km$.
- 反过来，如果存在一个整数 k，使得 $a=b+km$，那么 $km=a-b$。因此，m 整除 $a-b$，并且 $a \equiv b \pmod{m}$.

思考：同余关系具有什么性质？

More on Congruences

自反性: 任何正整数都和它自身同余 $a \equiv a \pmod{m}$

对称性: $a \equiv b \pmod{m} \implies b \equiv a \pmod{m}$

传递性: $a \equiv b \pmod{m}$, $b \equiv c \pmod{m} \implies a \equiv c \pmod{m}$

(mod m) 和 mod m 符号关系

在 $a \equiv b \pmod{m}$ 和 $a \bmod m = b$ 中，“mod”的用法是不同的.

- $a \equiv b \pmod{m}$ 是整数集上的一个关系，表示 a 和 b 在模 m 意义下同余.
- 在 $a \bmod m = b$ 中，"mod" 表示一个函数，该函数返回 a 除以 m 后的余数.

这两个符号之间的关系在以下定理中得到明确说明.

定理 3:设 a 和 b 为整数，m 为正整数，则 $a \equiv b \pmod{m}$ 当且仅当 $a \bmod m = b \bmod m$.

同余式的加和乘

定理 4:设 m 为正整数。如果 $a \equiv b \pmod{m}$ 且 $c \equiv d \pmod{m}$，则 $a+c \equiv b+d \pmod{m}$，$ac \equiv bd \pmod{m}$且$a^k \equiv b^k \pmod{m}$ 其中k是非负整数

证明:

- 因为 $a \equiv b \pmod{m}$ 且 $c \equiv d \pmod{m}$，根据定理 2，存在整数 s 和 t，使得 $b=a+sm$和 $d=c+tm$。
- 因此,
  - $b + d = (a + sm) + (c + tm) = (a + c) + m(s + t)$
  - $b\,d = (a + sm)(c + tm) = ac + m(at + cs + stm)$.
- 因此, $a + c \equiv b + d \pmod{m}$ and $ac \equiv bd \pmod{m}$.

例题: 因为 $7 \equiv 2 \pmod{5}$ 并且 $11 \equiv 1 \pmod{5}$,根据定理 4，可以得出

$$18 = 7 + 11 \equiv 2 + 1 = 3 \pmod{5}$$

$$77 = 7 \cdot 11 \equiv 2 \cdot 1 = 2 \pmod{5}$$

Algebraic Manipulation of Congruences (同余式的代数运算)

将有效同余的两边同时乘以一个整数会保持其有效性.

- 如果 $a \equiv b \pmod{m}$，那么 $c \cdot a \equiv c \cdot b \pmod{m}$ 也成立，其中 c是任意整数，这是通过定理 4 的 $d=c$ 得出的.

将一个整数加到有效同余的两边也会保持其有效性.

- 如果 $a \equiv b \pmod{m}$ 成立，那么 $c+a \equiv c+b \pmod{m}$ 也成立，其中 c 是任意整数，这也是通过定理 4 的 $d=c$得出的.

然而，将同余两边同时除以一个整数并不总能产生有效的同余关系.

- **例子**:同余 $14 \equiv 8 \pmod{6}$ 成立。但是将两边同时除以 2 并不会产生有效的同余，因为 14/2=7和 8/2=4，但 $7 \not\equiv 4 \pmod{6}$.

计算mod m 函数的和与积

定理4推论:设 m 为正整数，a 和 b 为整数. Then

$(a + b)$ (**mod** m) = ((a **mod** m) + (b **mod** m)) **mod** m

and

ab **mod** m = ((a **mod** m) (b **mod** m)) **mod** m.

思考：如何计算 2^{644} **mod 645**？

二进制模幂算法

在密码学中, 能够高效计算 b^n **mod** m 是非常重要的

使用n的二进制展开, $n = (a_{k-1},\ldots,a_1,a_0)_2$, 来计算b^n .

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

这个算法依次寻找b **mod** m, b^2 **mod** m, b^4 **mod** m, ..., 并且将它们相乘起来当 $a_j = 1$ 时.

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 }

$O((\log m)^2 \log n)$ bit operations are used to find b^n **mod** m.

二进制模幂算法

计算 $3^{644} \bmod 645$

644的二进制展开式为 $(1010000100)_2$， k-1 = 9

首先令x=1, power= 3 **mod** 645 =3.

然后通过连续地取平方并模645来更新power值，即power: = (power • power) mod 645

a_i是645的二进制展开式的第i位.

如果a_j=1，就用x当前值乘以power 并模645来更新x值，即x : = (x • power) mod 645

具体过程如下:

二进制模幂算法

计算 $3^{644} \bmod 645$

644的二进制展开式为 $(1010000100)_2$， k-1 = 9

i=0: 因为a_0=0,所以有x=1, power=$3^2 \bmod 645=9$

i=1: 因为a_1=0,所以有x=1, power=$9^2 \bmod 645=81$

i=2: 因为a_2=1,所以有x=$1 \cdot 81 \bmod 645$, power=$81^2 \bmod 645=111$

i=3: 因为a_3=0,所以有x= 81, power=$111^2 \bmod 645=66$

i=4: 因为a_4=0,所以有x= 81, power=$66^2 \bmod 645=486$

i=5: 因为a_5=0,所以有x= 81, power=$486^2 \bmod 645=126$

二进制模幂算法

计算 $3^{644} \bmod 645$

644的二进制展开式为 $(1010000100)_2$， k-1 = 9

i=5: 因为a_5=0,所以有x= 81, power=$486^2 \bmod 645=126$

i=6: 因为a_6=0,所以有x=81, power=$126^2 \bmod 645=396$

i=7: 因为a_7=1,所以有x=$(81 \cdot 396) \bmod 645=471$,power=$396^2 \bmod 645=81$

i=8:因为a_8=0,所以有x=471,power=81^2 mod 645=111

i=9:因为a_9=1,所以有x=$(471 \cdot 111) \bmod 645=36$

根据以上步骤, 可以得出结果为$3^{644} \bmod 645=36$.

二进制模幂算法

练习：计算 $2^{644} \bmod 645$

$(644)_{10}=(1010000100)_2$

i=0: 因为a_0=0,所以有x=1, power=$2^2 \bmod 645=4$
i=1: 因为a_1=0,所以有x=1, power=$4^2 \bmod 645=16$
i=2: 因为a_2=1,所以有x=1 • 16 mod 645, power=$16^2 \bmod 645=256$
i=3: 因为a_3=0,所以有x= 16, power=$256^2 \bmod 645=391$
i=4: 因为a_4=0,所以有x= 16, power=$391^2 \bmod 645=16$
i=5: 因为a_5=0,所以有x= 16, power=$16^2 \bmod 645=256$
i=6: 因为a_6=0,所以有x=16, power=$256^2 \bmod 645=391$
i=7: 因为a_7=1,所以有x=(16 • 391) mod 645=451, power=$391^2 \bmod 645=16$
i=8: 因为a_8=0,所以有x=451, power=$16^2 \bmod 645=256$
i=9: 因为a_9=1,所以有x=(451 • 256) mod 645=1
根据以上步骤, 可以得出结果为$3^{644} \bmod 645=1$.

模m算术运算[1]

定义:给定 Zm 是一个包含从 0到 m–1的非负整数的集合，即 {0,1,…,m–1}

- 加法 $+_m$ 被定义为 $a +_m b = (a + b)$ **mod** m. 这是模m加法.
- 乘法$\cdot_m$ 被定义为$a \cdot_m b = (a \cdot b)$ **mod** m. 这是模m乘法.
- 进行这些运算被称为模 m 的算术运算.

例题: 计算 $7 +_{11} 9$ 和 $7 \cdot_{11} 9$.

解答: 使用上面的定义:

- $7 +_{11} 9 = (7 + 9)$ **mod** $11 = 16$ **mod** $11 = 5$
- $7 \cdot_{11} 9 = (7 \cdot 9)$ **mod** $11 = 63$ **mod** $11 = 8$

模m算术运算 2

算术模运算 $+_m$ and $\cdot_m$满足许多与普通加法和乘法相同的性质

- **封闭性:** 如果a 和 b *属于* $\mathbf{Z}_m$, 那么 $a +_m b$ 和 $a \cdot_m b$ 属于 $\mathbf{Z}_m$.
- **结合律:** 如果 $a, b,$ *和* c 属于 $\mathbf{Z}_m$, 那么 $(a +_m b) +_m c = a +_m (b +_m c)$ 以及 $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.
- **交换律:** 如果 a 和 b 属于$\mathbf{Z}_m$, 那么$a +_m b = b +_m a$ 以及 $a \cdot_m b = b \cdot_m a$.
- **单位元:** 0 和 1 分别是模加法和模乘法的单位元.
  - 如果 a 属于 $\mathbf{Z}_m$, 那么 $a +_m 0 = a$ 以及$a \cdot_m 1 = a$.

模m算术运算3

- **加法逆元:** 如果 $a \neq 0$ 属于 $\mathbf{Z}_m$，那么 $m - a$ 是a的模m加法逆元，0是它自身的模m加法逆元.
  - $a +_m (m - a) = 0$ and $0 +_m 0 = 0$
- **分配律:** 如果 $a, b,$ 和 c 属于 $\mathbf{Z}_m$，那么
  - $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$ 以及 $(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$.

乘法逆元在模 m 的运算中不总是存在。例如，2 在模 6 的情况下没有乘法逆元，因为没有整数 x 使得 $2 \cdot x \equiv 1 \bmod 6$.

质数与最大公约数

小节概要

质数与它们的性质

关于质数的一些猜想和开放性问题

最大公约数与最小公倍数

欧几里得与扩展欧几里得算法

最大公约数的线性组合表示

质数（素数）

定义: 设 p 是大于 1 的正整数，如果 p 的正因子只有 1 和 p，那么称 p 为素数或质数 ;否则，称 p 为合数.

例:整数 7 是质数，因为它的正因数只有 1 和 7；但 9 是合数，因为它可以被 3 整除.

The Fundamental Theorem of Arithmetic (算术基本定理)

定理:每一个大于 1 的正整数都可以唯一地表示为一个素数，或者表示为两个或更多素数的乘积，其中素因数以非递减序排列.

例子:

- $641 = 641$
- $100 = 2 \cdot 2 \cdot 5 \cdot 5 = 2^2 \cdot 5^2$
- $999 = 3 \cdot 3 \cdot 3 \cdot 37 = 3^3 \cdot 37$
- $1024 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^{10}$

如何判断一个整数是不是素数？

如何找出一个整数素因子分解式？

试除法

定理: 如果a是合数， 则a必有小于等于$\sqrt{a}$的素因子

证： $a=bc$, 其中$1<b<a$, $1<c<a$. 显然, b和c中必有一个小于等于$\sqrt{a}$. 否则, $bc>(\sqrt{a})^2=a$, 矛盾.

试除法

例子 判断157和161是否是素数.

解 $\sqrt{157}$, $\sqrt{161}$都小于13, 小于13的素数有: 2, 3, 5, 7, 11.

检查结果如下:

$2 \nmid 157, 3 \nmid 157, 5 \nmid 157, 7 \nmid 157, 11 \nmid 157$

结论: 157是素数.

$2 \nmid 161, 3 \nmid 161, 5 \nmid 161, 7 \mid 161$（$161=7\times23$）

结论：161是合数.

试除法

例子 找出7007的素因子分解式

解 $7007 < 84^2$，

首先不断地用素数去除7007，从2开始。2、3和5都除不尽7007。但是，7除尽7007，7007/7 = 1001。

下一步，从7开始不断地用素数去除1001。立刻发现7还能整除1001，即1001/7 = 143。

继续从7开始不断用素数去除143。11整除143，得143/11= 13 由于13为素数，这一过程完成。

$$7007 = 7 \cdot 1001 = 7 \cdot 7 \cdot 143 = 7 \cdot 7 \cdot 11 \cdot 13 = 7^2 \cdot 11 \cdot 13$$

The Sieve of Erastosthenes (埃拉托斯特尼筛法)

试除法是一种判定整数 n 是否为素数的方法，其过程是尝试每一个小于或等于 $\sqrt{n}$ 的整数 i，并查看 n 是否能被 i 整除.

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

Erastothenes
(276-194 B.C.)

The Sieve of Erastosthenes (埃拉托斯特尼筛法)

例子 寻找不超过100的素数

Integers divisible by 2 other than 2 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 3 other than 3 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

The Sieve of Erastosthenes (埃拉托斯特尼筛法)

例子 寻找不超过100的素数

Integers divisible by 5 other than 5 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 7 other than 7 receive an underline; integers in color are prime.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

The Fundamental Theorem of Arithmetic (算术基本定理)

定理: 每一个大于 1 的正整数都可以唯一地表示为一个素数，或者表示为两个或更多素数的乘积，其中素因数以非递减序排列.

定理: 设a>1, 则$a=p_1^{r1}p_2^{r2}\cdots p_k^{rk}$, 其中$p_1, p_2, \cdots, p_k$是互不相同的素数， $r_1, r_2, \cdots, r_k$ 是正整数，并且在不计顺序的情况下，该表示是唯一的.

定理中的表达式称作整数 a 的素因子分解

例子： $7007 = 7^2 \cdot 11 \cdot 13$

The Fundamental Theorem of Arithmetic (算术基本定理)

推论: 设$a=p_1^{r1}p_2^{r2}\cdots p_k^{rk}$, 其中$p_1, p_2,\cdots,p_k$ 是互不相同的素数， $r_1, r_2, \cdots, r_k$ 是正整数，则正整数d为a的因子的充分必要条件是$d=p_1^{s1}p_2^{s2}\cdots p_k^{sk}$, 其中$0\le s_i\le r_i$, $i=1,2,\cdots,k$

例1：21560有多少个正因子?

解：$21560=2^3\times5\times7^2\times11$

因此 21560的正因子的个数为$4\times2\times3\times2=48$.

The Fundamental Theorem of Arithmetic (算术基本定理)

例2：10!的二进制表示中从最低位数起有多少个连续的0?

解：2, 3, 4=2^2, 5, 6=2×3, 7, 8=2^3, 9=32, 10=2×5.

10! = $2^8 \times 3^4 \times 5^2 \times 7$

故10!的二进制表示中从最低位数起有8个连续的0

练习：20!的二进制表示中从最低位数起有多少个连续的0?

Infinitude of Primes
（无穷素数）

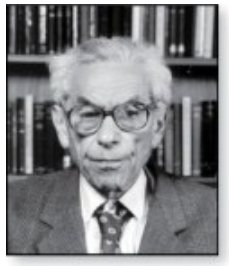
Euclid （欧几里得）
(325 B.C.E. – 265 B.C.E.)

定理: 存在无限多个素数.

证明: 假设素数的数量是有限的: $p_1, p_2, \ldots, p_n$

- Let $q = p_1 p_2 \cdots p_n + 1$
- 要么 q 是素数，要么根据算术基本定理，它是素数的乘积.
  - *但是没有任何一个* p_j 整除 q 因为如果 $p_j \mid q$, 那么 p_j 整除 $q - p_1 p_2 \cdots p_n = 1$.
  - 因此，有一个素数不在$p_1, p_2, \ldots, p_n$. 这个素数要么是 q，要么如果 q 是合数，它就是 q 的一个质因数.这与假设 $p_1,p_2,\ldots,p_n$是所有质数的集合相矛盾.
- 因此，素数的个数是无限的.

这个证明由欧几里得在《几何原本》（“The Elements”）中给出，被认为是所有数学证明中最美丽的之一。它是“天书”（The Book）中的第一个证明，这一概念灵感来自著名数学家保罗·埃尔德什（Paul Erdős）设想的完美证明集



Paul Erdős
(1913-1996)

素数的函数表示

Marin Mersenne
(1588-1648)

定义: 形如 $2^p - 1$, 其中 p 是素数，被称为梅森素数（Mersenne primes）.

- $2^2 - 1 = 3$, $2^3 - 1 = 7$, $2^5 - 1 = 37$, and $2^7 - 1 = 127$ 都是梅森素数.
- $2^{11} - 1 = 2047$ 不是梅森素数，因为$2047 = 23 \cdot 89$.
- 有一种高效的方法来确定$2^p - 1$ 是否是素数.
- 已知最大的素数是梅森素数.
- 截至 2018年早期，已发现 50 个梅森素数，其中最大的一个是 $2^{77,232,917} - 1$，这是一个有2300 万位的十进制数字
- 大互联网梅森素数搜索（Great Internet Mersenne Prime Search，GIMPS）是一个分布式计算项目，旨在寻找新的梅森素数。http://www.mersenne.org/

当p是合数时, 2^p-1一定是合数，$2^{ab}-1=(2^a-1)(2^{a(b-1)}+2^{a(b-2)}+\ldots+2^a+1)$.

素数的分布

数学家们一直对素数在正整数中的分布感兴趣。在 19 世纪，素数定理（Prime Number Theorem）被证明了，该定理给出了不超过 x 的素数个数的渐近估计.

素数定理:不超过 x 的素数个数与 $x/\ln x$的比率随着 x 的增大而趋近于 1. (其中 $\ln x$ 是 x 的自然对数)

- 该定理得知，不超过 x 的质数个数可以用 $x/\ln\ x$来近似.
- 随机选择一个小于 n 的正整数是素数的概率大约是 $(n/\ln n)/n = 1/\ln n$.

n	10^3	10^4	10^5	10^6	10^7
$\pi(n)$	168	1 229	9 592	78 498	664 579
$\frac{n}{\ln n}$	145	1 086	8 686	72 382	620 421
$\frac{\pi(n)}{n/\ln n}$	1.159	1.132	1.104	1.085	1.071

生成素数

生成大素数的问题在理论和实践中都具有重要意义.

我们将看到，找到有数百位数字的大素数在密码学中非常重要.

到目前为止，还没有找到一种总能生成素数的有用闭合公式。没有一个简单的函数 f(n) 能够使 f(n) 对所有正整数 n 都为素数.

然而，$f(n) = n^2 - n + 41$ 对于所有整数 1, 2, …, 40 都是素数。因为这个原因，我们可能会推测 f(n) 对所有正整数 n 都是素数。但事实上，$f(41) = 41^2$ 不是素数.

更一般地说，没有一个具有整数系数的多项式 f(n) 使得 f(n) 对所有正整数 n 都为素数.

Conjectures about Primes (关于素数的猜想)

尽管素数已经被广泛研究了几个世纪，关于它们的许多猜想仍未解决，包括:

哥德巴赫猜想：每个大于 2 的偶整数 n 都是两个素数之和。该猜想已经通过计算机验证了所有不超过 $4 \cdot 10^{18}$的正偶整数（截止2018年）。多数数学家相信这一猜想是正确的.

孪生素数猜想：孪生素数猜想认为存在无穷多对孪生素数。孪生质数是指相差 2 的素数对。例子包括 3 和 5，5 和 7，11 和 13 等。截至 2011 年中，发现的最大的孪生质数对是 $65{,}516{,}468{,}355 \cdot 2^{33{,}333} \pm 1$,它们有 100,355 位十进制数字.

Greatest Common Divisor 1 (最大公约数)

定义:设 a 和 b 为整数，且不全为零。能够同时整除 a 和 b 的最大整数 d 称为 a 和 b 的最大公约数，记作 gcd(a,b).

例: gcd(24,36)=12 , gcd(17,22)=1

最大公约数2

定义:如果两个整数 a 和 b 的最大公约数为 1，则称 a 和 b 是互素的

例子1: 17 和 22

定义: 整数$a_1, a_2, \ldots, a_n$ 是两两互素的，如果当 $1 \le i<j \le n$时有 $\gcd(a_i, a_j) = 1$.

例子2:判断整数 10、17 和 21 是否两两互素

解: 因为 gcd(10,17) = 1, gcd(10,21) = 1, and gcd(17,21) = 1, 所以10, 17, 和21 两两互素.

例子3:判断整数 10、19 和 24 是否两两互素

解: 因为 gcd(10,24) = 2, 所以10, 19, 和 24 不是两两互素

用素因子分解式找最大公约数

假设 a 和 b 的素因数分解是:

$$a = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}, \qquad b = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n},$$

其中每个指数都是非负整数，且两个素因数分解式中出现的所有素数都包含在两者中. 那么:

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

这个公式是有效的，因为等号右边的整数可以同时整除 a 和 b。没有比它更大的整数能够同时整除 a 和 b.

例子: $120 = 2^3 \cdot 3 \cdot 5$ $500 = 2^2 \cdot 5^3$

解: $\gcd(120,500) = 2^{\min(3,2)} \cdot 3^{\min(1,0)} \cdot 5^{\min(1,3)} = 2^2 \cdot 3^0 \cdot 5^1 = 20$

使用素因数分解式来求两个正整数的最大公约数并不高效，因为目前没有高效的算法来找出正整数的素因数分解.

Least Common Multiple (最小公倍数)

定义:正整数 a 和 b 的最小公倍数是同时能被 a 和 b 整除的最小正整数，记作 lcm(a,b).

最小公倍数也可以通过素因数分解来计算.

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

这个数字可以被 a和 b 同时整除，并且没有更小的数字可以同时被 a 和 b 整除.

例子: lcm($2^3 3^5 7^2$, $2^4 3^3$) = $2^{\max(3,4)}\ 3^{\max(5,3)}\ 7^{\max(2,0)} = 2^4\ 3^5\ 7^2$

两个整数的最大公约数和最小公倍数之间的关系是:

定理: a，b为正整数. 那么ab = gcd(a,b) ·lcm(a,b)

Euclidean Algorithm[1] (欧几里得算法)

欧几里得算法是一种高效计算两个整数最大公约数的方法。其基于的理念是，当 $a>b$ 且 c 是 a 除以 b 的余数时，gcd (a,b) 等于 gcd (b,c).

Euclid
(325 B.C.E. – 265 B.C.E.)

例子: 找到 gcd(91, 287):

- $287 = 91 \cdot 3 + 14$ Divide 287 by 91
- $91 = 14 \cdot 6 + 7$ Divide 91 by 14
- $14 = 7 \cdot 2 + 0$ Divide 14 by 7

Stopping condition

gcd(287, 91) = gcd(91, 14) = gcd(14, 7) = 7

Euclidean Algorithm[1] (欧几里得算法)

欧几里得算法用伪代码表示如下:

```
procedure gcd(a, b: positive integers)
x := a   y := b
while   y ≠ 0
      r := x mod y
      x := y
      y := r
return x {gcd(a,b) is x}
```

欧几里得算法的正确性[1]

引理: 令 $a = bq + r$, 其中 a, b, q, 和 r 都是整数. 那么 $\gcd(a,b) = \gcd(b,r)$.

证明:

- 假设 d 同时整除 a 和 b。那么 d 也整除 $a-bq=r$。因此，a 和 b 的任何公约数也必须是 b 和 r 的公约数.
- 假设 d 同时整除 b 和 r。那么 d 也整除 $bq+r=a$。因此，b 和 r 的任何公约数也必须是 a 和 b 的公约数.
- 因此, $\gcd(a,b) = \gcd(b,r)$.

欧几里得算法的正确性2

假设 a 和 b 是正整数且 $a \geq b$。设 $r_0=a$和 $r_1=b$。通过连续应用除法算法，我们得到如下结果:

$$
\begin{aligned}
r_0 &= r_1 q_1 + r_2 && 0 \leq r_2 < r_1, \\
r_1 &= r_2 q_2 + r_3 && 0 \leq r_3 < r_2, \\
&\vdots \\
r_{n-2} &= r_{n-1} q_{n-1} + r_n && 0 \leq r_n < r_{n-1}, \\
r_{n-1} &= r_n q_n.
\end{aligned}
$$

$$
\begin{aligned}
2415 &= 945 \cdot 2 + 525 \\
945 &= 525 \cdot 1 + 420 \\
525 &= 420 \cdot 1 + 105 \\
420 &= 105 \cdot 4 + 0.
\end{aligned}
$$

最终，余数为零会出现在序列中: $a = r_0 > r_1 > r_2 > \cdots \geq 0$.这个序列中最多不能包含超过 a 项

根据引理1

$\gcd(a,b) = \gcd(r_0,r_1) = \cdots = \gcd(r_{n-1},r_n) = \gcd(r_n, 0) = r_n$.

因此，最大公约数是这个除法序列中的最后一个非零余数.

最大公约数表示成一个线性组合

定理:如果 a 和 b 是任意整数，且不全为零，那么 gcd(a,b)是集合{$ax + by : x, y \in \mathbb{Z}$} 中的最小正整数元素.

证明:

设 s 是 a 和 b 的最小正线性组合.

设 q 是 a 除以 s 的商. 那么 **a mod s**=a-qs=a-q(ax+by)=a(1-qx)+b(-qy).

因此，a mod s是 a 和 b 的线性组合.

由于 s 是所有线性组合中的最小正数，并且0<=**a mod s** <s，

a mod s不能是正数. 因此 **a mod s**=0.

这意味着 s 是 a 的因子，同样 s 也是 b 的因子。因此 s 是 a 和 b 的公约数, gcd(a,b)>=s.

由于 gcd (a,b) 同时整除 a 和 b，且 s 是 a 和 b 的线性组合, 我们有 gcd(a,b) | s. 但是 gcd(a,b) | s 并且 s>0 意味着 gcd(a,b)<=s.

推论: 对于整数 a 和 b, 如果 d |a 并且 d|b, 那么 d|gcd(a,b).

最大公约数表示成一个线性组合

Étienne Bézout

(1730-1783)

Bézout's Theorem (贝祖定理):如果 a 和 b 是正整数，那么存在整数 s 和 t 使得 gcd $(a,b)=sa+tb$.

定义:如果 a 和 b 是正整数，那么使得 gcd $(a,b)=sa+tb$的整数 s 和 t 称为 a 和 b 的 贝祖系数.

根据 贝祖定理，整数 a 和 b 的最大公约数可以表示为 $sa+tb$，其中 s 和 t 是整数。这是 a 和 b 的一个线性组合，其系数为整数.

- gcd(6,14) = (−2)·6 + 1·14

求贝祖系数

例子: 将 gcd(252,198)=18 表示为 252 和 198 的线性组合.

解: 首先使用欧几里得算法证明 gcd(252,198)=18

i. 252 = 1·198 + 54
ii. 198 = 3 ·54 + 36
iii. 54 = 1 ·36 + 18
iv. 36 = 2 ·18

- 现在从上面第（iii）式和第（i）式反向推导
  - 18 = 54 − 1 ·36
  - 36 = 198 − 3 ·54
- 将第二个方程代入第一个方程中:
  - 18 = 54 − 1 ·(198 − 3 ·54)= 4 ·54 − 1 ·198
- 接着将 54=252−1·198 (由第(i)式得) 代入上式:
  - 18 = 4 ·(252 − 1 ·198) − 1 ·198 = 4 ·252 − 5 ·198

这个方法展示了“两步法”，先使用欧几里得算法找到最大公约数，然后通过回代的方式将最大公约数表示为原始两个整数的线性组合。还有一种称为扩展欧几里得算法的“单步法”，可以在进行欧几里得算法的同时贝祖系数，将最大公约数直接表示为线性组合(*扩展欧几里得算法*).

扩展欧几里得算法

$\gcd(a,b) = r_n$

设 $r_0=a$和 $r_1=b$

$$r_0 = r_1 q_1 + r_2$$

$$r_1 = r_2 q_2 + r_3$$

$$\vdots$$

$$r_{n-2} = r_{n-1} q_{n-1} + r_n$$

$$r_{n-1} = r_n q_n.$$

$s_0=1,\ s_1=0 \qquad t_0=0,\ t_1=1$

$s_i=s_{i-2}-s_{i-1}q_{i-1} \qquad t_i=t_{i-2}-t_{i-1}q_{i-1} \qquad (i= 1, 2, \dots ,$ n)

$as_i+bt_i = r_i \qquad (i= 1, 2, \dots , n)$

$as_n+bt_n = r_n$

已知 $\gcd(a,b) = r_n$

所以 $as_n+bt_n = \gcd(a,b)$

所以 $s= s_n, \qquad t=t_n$

扩展欧几里得算法

例子: 将 gcd(100,35) 表示为 100 和 35 的线性组合.

解: r_0=100 和 r_1=35

100 = 35 × 2 + 30

35 = 30 × 1 + 5

30 = 5 × 6

s_0=1, s_1=0 t_0=0, t_1=1

$s_2= s_0-s_1q_1= 1-0 \times 2 = 1$ $t_2= t_0-t_1q_1= 0-1 \times 2 = -2$

$s_3= s_1-s_2q_2= 0-1 \times 1 = -1$ $t_3= t_1-t_2q_2= 1-(-2) \times 1 = 3$

$gcd(100,35)= 100s_3+35t_3= 100 \times (-1) + 35 \times 3$

Dividing Congruences by an Integer

将有效同余式的两边同时除以一个整数，并不总是能得到一个有效的同余式。

但如果这个整数与模数互素，则可以得到一个有效的同余式:

定理 :设 m 为正整数，a、b 和 c 为整数. 如果 $ac \equiv bc \pmod{m}$ 并且 $\gcd(c,m) = 1$, 那么 $a \equiv b \pmod{m}$.

证明: 因为 $ac \equiv bc \pmod{m}$, 那么$m \mid ac - bc = c(a - b)$ 由于 $\gcd(c,m) = 1$, 所以 $m \mid a - b$. 因此, $a \equiv b \pmod{m}$.

算术基本定理的唯一性证明

引理 1: 如果a,b,c为正整数，gcd(a,b)=1且 a|bc，则a|c .

证明：ax+by=1, 两边同乘c，acx+bcy=c，则a|c.

引理 2: 如果素数$p|a_1a_2\cdots a_n$，其中$a_1,\cdots,a_n$都为正整数，对于某个i，$p|a_i$

证明: 归纳法 n=1，显然成立；设对n成立，证对n+1也成立，即证

如果$p|a_1a_2\cdots a_n\ a_{n+1}$，则$p|\ a_i$；(1)如果$gcd(p, a_1a_2\cdots a_n)=1$,则$p|a_{n+1}$；

(2)如果$gcd(p, a_1a_2\cdots a_n)=p$,即$p|a_1a_2\cdots a_n$，则$p|a_i$，得证.

证明：反证法

$n= p_1p_2\cdots p_s= q_1q_2\cdots q_t$， p_i和q_j均为素数且各自都是非降序排列；

再把中间相同的公因子去除，使得 $p_{i1}p_{i2}\cdots p_{iu}= q_{j1}q_{j2}\cdots q_{jv}$，根据上述引理 2，$p_{i1}\ |q_{jk}$，因为都为素数且不相同，导致不可能，得证。

求解同余方程

小节概要4

Linear Congruences(线性同余方程)

The Chinese Remainder Theorem(中国剩余定理)

Fermat's Little Theorem(费马小定理)

Pseudoprimes(伪素数)

Primitive Roots and Discrete Logarithms

(原根和离散对数)

Linear Congruences (线性同余式)

定义:形如 $ax \equiv b \pmod{m}$ 的同余式，其中 m 为正整数，a 和 b 是整数，x 是变量，称为线性同余式.

线性同余式 $ax \equiv b \pmod{m}$ 的解是所有满足该同余式的整数 x.

定义: 如果整数 $\bar{a}$ 满足$\bar{a}a \equiv 1 \pmod{m}$称 $\bar{a}$为 a 的模 m 的逆元.

例子: 5 是 3 模 7 的逆元，因为 $5 \cdot 3 = 15 \equiv 1 \pmod{7}$

求解线性同余式的一种方法是使用 a 的逆元 $\bar{a}$ （如果它存在）。虽然不能直接将同余式的两边除以 a，但可以通过乘以 $\bar{a}$ 来解 x.

a modulo m的逆元

以下定理保证了，当 a 和 m 互素时，a 在模 m 下的逆元存在。两个整数 a 和 b 当 gcd (a,b)=1时互素.

定理 1:如果 a 和 m 是互素的整数且 m>1，那么 a 在模 m 下有逆元。此外，这个逆元在模 m 下是唯一的。（这意味着存在一个唯一的正整数 $\bar{a}$ 小于 m，它是 a 在模 m 下的逆元，且 a 的任何其他模 m 下的逆元都与 $\bar{a}$ 在模 m 下同余

证明:由于 gcd (a,m)=1，根据贝祖定理，存在整数 s 和 t 使得:

- 因此, $sa + tm \equiv 1 \pmod{m}$.
- 由于 $tm \equiv 0 \pmod{m}$, 所以$sa \equiv 1 \pmod{m}$
- 因此，s 是 a 在模 m 下的逆元.
- 逆元的唯一性也可以被证明.

求解逆元1

欧几里得算法和贝祖系数为我们提供了一种系统的方法来寻找逆元.

例子:求 3 在模 7 下的逆元.

解:由于 gcd(3,7)=1，根据上页定理1，3 在模 7 下的逆元存在.

- 使用欧几里得算法: 7 = 2·3 + 1.
- 从这个等式，我们得到：−2·3+1·7=1,可以看出，−2 和 1 是 3 和 7 的贝祖系数.
- 因此，−2 是 3 在模 7 下的逆元.
- 同时，任何与 −2 在模 7 下同余的整数也是 3 的模 7 逆元，例如 5、−9、12 等.

求解逆元2

例子:求 101 在模 4620 下的逆元.

解:先使用欧几里得算法证明 gcd(101,4620)=1.

4620 = 45·101 + 75

101 = 1·75 + 26

75 = 2·26 + 23

26 = 1·23 + 3

23 = 7·3 + 2

3 = 1·2 + 1

2 = 2·1

因为最后一个非零余数是 1,
gcd(101,4260) = 1

贝祖系数 : – 35 and 1601

Working Backwards:

1 = 3 – 1·2

1 = 3 – 1·(23 – 7·3) = – 1 ·23 + 8·3

1 = –1·23 + 8·(26 – 1·23) = 8·26 – 9 ·23

1 = 8·26 – 9 ·(75 – 2·26)= 26·26 – 9 ·75

1 = 26·(101 – 1·75) – 9 ·75

= 26·101 – 35 ·75

1 = 26·101 – 35 ·(4620 – 45·101)

= – 35 ·4620 + 1601·101

101 在模 4620 下的逆元为 1601

用逆元求解线性同余式

我们可以通过两边同时乘以 $\bar{a}$ 来解方程 $ax \equiv b \pmod{m}$.

例子:解方程 $3x \equiv 4 \pmod{7}$

解:之前已经找到 -2 是 3 在模 7 下的逆元。我们将方程两边同时乘以 -2

$$-2 \cdot 3x \equiv -2 \cdot 4 \pmod{7}.$$

因为 $-6 \equiv 1 \pmod{7}$ 并且 $-8 \equiv 6 \pmod{7}$, 因此如果x 是解, 则有 $x \equiv -8 \equiv 6 \pmod{7}$

我们需要确定每一个满足 $x \equiv 6 \pmod{7}$ 的 x 是否都是方程的解。假设 $x \equiv 6 \pmod{7}$ ，可验证 $3x \equiv 3 \cdot 6 = 18 \equiv 4 \pmod{7}$，这表明所有这样的 x 都满足这个同余式.

因此，解集是所有满足 $x \equiv 6 \pmod{7}$ 的整数，即 $x = 6, 13, 20$或者 $x = -1, -8, -15$

The Chinese Remainder Theorem (中国剩余定理) 1

在公元一世纪，中国数学家孙子提出了以下问题：“有物不知其数，三分之余二，五分之余三，七分之余二，此物几何？”

这个谜题可以被转化为解以下同余方程组的问题:

$x \equiv 2 \pmod 3$,

$x \equiv 3 \pmod 5$,

$x \equiv 2 \pmod 7$

我们将看到一个被称为中国剩余定理的定理是如何用于解决孙子的这个问题的.

中国剩余定理2

定理 2: (*The Chinese Remainder Theorem*) 设 $m_1,m_2,\ldots,m_n$是两两互素的大于 1 的正整数并且$a_1,a_2,\ldots,a_n$ 是任意整数.对于以下同余方程组

$x \equiv a_1 \pmod{m_1}$

$x \equiv a_2 \pmod{m_2}$

.

.

.

$x \equiv a_n \pmod{m_n}$

此方程组有唯一的模 $m = m_1m_2 \cdots m_n$的解

(即，存在一个解 x，使得 $0 \le x < m$，且所有其他解均与该解模 m 同余.)

证明:我们将通过描述一种构造解的方法来证明解的存在性.

中国剩余定理3

为了构造解，首先令$M_k=m/m_k$ 对于 $k = 1,2,\ldots,n$ 以及 $m = m_1 m_2 \cdots m_n$.
因为 $\gcd(m_k, M_k) = 1$, 根据前述定理1, 存在整数y_k, 它是 M_k 在 模 m_k下的逆元， *使得*

$$M_k\, y_k \equiv 1 \pmod{m_k}.$$

然后构造和

$$x = a_1\, M_1\, y_1 + a_2\, M_2\, y_2 + \cdots + a_n\, M_n\, y_n .$$

注意到因为$M_j \equiv 0 \pmod{m_k}$ 当 $j \neq k$,所以在这个和中，除了第 k 项外，所有项模 m_k 都同余于 0 .
由于 $M_k\, y_k \equiv 1 \pmod{m_k}$, 我们有 $x \equiv a_k\, M_k\, y_k \equiv a_k \pmod{m_k}$, 对于 $k = 1,2,\ldots,n$.
因此，x 是以下 n 个同余方程的同时解.

$$x \equiv a_1 \pmod{m_1}$$
$$x \equiv a_2 \pmod{m_2}$$
$$\vdots$$
$$x \equiv a_n \pmod{m_n}$$

中国剩余定理解的唯一性证明

设有两两互质的正整数$n_1, n_2, ..., n_k$，以及任意整数$a_1, a_2, ..., a_k$，那么同余方程组

$$\begin{cases} x \equiv a_1 \pmod{n_1} \\ x \equiv a_2 \pmod{n_2} \\ \vdots \\ x \equiv a_k \pmod{n_k} \end{cases}$$

有唯一解模 $N = n_1 n_2 ... n_k$。

唯一性：要证明解的唯一性，假设存在两个解 x 和 y，它们都满足上述的同余方程组。那么我们有

$$x \equiv a_i \pmod{n_i} \quad 和 \quad y \equiv a_i \pmod{n_i} \quad 对于所有\ i = 1, 2, ..., k.$$

因此，对于所有的 i，我们有

$$x \equiv y \pmod{n_i}.$$

这意味着 n_i 整除 $x - y$ 对于所有的 i。

由于n_1,n_2…和n_k两两互质,且n_1|(x-y),n_2|(x-y)…,n_k|(x-y)，可证明N=(n_1*n_2*..*n_k) | (x-y)；

既然 N|(x−y)，这意味着 x≡y (mod N)。因此，任何两个解 x 和 y 在模 N 下都是相同的，这证明了解的唯一性。

中国剩余定理4

例:我们将应用中国剩余定理来解决孙子的问题:

$x \equiv 2 \pmod 3$, $x \equiv 3 \pmod 5$, $x \equiv 2 \pmod 7$.

- 令 $m = 3 \cdot 5 \cdot 7 = 105$, $M_1 = m/3 = 35$, $M_2 = m/5 = 21$, $M_3 = m/7 = 15$.
- 我们注意到
  - 2 是M_1 = 35 模 3的逆元 因为 $35 \cdot 2 \equiv 2 \cdot 2 \equiv 1 \pmod 3$
  - 1 是M_2 = 21 模 5的逆元 因为 $21 \equiv 1 \pmod 5$
  - 1 是M_3 = 15 模 7的逆元 因为 $15 \equiv 1 \pmod 7$
- 因此,

  $x = a_1M_1y_1 + a_2M_2y_2 + a_3M_3y_3$

  $= 2 \cdot 35 \cdot 2 + 3 \cdot 21 \cdot 1 + 2 \cdot 15 \cdot 1 = 233 \equiv 23 \pmod{105}$
- 所以，23 是满足所有方程的最小正整数解

反向替换方法

我们还可以通过将线性同余方程组中的每个同余方程转化为等式，代入变量的值到另一个同余方程中，并持续进行这个过程，直到解决所有同余方程，从而解决具有互素模数的线性同余方程组。这种方法称为回代法.

例:使用回代法找到所有满足以下条件的整数 x使得 $x \equiv 1 \pmod 5$, $x \equiv 2 \pmod 6$, and $x \equiv 3 \pmod 7$.

解:第一个同余方程可以重写为 $x=5t+1$，其中 t 是一个整数.

- 代入第二个同余方程得到$5t + 1 \equiv 2 \pmod 6$.
- 解这个方程得$t \equiv 5 \pmod 6$.
- 使用定理4得到 $t=6u+5$，其中 u 是一个整数.
- *代回*$x = 5t + 1$, 解得 $x = 5(6u + 5) + 1 = 30u + 26$.
- 再代入第三个方程得到$30u + 26 \equiv 3 \pmod 7$.
- 解得$u \equiv 6 \pmod 7$.
- 使用定理4得到 $u=7v+6$，其中 v 是一个整数.
- 将 $u=7v+6$ 代入 $x=30u+26$ 中得到,$x = 30(7v + 6) + 26 = 210u + 206$.

将这个结果转化为同余方程，我们得到解$x \equiv 206 \pmod{210}$.

Wilson Theorem (威尔逊定理)

Wilson定理：P是素数当且仅当 (p-1)! ≡ -1(mod *p*).

引理：如p为素数且 $x^2 \equiv 1 \pmod p$，则$x \equiv 1 \pmod p$或 $x \equiv -1 \pmod p$.

引理证明：$p \mid (x^2-1)$,即 $p \mid (x+1)(x-1)$,因p为素数,则 $p \mid (x+1)$或 $p \mid (x-1)$，得证。

Wilson定理证明：

① 先证当(p-1)! ≡ -1(mod *p*)时, p为素数：

设$n \mid p$且$n \neq p$，则$n \in \{1, 2, \cdots, p-1\}$，能推导出 $n \mid (p-1)!$；根据条件，

$p \mid ((p-1)!+1)$，因为$n \mid p$,则 推出 **$n \mid ((p-1)!+1)$**；又已推出 $n \mid (p-1)!$, 得$n \mid 1$， 则n=1，推出p的因子只有p和1，则p为素数.

② 再证当p为素数时，(p-1)! ≡ -1(mod p)成立：

(p-1)!=1*(p-1)*(2*3*…*(p-2))=(p-1)*((p-3)/2对互逆的数相乘)(mod p)=(p-1) (mod p)=-1 (mod p), 得证.

引例：6! (mod 7)=1*(2*4)*(3*5)*6 (mod 7)= 6 (mod 7)= -1 (mod 7);

欧拉函数与欧拉定理

欧拉函数ϕ(n)： 比n小且与n互素的正整数的个数.

例：ϕ(6)=2 [1,5]；　　ϕ(7)=6 [1,2,3,4,5,6]

欧拉函数ϕ(n)的计算： 如果p是素数，则ϕ(p)=p-1. 如果n=pq，且p和q是两个不同的素数，则ϕ(n)=(p-1)(q-1).

欧拉定理：如果 gcd(a,n)=1，则$a^{\phi(n)} \equiv 1 \pmod n$.

证明：设X={x | x为小于n的正整数且gcd(x,n)=1}， 则 |X|=ϕ(n);

对元素a且gcd(a,n)=1,有集合aX={ax mod n| x∈X}；

因为gcd(a, n)=gcd(x, n)=1, 可知gcd(ax, n)=1； 再看aX集合中任意两个元素

$ax_i \equiv ax_j \pmod n$是否成立；如成立，则推出$n|(x_i-x_j)$, 即$x_i=x_j$，矛盾⇒不成立

因此aX=X, 即两个集合相等;

将集合X和集合aX中所有元素相乘，得： $\prod_{x\in X} x = \prod_{y\in aX} y = \prod_{x\in X} ax \mod n$

再将以上等式两边都乘x mod n的逆，等式最左边为1，等式最右边为$a^{\phi(n)}$,

因此$a^{\phi(n)} \equiv 1 \pmod n$, 得证.

欧拉函数与欧拉定理

欧拉定理：如果 gcd(a,n)=1，则$a^{\phi(n)} \equiv 1 \pmod{n}$.

欧拉定理应用：求解 a 在模 n 下的逆元，即$a^{\phi(n)-1}\, a \equiv 1 \pmod{n}$

例: 求 3 在模 7 下的逆元

解:

gcd(3, 7) = 1

$3^{\phi(7)-1}\, 3 \equiv 1 \pmod{7}$

逆元是$3^{\phi(7)-1} = 3^{(7-1)-1} = 243$

存在唯一小于7的正整数 243 mod 7 = 5，5也是3 在模 7 下的逆元 5

练习: 求 3 在模 7 下的逆元

解: 逆元是3

欧拉定理推论

欧拉定理推论：如果 gcd(a,n)=1，则$a^x \equiv a^{x \pmod{\phi(n)}} \pmod{n}$.

证明：x=q*ϕ(n)+r, r= x mod ϕ(n),

则 $a^x = a^{q\phi(n)+r} = (a^{\phi(n)})^q a^r \equiv a^r \equiv a^{x \pmod{\phi(n)}} \pmod{n}$.

欧拉定理推论应用：计算2^{999} mod 21=?

因n=21=3*7，则ϕ(21)=2*6=12, 又因为gcd(2,21)=1,

则2^{999} mod 21=$2^{999 \bmod 12}$=2^3=8 (mod 21)

在RSA公钥密码算法正确性证明中也会用到上述推论.

Fermat's Little Theorem (费马小定理)

定理 3:如果 p 是一个素数且 a 是一个不被 p 整除的整数，则有

$a^{p-1} \equiv 1 \pmod{p}$

此外，对于每个整数 a，都有

$a^p \equiv a \pmod{p}$

Pierre de Fermat

(1601-1665)

费马小定理应用：计算整数的大次幂的模 p 的余数

例: 计算 7^{222} **mod** 11.

根据费马小定理，我们知道$7^{10} \equiv 1 \pmod{11}$, 所以 $(7^{10})^k \equiv 1 \pmod{11}$, 对于每个正整数 k. 因此,

$$7^{222} = 7^{22 \cdot 10 + 2} = (7^{10})^{22} 7^2 \equiv (1)^{22} \cdot 49 \equiv 5 \pmod{11}.$$

因此, 7^{222} **mod** 11 = 5.

Pseudoprimes(伪素数)1

根据费马小定理，对于素数 $n>2$，有

$$2^{n-1} \equiv 1 \pmod{n}.$$

但如果这个同余方程成立，n 不一定是素数.合成整数n 满足 $2^{n-1} \equiv 1 \pmod{n}$被称为该基数2下的伪素数.

例: 整数 341 是以2为基数的伪素数.

$341 = 11 \cdot 31$

$2^{340} \equiv 1 \pmod{341}$

我们可以将 2 替换为任意整数$b \geq 2$.

定义:设 b 是一个正整数。如果 n 是一个合数，并且 $b^{n-1} \equiv 1 \pmod{n}$，那么 n 被称为以 b为基数的伪素数

伪素数2

给定一个正整数 n，如果$2^{n-1} \equiv 1 \pmod{n}$:

- 如果 n 满足该同余方程，它可能是素数，也可能是以2为基数的伪素数
- 如果 n 不满足该同余方程，它是合数

使用额外的基数 b 进行类似的测试，可以提供更多关于 n 是否为素数的证据

与素数相比，在不超过正实数 x 的正整数中，基数 b 的伪素数相对较少

- 例如，在小于 10^{10}的正整数中，有 $455,052,512$ 个素数，但只有 $14,884$ 个基数 2 的伪素数.

卡米切尔数

定义: 一个合数 n，如果对所有满足 gcd(b,n)=1的正整数 b 都满足 $b^{n-1} \equiv 1 \pmod{n}$，则称 n 为卡米切尔数

例子: 整数561是一个卡米切尔数。证明如下:

- 561 是合数, 因为 561 = 3 · 11 · 13.
- 如果 gcd(b, 561) = 1, 那么 gcd(b, 3) = 1, 那么gcd(b, 11) = gcd(b, 17) =1.
- 根据费马小定理: $b^2 \equiv 1 \pmod{3}$, $b^{10} \equiv 1 \pmod{11}$, $b^{16} \equiv 1 \pmod{17}$.
- 则

$$b^{560} = (b^2)^{280} \equiv 1 \pmod{3},$$
$$b^{560} = (b^{10})^{56} \equiv 1 \pmod{11},$$
$$b^{560} = (b^{16})^{35} \equiv 1 \pmod{17}.$$

- 由此可知，对于所有满足 gcd (b,561)=1 的正整数 b，都有 $b^{560} \equiv 1 \pmod{561}$。因此，561 是一个卡米切尔数.

Primitive Roots (原根)

定义: 模素数 p 的原根是 $\mathbf{Z}_p$ 中的一个整数 r，使得 $\mathbf{Z}_p$ 中的每一个非零元素都是 r 的某个幂次（即 $r^e \bmod p$）

例 1: 对于素数 11，2 是一个原根，因为 $\mathbf{Z}_{11}$ 中的每个元素都可以表示为 2 的某个幂。计算如下：

$2^1 = 2,\ 2^2 = 4,\ 2^3 = 8,\ 2^4 = 5,\ 2^5 = 10,$

$2^6 = 9,\ 2^7 = 7,\ 2^8 = 3,\ 2^9 = 6,\ 2^{10} = 1.$

例 2: 对于质数 11，3 不是一个原根，因为并非所有 $\mathbf{Z}_{11}$ 中的元素都是 3 的某个幂。计算如下：

$3^1 = 3, 3^2 = 9, 3^3 = 5, 3^4 = 4, 3^5 = 1,$ 当计算3的更高幂次时，这个模式会重复，并不能生成所有 $\mathbf{Z}_{11}$中的元素.

重要事实: 对于每个素数 p，总存在一个原根.

Discrete Logarithms (离散对数)

假设 p 是一个素数，r 是模 p 的一个原根。如果 a 是一个位于 1 和 p–1 之间的整数，即 a 是 $\mathbf{Z}_p$ 的一个元素，那么存在一个唯一的指数 e使得$r^e = a$ 在 $\mathbf{Z}_p$中成立， 也就是 $r^e \bmod p = a$.

定义：设 p 是一个素数，r 是模 p 的一个原根，且 a 是位于 1 到 p–1 之间的整数。如果 $r^e \bmod p = a$ ，且 $1 \le e \le p-1$，我们称 e 是以 r 为底数 a 模 p的离散对数， 写作$\log_r a = e$ (其中素数 p 是默认的).

例: 试找出以2 为底3 和5 模11的离散对数

因为以2为底3模11的离散对数是 8，即$2^8 \bmod 11 = 3$. 我们写作 $\log_2 3 = 8$

因为以2为底5模11的离散对数是 4，即$2^4 \bmod 11 = 5$.我们写作 $\log_2 5 = 4$

同余式的应用

小节概要

Hashing Functions (哈希函数)

Pseudorandom Numbers (伪随机数)

Check Digits (数字校验位)

Hashing Functions (哈希函数)

定义: 哈希函数 h 将具有键 k 的记录分配到内存位置 $h(k)$.

- 一个常见的哈希函数是 $h(k)=k \bmod m$，其中 m 是内存位置的总数.
- 由于这个哈希函数是满射的，所有内存位置都是可能的.

例子: 设哈希函数为 $h(k)=k \bmod 111$。这个哈希函数将具有社会安全号码作为键的客户记录分配到内存位置，如下所示:

h(064212848) = 064212848 **mod** 111 = 14

h(037149212) = 037149212 **mod** 111 = 65

h(107405723) = 107405723 **mod** 111 = 14,但由于位置 14 已经被占用，因此记录被分配到下一个可用的位置，即 15.

哈希函数不是一一对应的，因为可能的键比内存位置要多得多。当多个记录被分配到同一位置时，我们说发生了冲突。这里，通过将记录分配到第一个空闲位置解决了冲突.

对于冲突解决，我们可以使用**线性探测函数**: $h(k,i) = (h(k) + i) \bmod m$,其中 i 从 0 到 $m-1$ 变化.

还有许多其他处理冲突的方法.

伪随机数1

随机选择的数字在许多应用中都是必需的，包括计算机模拟.

随机数并不是真正的随机，因为它们是通过系统化的方法生成的.

线性同余法是生成伪随机数的一种常用方法.

生成伪随机数需要四个整数：模数 m、乘数 a、增量 c、以及种子 x_0, 其中 $2 \le a < m$, $0 \le c < m$, $0 \le x_0 < m$.

我们通过递归定义的函数生成伪随机数序列$\{x_n\}$,其中 $0 \le x_n < m$ 对于所有n成立。

递归定义的函数为： $x_{n+1} = (ax_n + c) \bmod m.$

如果需要介于 0 和 1 之间的伪随机数，可以将生成的数字除以模数，即x_n/m.

伪随机数2

例: 找到由线性同余法生成的伪随机数序列，模数为 m=9，乘数为 a=7，增量为 c=4，种子为x_0 = 3.

解:通过依次使用同余关系计算序列的项

$$
\begin{aligned}
x_1 &= 7x_0 + 4 \bmod 9 = 7 \cdot 3 + 4 \bmod 9 = 25 \bmod 9 = 7,\\
x_2 &= 7x_1 + 4 \bmod 9 = 7 \cdot 7 + 4 \bmod 9 = 53 \bmod 9 = 8,\\
x_3 &= 7x_2 + 4 \bmod 9 = 7 \cdot 8 + 4 \bmod 9 = 60 \bmod 9 = 6,\\
x_4 &= 7x_3 + 4 \bmod 9 = 7 \cdot 6 + 4 \bmod 9 = 46 \bmod 9 = 1,\\
x_5 &= 7x_4 + 4 \bmod 9 = 7 \cdot 1 + 4 \bmod 9 = 11 \bmod 9 = 2,\\
x_6 &= 7x_5 + 4 \bmod 9 = 7 \cdot 2 + 4 \bmod 9 = 18 \bmod 9 = 0,\\
x_7 &= 7x_6 + 4 \bmod 9 = 7 \cdot 0 + 4 \bmod 9 = \ \ 4 \bmod 9 = 4,\\
x_8 &= 7x_7 + 4 \bmod 9 = 7 \cdot 4 + 4 \bmod 9 = 32 \bmod 9 = 5,\\
x_9 &= 7x_8 + 4 \bmod 9 = 7 \cdot 5 + 4 \bmod 9 = 39 \bmod 9 = 3.
\end{aligned}
$$

生成的序列是：3,7,8,6,1,2,0,4,5,3,7,8,6,1,2,0,4,5,3,... 它在生成 9 项后重复.

通常，计算机使用增量 c=0的线性同余生成器。这种生成器称为纯乘法生成器

使用模数 $2^{31}-1$和乘数 $7^5 = 16{,}807$的生成器在重复之前会生成$2^{31}-2$个数字

校验位: UPCs

常见的检测数字串中错误的方法是添加一个额外的数字在末尾，该数字通过一个函数进行评估。如果最后的数字不正确，则假定该字符串不正确.

例子:零售产品通过其通用产品代码（UPC）进行标识。这些代码通常有12位十进制数字，其中最后一位是校验位。校验位通过以下同余式确定:

$$3x_1 + x_2 + 3x_3 + x_4 + 3x_5 + x_6 + 3x_7 + x_8 + 3x_9 + x_{10} + 3x_{11} + x_{12} \equiv 0 \pmod{10}.$$

a. 假设UPC的前11位数字是 79357343104。那么校验位是多少？

b. 041331021641 是否是一个有效的UPC?

解:

a. $3\cdot 7 + 9 + 3\cdot 3 + 5 + 3\cdot 7 + 3 + 3\cdot 4 + 3 + 3\cdot 1 + 0 + 3\cdot 4 + x_{12} \equiv 0 \pmod{10}$
$21 + 9 + 9 + 5 + 21 + 3 + 12 + 3 + 3 + 0 + 12 + x_{12} \equiv 0 \pmod{10}$
$98 + x_{12} \equiv 0 \pmod{10}$
$x_{12} \equiv 2 \pmod{10}$ 所以校验位是2.

b. $3\cdot 0 + 4 + 3\cdot 1 + 3 + 3\cdot 3 + 1 + 3\cdot 0 + 2 + 3\cdot 1 + 6 + 3\cdot 4 + 1 \equiv 0 \pmod{10}$
$0 + 4 + 3 + 3 + 9 + 1 + 0 + 2 + 3 + 6 + 12 + 1 = 44 \equiv 4 \not\equiv 0 \pmod{10}$
所以, 041331021641 不是有效UPC.

校验位:ISBNs

图书通过国际标准图书编号（ISBN-10）来识别，这是一个10位数字的代码。前9位数字用来标识语言、出版社和图书。第十位数字是校验位，通过以下的同余关系确定

$$x_{10} \equiv \sum_{i=1}^{9} i x_i \pmod{11}.$$

ISBN-10号码的有效性可以通过以下等式来评估 $\sum_{i=0}^{10} i x_i \equiv 0 \pmod{11}$

a. 假设 ISBN-10 的前 9 位数字是 007288008。校验数字是多少?

b. 084930149X 是有效的 ISBN-10 吗? X is used for the digit 10.

解:

a. $X_{10} \equiv 1\cdot 0 + 2\cdot 0 + 3\cdot 7 + 4\cdot 2 + 5\cdot 8 + 6\cdot 8 + 7\cdot 0 + 8\cdot 0 + 9\cdot 8 \pmod{11}.$
$X_{10} \equiv 0 + 0 + 21 + 8 + 40 + 48 + 0 + 0 + 72 \pmod{11}.$
$X_{10} \equiv 189 \equiv 2 \pmod{11}$. 因此, $X_{10} = 2$.

b. $1\cdot 0 + 2\cdot 8 + 3\cdot 4 + 4\cdot 9 + 5\cdot 3 + 6\cdot 0 + 7\cdot 1 + 8\cdot 4 + 9\cdot 9 + 10\cdot 10 =$
$0 + 16 + 12 + 36 + 15 + 0 + 7 + 32 + 81 + 100 = 299 \equiv 2 \not\equiv 0 \pmod{11}$

因此, 084930149X 不是有效ISBN-10.

单一错误是指识别号码中的一个数字错误，而置换错误是指两个数字意外交换的位置。这两种错误都可以通过 ISBN-10 的检查数字进行检测.

Cryptography
密码学

小节概要6

Classical Cryptography 古典密码学

Cryptosystems 密码系统

Public Key Cryptography 公钥密码学

RSA Cryptosystem RSA 密码系统

Cryptographic Protocols 密码协议

Primitive Roots and Discrete Logarithms

原根和离散对数

凯撒密码[1]

尤利乌斯·凯撒通过将字母表中的每个字母正向移动三个字母来创建秘密信息（将最后三个字母移到前面）。例如，字母 B 被替换为 E，字母 X 被替换为 A。这种使信息变得秘密的过程是加密的一个例子。

加密过程如下:

- 将每个字母替换为一个整数，范围从 0 到 25，表示字母在字母表中位置的前一个整数.
- 加密函数为 f(p) = (p + 3) mod 26。它将整数 p 从集合 {0,1,2,…,25} 替换为集合 {0,1,2,…,25} 中的 f(p).
- 将每个整数 p 替换为字母表中位置为 p + 1 的字母.

例: 使用凯撒密码加密信息 “MEET YOU IN THE PARK”.

解: 12 4 4 19　　24 14 20　　8 13　　19 7 4　　15 0 17 10.

现在将这些数字 p 替换为 f(p) = (p + 3) mod 26.

15 7 7 22　　1 17 23　　11 16　　22 10 7　　18 3 20 13.

将这些数字转换回字母，得到加密后的信息 “PHHW BRX LQ WKH SDUN.”

凯撒密码2

要恢复原始消息，使用f的迎=逆函数$f^{-1}(p) = (p - 3) \bmod 26$。因此，编码消息中的每个字母都反向移动三个字母，前三个字母移到最后三个字母。从加密消息恢复原始消息的过程称为**解密**。

凯撒密码是一种移位密码。可以把每个字母移动 k 位，k=3只是其中的一种可能。于是，加密函数可以用：

$$f(p) = (p + k) \bmod 26$$

解密函数可以用：

$$f^{-1}(p) = (p - k) \bmod 26$$

整数 k 称为**密钥**.

移位密码1

例 1: 用密钥为 k = 11 的移位密码加密明文消息 “STOP GLOBAL WARMING”.

解: 将每个字母替换为$\mathbf{Z}_{26}$中对应的元素，得到数字串

18 19 14 15　6 11 14 1 0 11　22 0 17 12 8 13 6

对以上每个数应用移位函数 f(p) = (p + 11) mod 26，得到

3 4 25 0　17 22 25 12 11 22　7 11 2 23 19 24 17

将这些数字翻译成字母，得到密文

“DEZA RWZMLW HLCXTYR”

移位密码2

例 2: 解密使用密钥 k = 7 的移位密码加密的密文消息 “LEWLYPLUJL PZ H NYLHA ALHJOLY”.

解: 将每个字母翻译成 $\mathbf{Z}_{26}$ 中的元素.

11 4 22 11 24 15 11 20 9 1 15 25 7 13 24 11 7 0 0 11 7 9 14 11 24

将每个数字按 −k = −7 模 26 移位，得到

4 23 15 4 17 8 4 13 2 4 8 18 0 6 17 4 0 19 19 4 0 2 7 4 17

将这些数字翻译回字母，得到解密后的消息：

“EXPERIENCE IS A GREAT TEACHER”

仿射密码

移位密码是仿射密码的一种特例，它使用如下形式的加密函数：

$$f(p) = (ap + b) \bmod 26$$

其中 a 和 b 是整数，其选择需保证 f 是一个双射函数（当且仅当 gcd(a,26) = 1 时，该函数是双射）

例: 当使用函数 f(p) = (7p + 3) mod 26 进行加密时，字母 K 被替换为哪个字母？

解: 由于10代表K，f(10) = (7·10 + 3) mod 26 = 21，所对应的字母为V

为了解密由移位密码加密的信息，需要解方程 $c \equiv ap + b \pmod{26}$ 来求解 p

- 从两边减去 b，得到$c - b \equiv ap \pmod{26}$.
- 将两边乘以 a 在模 26 下的逆元，逆元的存在条件是gcd(*a*,26) = 1.
- $\bar{a}(c - b) \equiv \bar{a}ap \pmod{26}$, 简化为$\bar{a}(c - b) \equiv p \pmod{26}$.
- $p \equiv \bar{a}(c - b) \pmod{26}$ 被用来确定 $\mathbf{Z}_{26}$ 中的 p

移位密码的密码分析

从密文中恢复明文的过程，而无需知道加密方法和密钥，被称为**密码分析**或破解密码。

对移位密码生成的密文进行密码分析的主要工具是密文中字母的相对频率。英文文本中最常见的九个字母及其出现频率是：E 13%，T 9%，A 8%，O 8%，I 7%，N 7%，S 7%，H 6%，R 6%

分析密文的步骤:

- 找出密文中各字母的频率.
- 假设最常见的字母是通过加密 E 生成的.
- 如果从 E 到最常见字母的位移值为 k，则将密文移 –k 看看是否有意义.
- 如果不成立，尝试假设是 T 并继续分析.

例: 我们截获了消息“ZNK KGXRE HOXJ MKZY ZNK CUXS”，并且知道它是通过移位密码生成的。我们尝试进行密码分析.

解: 密文中最常见的字母是 K。因此，可能字母被移位了 6 个位置，因为这样会将 E 映射到 K。将密文的字母移 -6 个位置，得到“THE EARLY BIRD GETS THE WORM”

Block Cipher (分组密码1)

将字母表中的每个字母替换为另一个字母的密码被称为字符密码或**单码密码**.

这类密码容易受到基于字母频率的密码分析攻击。为了避免这个问题，**分组密码**应运而生，它通过用一组字母替换另一组字母来提高安全性.

一种简单的分组密码称为**换位密码**。密钥是集合 {1,2,…,m} 的一个置换 σ，其中 m 是一个整数，它是从 {1,2,…,m} 到自身的单射函数.

加密消息时，将字母分成大小为m的分组（添加额外的字母以填满最后一个分组）。我们通过加密$p_1,p_2,\ldots,p_m$ 为 $c_1,c_2,\ldots,c_m = p_{\sigma(1)},p_{\sigma(2)},\ldots,p_{\sigma(m)}$

要解密$c_1,c_2,\ldots,c_m$则需要使用逆置换 σ^{-1} 进行还原

分组密码2

例: 使用基于集合 {1,2,3,4} 的置换 σ 来进行加密和解密操作，其中 σ(1) = 3, σ(2) = 1, σ(3) = 4, σ(4) = 2,

a. 加密明文 ″PIRATE ATTACK“

b. 解密密文 ″SWUE TRAEOEHS"，该密文使用了相同的置换 σ 进行加密

解:

a. 将明文划分为四个字母一组：PIRA TEAT TACK，应用置换 σ，即把第一个字母移到第三位，把第二个字母移到第一位，把第三个字母移到第四位，再把第四个字母移到第二位。 得到IAPR ETTA AKTC

分组密码2

例: 使用基于集合 {1,2,3,4} 的置换 σ 来进行加密和解密操作，其中 σ(1) = 3, σ(2) = 1, σ(3) = 4, σ(4) = 2,

a. 加密明文 ″PIRATE ATTACK“

b. 解密密文 ″SWUE TRAEOEHS"，该密文使用了相同的置换 σ 进行加密

解:

b. σ^{-1}: $\sigma^{-1}(1) = 2, \sigma^{-1}(2) = 4, \sigma^{-1}(3) = 1, \sigma^{-1}(4) = 3.$

将密文分成四个字母一组：SWUE TRAE OEHS，应用置换的逆 σ^{-1}.

解密后的结果是：USEW ATER HOSE.

将解密后的结果拆分为单词，得到：USE WATER HOSE

密码系统[1]

定义: 一个密码系统是一个五元组 (P, C, K, E, D)，其中

- P 是明文字符串的集合,
- C 是密文字符串的集合,
- K 是密钥空间（所有可能密钥的集合），
- E 是加密函数的集合
- D 是解密函数的集合.

与密钥 k 对应的加密函数属于 E，记作 E_k；用来解密由 E_k 加密的密文的解密函数属于 D，记作 D_k。因此:

对于所有明文字符串 p, 有$D_k(E_k(p)) = p$,

密码系统2

例: 将移位密码系列描述为一个密码系统

解: 假设消息是由 $\mathbf{Z}_{26}$ 中的元素组成的字符串

- P 是由 $\mathbf{Z}_{26}$ 中元素组成的字符串集合,
- C 是由 $\mathbf{Z}_{26}$ 中元素组成的字符串集合,
- K = $\mathbf{Z}_{26}$,
- E 包含形式为 $E_k(p)=(p+k) \bmod 26$ 的函数,
- D 与 E 相同，其中 $D_k(p)=(p-k) \bmod 26$

公钥密码学

所有古典密码，包括移位密码和仿射密码，都是**私钥密码系统**的实例。一旦知道加密密钥，可以快速找到解密密钥

所有希望使用私钥密码系统进行通信的各方必须共享密钥，并且保持其秘密

为了避免每对希望安全通信的双方都需要共享密钥，在1970年代密码学家引入了**公钥密码系统**。知道如何加密消息并不能帮助解密消息。因此，每个人都可以拥有公开的加密密钥，唯一需要保密的是解密密钥，而且只有消息的预期接收人能解密

最常用的公钥密码系统RSA

1976年，麻省理工学院的三位研究人员引入了一种现在被称为RSA系统的公钥密码系统

Ronald Rivest
(Born 1948)

Adi Shamir
(Born 1952)

Leonard
Adelman
(Born 1945)

现在已知这一方法早在之前就由克利福德·科克斯（Clifford Cocks）秘密为英国政府工作时发现

公钥加密密钥为(n, e)，其中n = pq（模数）是两个大素数p和q的乘积，这两个素数大约有300位数，而e是一个与(p−1)(q−1)互素的指数。可以使用计算机和随机性素数测试快速找到这两个大素数。但是，n = pq的长度大约有600位数，在合理的时间内无法分解因子

RSA加密

要使用RSA加密系统加密消息，使用密钥(n, e):

i. 将明文消息M转换为表示字母的两位整数序列。用00表示A，01表示B，以此类推.

ii. 将这些两位整数串联成数字字符串.

iii. 将这个字符串分成大小相等的组，每组的长度为2N，其中2N是一个大偶数使得2N位数字的整数（例如2525...25 ）不超过n

iv. 明文消息M现在是一系列整数$m_1, m_2, \ldots, m_k$

v. 使用函数$C = M^e \bmod n$加密每个组（一个整数）

例子: 使用密钥(2537, 13)加密消息"STOP"

- 2537 = 43· 59,
- p = 43 and q = 59是素数并且 gcd(e,(p−1)(q −1)) = gcd(13, 42· 58) = 1.

解: 将"STOP"中的字母转换为数字对应值18 19 14 15.

- 按4位数字一组将这些数字分组（因为2525 < 2537 < 252525），得到1819 1415
- 使用函数 $C = M^{13} \bmod 2537$加密每个组
- 因为 $1819^{13} \bmod 2537 = 2081$ 并且 $1415^{13} \bmod 2537 = 2182$, 所以加密后的消息是2081 2182

RSA解密

要解密RSA密文消息，需要解密密钥d，这是e模(p−1)(q−1)的逆。因为gcd(e, (p−1)(q−1)) = gcd(13, 42 * 58) = 1，所以逆是存在的

有了解密密钥d，我们可以通过计算$M = C^d \bmod p \cdot q$来解密每个组

例子: 收到的消息是0981 0461。如果它是用之前示例中的RSA密码加密的，那么解密后的消息是什么？

解: 消息是用n = 43 * 59和指数13加密的。e = 13的模42 * 58 = 2436的逆是d = 937

- 要解密一个组，使用 $M = C^{937} \bmod 2537$
- 因为 $0981^{937} \bmod 2537 = 0704$ 并且 $0461^{937} \bmod 2537 = 1115$所以解密后的消息是0704 1115。将这些数字转换回英文字母，得到的消息是HELP

RSA作为公钥系统的工作原理是，因为找到d的唯一已知方法是基于n的素因数分解。目前没有已知的可行方法来分解大数为素因数

RSA 密码系统

Two n bit primes p and q. Set $N = pq$.
Pick random e, $1 < e < n$ with $\gcd(e, \phi(N)) = 1$

public key N, e
private key p, q
encode(m), $m^e \pmod N$ m must satisfy $\gcd(m, N) = 1$
decode(c) $c^d \pmod N$ where $d = e^{-1} \pmod{\phi(N)}$
Note $\phi(N) = (p-1)(q-1)$

Correctness of RSA

$$\begin{aligned} D(E(m)) &= (m^e \pmod N)^d \pmod N \\ &= m^{ed} \pmod N \\ &= m^{ed \pmod{\phi(N)}} \pmod N \text{ by cor} \\ &= m \pmod N \text{ since } d = e^{-1} \pmod{\phi(N)} \\ &= m. \end{aligned}$$

$\gcd(m, N) = 1$ is required so we can use Euler's Theorem
$0 < m < N$ so $m \pmod N = m$.

RSA 加解密简单练习

练习：公钥: (n,e)=(35, 5),其中n=pq=5*7, gcd(e, ϕ(35))=gcd(5, 24)=1;

求解（1）解密密钥d；（2）取明文m=2，则加密的密文 c是多少；（3）写出解密过程

解：

（1）$d=e^{-1} \bmod \phi(35)=5$ [用欧几里得算法回代可得]

（2）取明文m=2, 这里gcd(m,n)=gcd(2,35)=1，则加密的密文 $c=m^e \bmod n=2^5 \bmod 35=32$

（3）解密：$c^d \bmod n=32^5 \bmod 35=32^2*32^2*32 \bmod 35=9*9*32 \bmod 35=11*32 \bmod 35= 352 \bmod 35=2$，和明文2一致

加密协议：密钥交换

加密协议 是由两个或更多方为了达到一个特定的安全目标而进行的消息交换

密钥交换 是一种协议，通过该协议，两方可以在没有任何过去共享的秘密信息的情况下，通过不安全的通信信道交换密钥。下面通过示例描述了迪菲-赫尔曼密钥协议

i. 假设Alice和Bob希望共享一个共同的密钥.
ii. Alice和Bob同意使用一个素数p和p的一个原根a.
iii. Alice选择一个秘密整数k1，并将 a^{k1} **mod** p发送给Bob.
iv. Bob选择一个秘密整数k2，并将 a^{k2} **mod** p发送给Alice.
v. Alice 计算 $(a^{k2})^{k1}$ **mod** p.
vi. Bob 计算 $(a^{k1})^{k2}$ **mod** p.

在协议结束时，Alice和Bob都有他们的共享密钥

$(a^{k2})^{k1}$ **mod** $p = (a^{k1})^{k2}$ **mod** p.

要从公开信息中找出秘密信息，攻击者需要从 a^{k1} **mod** p和 a^{k2} **mod** p分别找出k1和k2。这是离散对数问题的一个实例，当p和a足够大时，计算上被认为是不可行的.

迪菲-赫尔曼密钥交换简单例子

步骤1：选择公共参数

- Alice和Bob选择一个大质数p和一个原根g (模p), 这里设p=23,g=5;

步骤2：选择各自私钥

- Alice和Bob各自秘密选择一个大的随机整数作为私钥. 假设Alice选择的私钥为a，Bob选择的私钥为b，这里设a=6, b=15

步骤3：各自计算公钥

- Alice计算她的公钥A，公式为：$A = g^a \bmod p$, 这里$A=5^6 \bmod 23=8$
- Bob计算他的公钥B，公式为：$B = g^b \bmod p$,这里$B=5^{15} \bmod 23=19$
- Alice和Bob将他们的公钥发送给对方。

步骤4：各自计算共享密钥

- Alice接收到Bob的公钥B后，计算共享密钥，公式为： $B^a \bmod p$, 这里公式为 $19^6 \bmod 23=2$
- Bob接收到Alice的公钥A后，计算共享密钥，公式为： $A^b \bmod p$, 这里公式为$8^{15} \bmod 23=2$
- 由于模运算的性质，Alice和Bob计算出的共享密钥是相同的

加密协议：数字签名[1]

添加数字签名到消息中是一种确保消息的接收者知道消息来自那个该来自的人

假设Alice的RSA公钥是(n, e)，她的私钥是d。Alice使用加密函数 $E_{(n,e)}(x) = x^e \bmod n$对明文消息x进行加密。她使用解密函数 $D_{(n,e)}(y) = y^d \bmod n$对密文消息y进行解密.

Alice想要发送一条消息M，以便每个收到消息的人都知道消息确实来自于她

1. 她将消息转换为数字等效值，并将其拆分成组，就像在RSA加密中一样
2. 然后，她对这些组应用她的解密函数 $D_{(n,e)}$，并将结果发送给所有预期的接收者.
3. 接收者使用Alice的加密函数进行处理，结果是原始的明文，因为 $E_{(n,e)}(D_{(n,e)}(x)) = x$.

因此，每个收到消息的人都可以确信消息确实来自于Alice.

加密协议：数字签名2

例子: 假设Alice的RSA加密系统与之前的例子相同，密钥为(2537, 13)， 2537 = 43· 59，p = 43 和 q = 59 是质数，且 gcd(e,(p−1)(q −1)) = gcd(13, 42· 58) = 1

她的解密密钥是 d = 937

她想要发送消息“MEET AT NOON”给她的朋友们，以确保他们能确认消息是来自她

解: 爱丽丝将消息翻译成数字组 1204 0419 0019 1314 1413

1. 然后，她对每个组应用她的解密变换 $D_{(2537,13)}(x) = x^{937}$ **mod** 2537
2. 她发现（使用她的笔记本电脑、二进制模幂算法）1204^{937} **mod** 2537 = 817, 419^{937} **mod** 2537 = 555 , 19^{937} **mod** 2537 = 1310, 1314^{937} **mod** 2537 = 2173, and 1413^{937} **mod** 2537 = 1026.
3. 她发送0817 0555 1310 2173 1026

当她的朋友之一收到消息时，他们对每个组应用爱丽丝的加密变换$E_{(2537,13)}$.然后他们获得原始消息，并将其翻译回英语字母

数字签名简单例子

我们还可以进一步发送签名的秘密消息（即，不仅用了数字签名还用对方公钥进行加密）

设Alice和Bob利用RSA公钥密码体系进行通信，Alice的公钥: $N_A=35$, $e_A=5$；Bob的公钥$N_B=33$,$e_B=3$，可得Alice的私钥为$(N_A,d_A)=(35,5)$，Bob的私钥为$(N_B,d_B)=(33,7)$

Alice向Bob发送的明文4并加了其签名的密文为：

$E_B(D_A(4))= E_B(4^5 \bmod 35)= E_B(9)=9^3 \bmod 33=729 \bmod 33=3$;

Bob的解密过程为：

$E_A(D_B(3))= E_A(3^7 \bmod 33)= E_A(9)=9^5 \bmod 35=81*81*9 \bmod 35=144 \bmod 35=4$

Homomorphic Encryption (同态加密)

全同态加密(Fully Homomorphic Encryption)：在计算过程中，对密文不解密，能直接在密文上进行各种运算，直到最终需要计算结果时，再进行解密得到最终解，且该最终解和明文上经过这些运算后得到的解相同

简单例子：$ENC((m_1+m_2)*m_3)= (ENC(m_1)+ENC(m_2))*ENC(m_3)$

部分同态加密(Partially Homomorphic)：只对某个运算具有同态特性，譬如只允许在密文上进行加法，或只进行乘法操作

加法同态(Additively Homomorphic)：$ENC(m_1+m_2)=(ENC(m_1)+ENC(m_2)$

乘法同态(Multiplicatively Homomorphic)：$ENC(m_1*m_2)= (ENC(m_1)*ENC(m_2)$

RSA密码系统是乘法同态，而不是加法同态：

$E_{(n,e)}(m_1)*E_{(n,e)}(m_2)=m_1^e \bmod n * m_2^e \bmod n=(m_1*m_2)^e \bmod n=E_{(n,e)}(m_1*m_2)$

即，明文乘积的密文等于各自密文的乘积

Zero Knowledge Proof (零知识证明)

零知识证明: 是一方（证明者Prover）向另一方（验证者Verifier）证明某命题的方法，特点是过程中除“该命题为真”之事外，不泄露任何信息。因此，可理解成“零泄密证明”

零知识证明有三个关键属性：

1.完备性（Completeness）：若命题为真，则诚实（意即依协议行事）的证明者能说服诚实验证者

2.可靠性（Soundness）：若命题为假，则作弊证明者仅有极小机会能说服诚实验证者该事为真

3.零知识（Zero-Knowledge）：验证者不能从证明中获取任何超出该命题为真的信息

证明者向验证者证明其知道离散对数，但验证者不能知道具体值(Schnorr零知识证明协议)：

一个质数p和模p的原根g; 证明者有个小于p的私钥x, 其公钥$y=g^x \bmod p$；接下来每轮的操作如下：

证明者随机选取一个小于p的整数r，并计算$a= g^r \bmod p$ 发给验证者；

验证者发送一个随机选取的比特e(0或1)给证明者；

证明者计算并发送s=(r+ex) mod (p-1)给验证者；

验证者验证$g^s=a*y^e \bmod p$是否成立，如成立则验证者相信证明者知道离散对数x.

重复以上每轮操作n次

完备性：如果证明者确实知道x，则每次验证者的验证均成立；

可靠性：如证明者假装知道x，当e=0时，验证成立；当e=1时，因其不能求出x，验证不成立，因此作弊的证明者仅能以1/2的概率通过一轮验证，但重复足够多轮，成功作弊的概率可降到任意小.

但若e值不是随机，则证明者可成功作弊: e=0, 如常继续选r，计算$a= g^r \bmod p$ 发给验证者, 验证者的验证总是成立；e=1，则随机取一个 r'，计算 $a' = g^{r'} \cdot (g^x)^{-1} \bmod p$，然后发送$r'$和 a' 值予证明者(伪装成s和a); 此时验证者计算的值为 $g^{r'} \bmod p$，等于 $a' \cdot y$，验证成立.

零知识：公钥y不会泄露x的值；s即(ex+r) mod (p-1)的值，可视为 x mod (p-1)的加密，若r确为随机，在0至(p-2)间均匀分布，则(ex+r) mod (p-1)也同样均匀分布，不会泄露任何关于x的信息.

如何处理大整数

如果一个整数超过64位，如何进行算术运算？

利用中国剩余定理（余数系统: Residue Number System），把大整数表示成一个tuple

如何处理大整数

定义一个余数系统如下

对于任意整数x, $x=(x_1|x_2|...|x_k)RNS(P_1|P_2|...|P_k)$

其中 $x_i = x \bmod P_i$ 并且 $\forall$ i,j P_i 与 P_j互质

例子: 84 = (0|4|0)RNS(7|5|3)

1 = (1|1|1)RNS(7|5|3)

2 = (2|2|2)RNS(7|5|3)

3 = (3|3|0)RNS(7|5|3)

如何处理大整数

加法： $x+y = ((x_1+y_1)_{p_1} | (x_2+y_2)_{p_2} | \ldots | (x_k+y_k)_{p_k})RNS(P_1|P_2|\ldots|P_k)$

where $x_i = (x)_{p_i}$ and $y_i = (y)_{p_i}$

乘法： $x*y = ((x_1*y_1)_{p_1} | (x_2*y_2)_{p_2} | \ldots | (x_k*y_k)_{p_k})RNS(P_1|P_2|\ldots|P_k)$

where $x_i = (x)_{p_i}$ and $y_i = (y)_{p_i}$

例子: 84 = (0|4|0)RNS(7|5|3)

3 = (3|3|0)RNS(7|5|3)

3×84 = 252 = (0|2|0)RNS(7|5|3)

3×84 = ((0×3) mod 7|(4×3) mod 5|(0×0) mod 3)RNS(7|5|3)

= (0|2|0)RNS(7|5|3)