



数论算法

Number-Theoretic Algorithms

主讲人 徐云

Fall 2018, USTC



数论算法 (Ch31)

1 初等数论记号 (31.1)

2 最大公约数 (31.2)

3 模运算 (31.3)

4 求解线性模方程 (31.4)

5 中国余数定理 (31.5)

6 RSA公钥密码系统 (31.7)

7 素数判定 (31.8)



初等数论记号 (31.1)

- 算术计算中的输入规模和成本
- 除定理
- 公约数和最大公约数
- 互素数和最大公约数

算术计算中的输入规模和成本

■ Def. 1

一个输入整数 a_1, a_2, \dots, a_k 的算法是多项式时间算法，如果其运行时间是以 $\log a_1, \log a_2, \dots, \log a_k$ 为多项式时间的，即以输入参数二进制编码长度的多项式时间的。

- 如，两个 β 位的整数乘法，普通乘法使用 $\Theta(\beta^2)$ ，改进的算法使用 $\Theta(\beta^{\log 3})$ ，甚至 $\Theta(\beta \log \beta \log \log \beta)$

除定理

- 概念：可除性， 素数， 合数

- Th31.1

对任何整数 a 和正整数 n , 存在唯一整数 q 和 r , 使得
 $a=qn+r$, 这里 $q=\lfloor a/n \rfloor$, $0 \leq r < n$ 。

注: q 为商, r 为余数

- Def. 2

模 n 的等价类: $[a]_n = \{ a+kn \mid k \in \mathbb{Z} \}$

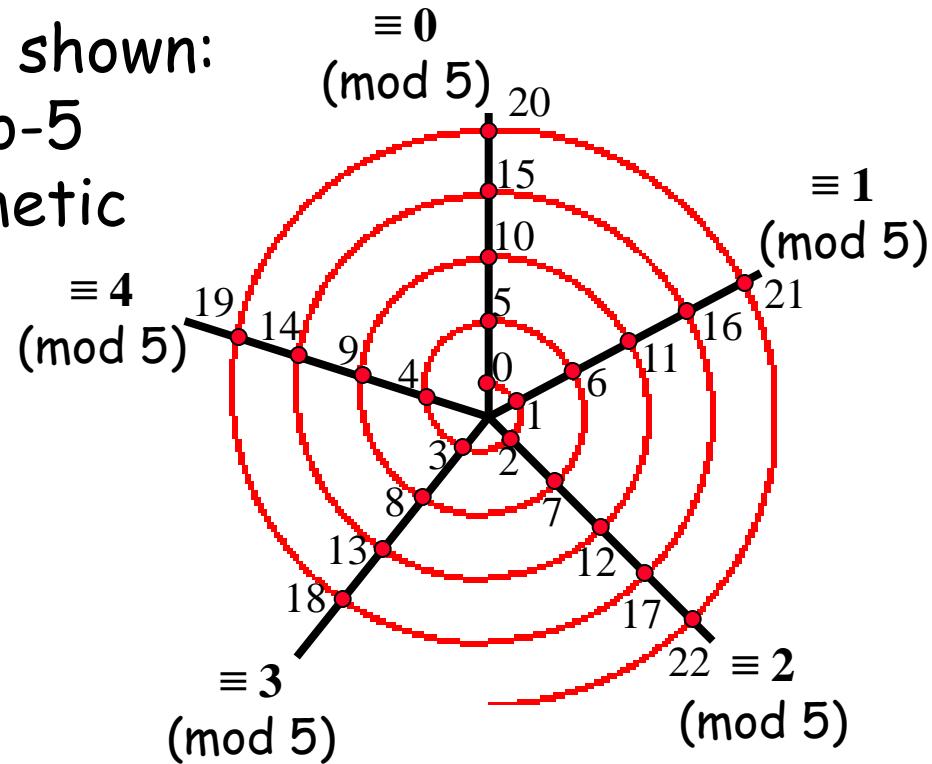
- 性质:

如果 $a \in [b]_n$, 则 $a \equiv b \pmod{n}$

除定理

■ Spiral Visualization of mod:

Example shown:
modulo-5
arithmetic



公约数和最大公约数

- 概念：约数，公约数，最大公约数

- Th31.2

a, b 为不全为零的两个整数，则最大公约数 $\gcd(a, b)$ 是 $\{ax+by \mid x, y \in \mathbb{Z}\}$ 中最小的正整数。

-系1：对所有整数 a 和 b ，如果 $d|a, d|b$ ，则 $d|\gcd(a, b)$

-系2：对所有整数 a 和 b ，和非负整数 n ，有

$$\gcd(an, bn) = n\gcd(a, b)$$

-系3：对所有正整数 n 、 a 和 b ，如果 $n|ab$ 且 $\gcd(a, n)=1$ ，则 $n|b$

互素数和素数唯一性分解定理

■ Def. 3 :

如果 $\gcd(a, b) = 1$, 称整数 a 和 b 为互素数

■ Th31.6

$\forall a, b, p \in \mathbb{Z}$, 如果 $\gcd(a, p) = 1$ 和 $\gcd(b, p) = 1$, 则
 $\gcd(ab, p) = 1$

■ Th31.8(唯一分解定理)

一个合数 a 能被唯一写成形式 $a = p_1^{e_1} p_2^{e_2} \dots p_r^{e_r}$

这里 p_i 是素数, $p_1 < p_2 < \dots < p_r$, e_i 是正整数

如, $6000 = 2^4 \times 3 \times 5^3$



数论算法 (Ch31)

1 初等数论记号 (31.1)

2 最大公约数 (31.2)

3 模运算 (31.3)

4 求解线性模方程 (31.4)

5 中国余数定理 (31.5)

6 RSA公钥密码系统 (31.7)

7 素数判定 (31.8)



最大公约数 (31.2)

- 一种直观的解GCD
- Euclid's Algriothm
- Euclid's Algriothm的运行时间
- 扩展的Euclid's Algriothm

一种直观解GCD

$$a = p_1^{e_1} p_2^{e_2} \dots p_r^{e_r} \text{ 和 } b = p_1^{f_1} p_2^{f_2} \dots p_r^{f_r}$$
$$\Rightarrow \gcd(a, b) = p_1^{\min(e_1, f_1)} p_2^{\min(e_2, f_2)} \dots p_r^{\min(e_r, f_r)}$$

注：这种解法需要整数的素因子分解，而素因子分解是一个
很难的问题(**NP**难题)

Euclid's Algorithm

- Th.31.9(GCD Recursion)

对任何非负整数a和正整数b, 有 $\text{gcd}(a, b) = \text{gcd}(b, a \bmod b)$

- Euclid's Algorithm

```
Euclid(a, b)
{ if b=0 then
    return a;
else
    return Euclid(b, a mod b);
}
```

- 例: $\text{gcd}(30, 21)$

Euclid's Algorithm的运行时间

■ Th.31.11(拉姆定理)

对整数 $k \geq 1$, 如果 $a > b \geq 1$ 且 $b < F_{k+1}$,

则 $\text{Euclid}(a, b)$ 递归调用的次数小于 k 。

注:

- Euclid's Alg. 递归调用的次数为 $O(\log b)$

- 算法应用到二个 β 位整数上,

算法耗费 $O(\beta)$ 算术运算和

$O(\beta^3)$ 位运算



Gabriel Lamé
1795-1870

扩展的Euclid's Algorithm

- 问题

Find x, y s.t. $\gcd(a, b) = ax + by$

- 算法

ExtendedEuclid(a, b)

{ if $b=0$ then

 return ($a, 1, 0$);

$(d', x', y') \leftarrow \text{ExtendedEuclid}(b, a \bmod b);$

$(d, x, y) \leftarrow (d', y', x' - \lfloor a/b \rfloor y');$

 return (d, x, y);

}

- 例：把 $3 = \gcd(99, 78)$ 表示成 99 和 78 的线性组合



数论算法 (Ch31)

1 初等数论记号 (31.1)

2 最大公约数 (31.2)

3 模运算 (31.3)

4 求解线性模方程 (31.4)

5 中国余数定理 (31.5)

6 RSA公钥密码系统 (31.7)

7 素数判定 (31.8)



模运算 (31.3)

- 模n的加法群和乘法群
- Euclid's phi Function
- Lagrange's Theorem

模n的加法群和乘法群

■ Def. 1:

设 $Z_n = \{ [a]_n \mid 0 \leq a \leq n-1 \}$, 定义加法运算 $+_n$:

$$[a]_n +_n [b]_n = [a+b]_n,$$

则 $(Z_n, +_n)$ 是有限Abelian群

如, $(Z_6, +_n)$, $Z_6 = \{[0]_6, [1]_6, [2]_6, [3]_6, [4]_6, [5]_6\}$

■ Def. 2:

设 $Z_n^* = \{ [a]_n \in Z_n \mid \gcd(a, n) = 1 \}$, 定义乘法运算 \times_n :

$$[a]_n \times_n [b]_n = [a \times b]_n,$$

则 (Z_n^*, \times_n) 是有限Abelian群

如, (Z_{15}^*, \times_n) , $Z_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$

Euclid's phi Function

- Theorem(Euclid's phi Func.)

令 $\varphi(n)$ 为 Z_n^* 的size, 则有 $\varphi(n) = n \prod_{p|n} \left(1 - \frac{1}{p}\right)$

- 例:

$$Z_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$$

$$|Z_{15}^*| = 15 \left(1 - \frac{1}{3}\right) \left(1 - \frac{1}{5}\right) = 8$$

Lagrange's Theorem

- Th. 31.15 (Lagrange's Theorem)

如果 H 是有限群 G 的子群，那么 $|H|$ 整除 $|G|$

系: $|H| \leq |G|/2$



数论算法 (Ch31)

1 初等数论记号 (31.1)

2 最大公约数 (31.2)

3 模运算 (31.3)

4 求解线性模方程 (31.4)

5 中国余数定理 (31.5)

6 RSA公钥密码系统 (31.7)

7 素数判定 (31.8)



求解线性模方程 (31.4)

- 问题描述
- $\langle a \rangle$ 群表示和构造定理
- 求解方法

问题描述

求 x 满足下面模方程：

$$ax \equiv b \pmod{n}$$

这里 $a > 0, n > 0$

$\langle a \rangle$ 群表示和构造定理

■ Def.:

令 $\langle a \rangle$ 是 \mathbb{Z}_n 上由 a 生成的子群, $\langle a \rangle = \{ ax \bmod n \mid x \geq 0, x \in \mathbb{Z} \}$

■ 性质:

方程 $ax \equiv b \pmod{n}$ 有解 $\Leftrightarrow b \in \langle a \rangle$

■ Th31.20(构造定理)

对正整数 a 和 n , $d = \gcd(a, n)$, 则有

$$\langle a \rangle = \langle d \rangle = \{ 0, d, 2d, \dots, (n/d - 1)d \}$$

为 \mathbb{Z}_n 上子群, 且 $|\langle a \rangle| = n/d$

系13.21: $ax \equiv b \pmod{n}$ 有解 $\Leftrightarrow \gcd(a, n) | b$

<a>群表示和构造定理

- 例：判断 $4x \equiv 2 \pmod{6}$ 和 $4x \equiv 3 \pmod{6}$ 有无解

$\because \gcd(4, 6) | 2 \quad \therefore 4x \equiv 2 \pmod{6}$ 有解

$\because \gcd(4, 6)$ 不整除 3 $\quad \therefore 4x \equiv 3 \pmod{6}$ 无解

注：

$$i: 0 \ 1 \ 2 \ 3 \ 4 \ 5$$

$$\langle 4 \rangle: 4i \bmod 6 = 0 \ 4 \ 2 \ 0 \ 4 \ 2$$

可以看出：第一个方程有二个解 2, 5；
第二个方程无解

求解方法

■ 先求特解

- Th31.23 设 $d = \gcd(a, n)$ 且 $d = ax' + ny'$, 对某个整数 x' 和 y' 。

如果 $d|b$, 则有特解 $x_0 = x'(b/d) \bmod n$

证:

$$ax_0 \equiv ax'(b/d) \pmod{n}$$

$$\equiv d(b/d) \pmod{n} // \because ax' + ny' = d \therefore ax' \equiv d \pmod{n}$$

$$\equiv b \pmod{n}$$

■ 求全部解

- Th31.24 设 x_0 为 $ax \equiv b \pmod{n}$ 的一个解, 则有 d 个不同解:

$$x_i \equiv x_0 + i(n/d) \quad i=0, 1, \dots, d-1$$

- 算法: `ModularLinearEquationSolver(a, b, n)`

求解方法

- 示例: $14x \equiv 30 \pmod{100}$

解:

① 调用 ExtendedEuclid(14, 100) $\Rightarrow (d, x', y') = (2, -7, 1)$

② $\because 2|30, \therefore$ 特解 $x_0 = -7 \times 30/2 \pmod{100} = 95$

③ 调用求全部解算法: 二个解 95, 45

$$x_0 = 95 + 0 \times 100/2 = 95$$

$$x_1 = 95 + 1 \times 100/2 = 145 \equiv 45 \pmod{100}$$



数论算法 (Ch31)

1 初等数论记号 (31.1)

2 最大公约数 (31.2)

3 模运算 (31.3)

4 求解线性模方程 (31.4)

5 中国余数定理 (31.5)

6 RSA公钥密码系统 (31.7)

7 素数判定 (31.8)



中国余数定理 (31.5)

- 一个中国古代问题
- a 模 n 的逆存在唯一性定理
- 中国余数定理

一个中国古代问题

■ 问题

某物不知其数，
三分之余二，
五分之余三，
七分之余二，
此物几何？

■ 该问题可以表示成线性同余方程组为

$$\begin{cases} x \equiv 2 \pmod{3} \\ x \equiv 3 \pmod{5} \\ x \equiv 2 \pmod{7} \end{cases}$$

a 模 n 的逆存在唯一性定理

■ Def. :

若 x 使得 $xa \equiv 1 \pmod{n}$, 称 x 为 a 模 n 的逆

■ Theorem:

若 a 和 n 互素, $n > 1$, 则唯一存在 a 模 n 的逆。

注:

- 以下记 a 模 n 的逆为 a^{-1} , 使 $a^{-1}a \equiv 1 \pmod{n}$
- 本定理实际上给出了求 a 模 n 的逆的方法

■ 例: (1)求3模7的逆; (2)求 $3x \equiv 4 \pmod{7}$

中国余数定理

■ CRT(China Remainder Theorem)

令 n_1, n_2, \dots, n_k 为两两互素的正整数，则同余方程组

$$x \equiv a_1 \pmod{n_1}$$

$$x \equiv a_2 \pmod{n_2}$$

...

$$x \equiv a_k \pmod{n_k}$$

有唯一的模 $n = n_1 n_2 \dots n_k$ 解 x 。

注：本定理实际上给出了求解方法

■ 例(一个中国古代问题)



数论算法 (原书Ch31)

1 初等数论记号 (31.1)

2 最大公约数 (31.2)

3 模运算 (31.3)

4 求解线性模方程 (31.4)

5 中国余数定理 (31.5)

6 RSA公钥密码系统 (31.7)

7 素数判定 (31.8)



RSA公钥密码系统 (31.7)

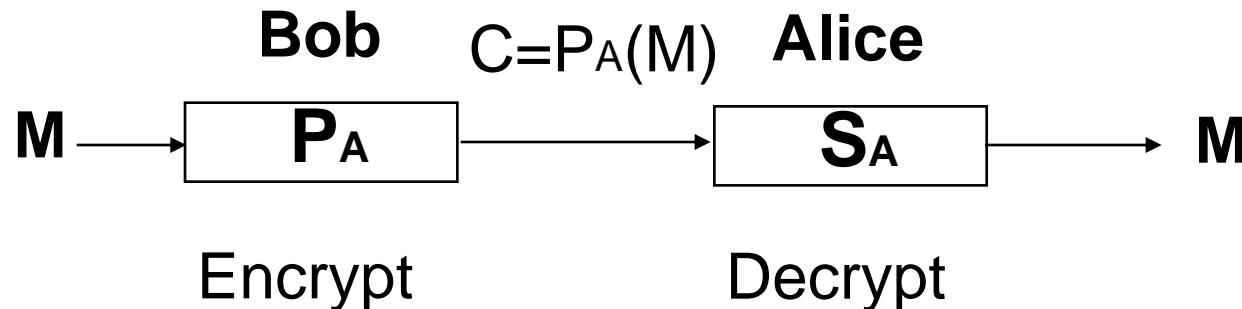
- 公钥密码系统
- RSA 系统
- RSA 的正确性

公钥密码系统

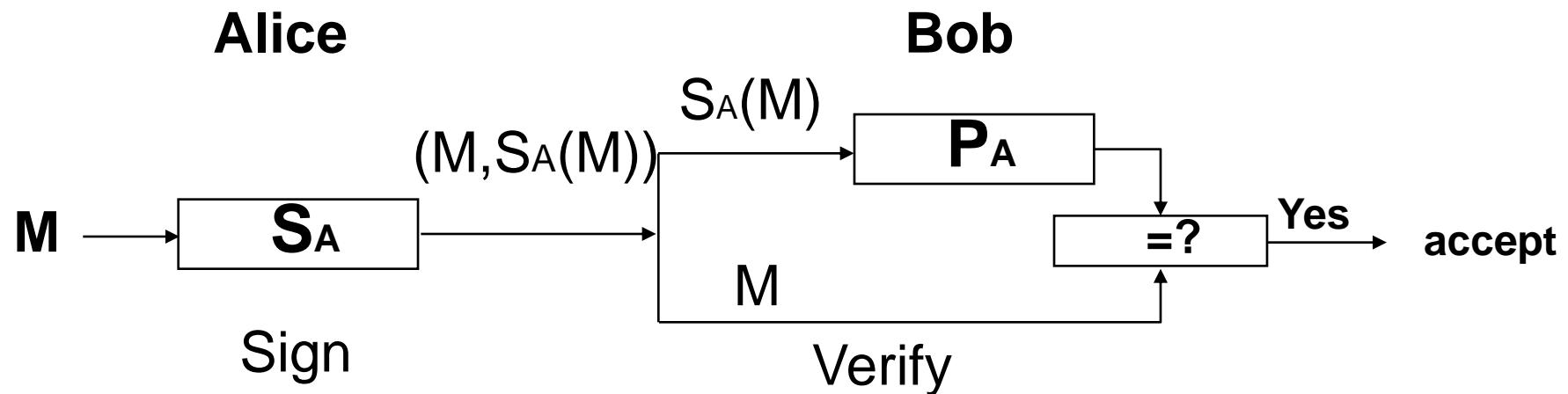
- 特征：

- 公开加密密钥，而解密密钥保密。

- 公钥系统加解密过程



- 数字签名过程



RSA系统

■ 构造过程

- ① 随机选择两个大素数 p 和 q , $p <> q$
- ② 计算 $n = pq$
- ③ 选择小的奇整数 e , 使得 e 与 $(p-1)(q-1)$ 互素
- ④ 计算 e 模 $(p-1)(q-1)$ 的逆 d , 即 $ed \equiv 1 \pmod{(p-1)(q-1)}$
- ⑤ 公布 e, n 为 RSA 的公钥
- ⑥ 保存 d, p, q 为 RSA 的私钥

■ 加解密过程

- 加密: $P(M) = M^e \pmod{n} (=C)$
- 解密: $S(C) = C^d \pmod{n}$

■ 示例: Alice 发送信息“9726”给 Bob

RSA的正确性

- Fermat定理

如果 p 是素数, 则 $a^{p-1} \equiv 1 \pmod{p}$

- Th31.36(Correctness of RSA)

RSA算法是正确的



数论算法 (Ch31)

1 初等数论记号 (31.1)

2 最大公约数 (31.2)

3 模运算 (31.3)

4 求解线性模方程 (31.4)

5 中国余数定理 (31.5)

6 RSA公钥密码系统 (31.7)

7 素数判定 (31.8)



素数判定 (31.8)

- 素数的分布
- 简单的素数测试算法
- 伪素数测试算法
- Miller-Rabin随机算法

素数的分布

- Def. :

令 $\pi(n)$ 表示小于或等于 n 的素数个数

如, $\pi(15) = 6$, 因为2, 3, 5, 7, 11, 13

- Theorem 31.37(Prime number theorem):

$$\lim_{n \rightarrow \infty} \frac{\pi(n)}{n / \ln n} = 1$$

注: 随机选择一个整数是素数的概率为 $1/\ln n$

简单的素数测试算法

■ Theorem:

如果 n 被某个整数 $2, 3, \dots, \lfloor n^{(1/2)} \rfloor$ 整除，则 n 不是素数。

注：

- 该算法最坏时间是 $\Theta(n^{(1/2)})$
- 如果 n 是 β bit，即 $\beta = \lfloor \log n \rfloor + 1$ ，则 $n^{(1/2)} = \Theta(2^{(\beta/2)})$

伪素数测试算法

■ Def. :

n 称为基 a 的伪素数，如果 n 是合数且满足

$$a^{(n-1)} \equiv 1 \pmod{n} \quad (31.8.1)$$

注：

- 由Fermat定理知：素数一定满足上式，但满足上式的 n 未必一定是素数。

如，341有 $2^{(340)} \equiv 1 \pmod{341}$ ，而 $341 = 11 \times 31$

- 伪素数比较稀少：10,000以内仅有22个，前4个为
341, 561, 645, 1105

- 系：如果存在 a ，使得(31.8.1)式不成立，则 n 一定是合数

伪素数测试算法

■ 伪素数测试算法

PseudoPrime(n)

```
{ if ModularExponentiation(2, $n-1$ , $n$ )  $\not\equiv$  1 (mod  $n$ ) then
```

```
    return Composite;
```

```
else
```

```
    return Prime;
```

```
}
```

注：

- 该算法判断出合数总是正确的；

- 如果算法给出的结论是素数，仅在基2的伪素数情形出错。

伪素数测试算法

■ Def. :

如果对所有的 $a \in Z_n^*$, 合数 n 满足(31.8.1)式, 则称 n 为Carmichael数。

注:

- Carmichael数非常稀少, 小于100,000,000的Car数仅有255个, 前3个数为561, 1105, 1729;
- 但这类数有无穷多个。
- 说明存在Car伪素数, 对任何基 a 都使得(31.8.1)式成立, 这样造成伪素数测试算法的失败。



Miller-Rabin 随机算法

- 伪素数测试算法的缺陷
- MR 算法的改进措施
- MR 随机算法
- 计算示例
- 算法的复杂性和误判率
- MR v.s. Pseudo Prime

伪素数测试算法的缺陷

- 仅选择一个基进行测试

如，对 $a=2$ ，用条件 $a^{n-1} \equiv 1 \pmod{n}$ 来判断。

存在的缺陷：

- 尽管出错率很小，但与 n 相关。

n 若是随机选取的512位二进制数，出错率为 $< 1/10^{20}$ ；

n 若是随机选取的1024位二进制数，出错率为 $< 1/10^{41}$ 。

- 选择所有的基进行测试

即，对所有 $a \in \{2, 3, \dots, n-1\}$ 。

存在的缺陷：

- 工作量很大，但并不能排除Carmichael数。

MR算法的改进措施

- 措施1：随机选择若干个基，以减少计算量

即，随机选择 s 个基 a ， $a \in \{2, 3, \dots, n-1\}$ 。进行Fermat定理测试。

- 措施2：在计算 $a^{(n-1)} \bmod n$ 过程中，加入合数证人(Witness)的查找

- 即查找是否存在模 n 余1的非平凡平方根
(依据P560系31.35)

- 系31.35

如果存在模 n 余1的非平凡平方根，则 n 是合数

MR随机算法

Alg. 由MR和Witness二个例程组成,

设 $n-1=2^t u$, 这里 $t \geq 1$, u 是奇整数

$$\therefore a^{n-1} \equiv (a^u)^{2^t} \pmod{n}$$

■ Witness(a, n)

```
{    $x_0 \leftarrow \text{ModularExponentiation}(a, u, n);$ 
    for i  $\leftarrow 1$  to t do
    {    $x_i \leftarrow (x_{i-1}^2 \bmod n);$ 
        if  $x_i = 1$  and  $x_{i-1} \neq 1$  and  $x_{i-1} \neq n-1$  then return True;
    }
    if  $x_t \neq 1$  then return False;
    return False;
}
```

MR随机算法

■ Witness(a, n)

```
{    $x_0 \leftarrow \text{ModularExponentiation}(a,u,n);$ 
    for i  $\leftarrow 1$  to t do
    {    $x_i \leftarrow (x_{i-1}^2 \bmod n);$ 
        if  $x_i=1$  and  $x_{i-1} \neq 1$  and  $x_{i-1} \neq n-1$  then return True;
    }
    if  $x_t \neq 1$  then return False;
    return False;
}
```

注: - 如果该算法返回True, 分二种情形:

- 1)从for循环里返回, 即找到了witness
 - 2)从for循环外返回, 即n不满足Fermat定理
- 如果返回False, 未找到witness

MR随机算法

■ MillerRabin(n, s)

```
{
```

```
    for j ← 1 to s do
    {   a ← random(1, n-1);
        if Witness(a, n) then
            return Composite;
```

```
}
```

```
    return Prime;
```

```
}
```

注:

- 返回合数， 结论是100%正确；
- 返回素数， 存在误判的可能；
- 问题的是算法的复杂性和误判率。

计算示例

- 示例： $n=561$ 是素数吗？

$n-1=560=2^4 \times 35$, 取 $a=7$ 作为测试基

$$x_0 \equiv a^{35} \equiv 241 \pmod{561}$$

...

$$\Rightarrow X = \langle 241, 298, 166, 67, 1 \rangle$$

最后一步表明，发现另外一个1的非平凡平方根，即

$$a^{280} \equiv 67 \pmod{561}, \quad a^{560} \equiv 1 \pmod{561}$$

因此， $a=7$ 是 n 为合数的一个证人(Witness)

算法的复杂性和误判率

■ 算法的复杂性

设 n 为 β -bit整数，

算法需要 $O(s\beta)$ 算术运算、 $O(\beta^3)$ 位运算。

■ 算法的误判率

- Th31.38

n 为奇合数，则 n 的Witness数大于 $(n-1)/2$ 。

- Th31.39

MR算法的误判率不超过 2^{-s} 。

MR v.s. Pseudo Prime

- 后者的误判率与 n 相关，不可控；
前者仅与 s 相关，可控制误差；
- MR算法的优点不存在坏的输入，缺点取决于 s 和基 a 的选择。
即witness带有幸运性质。
- MR算法是第1个多项式时间的随机算法，确定性算法直到
2002年由三位印度科学家提出。



End