第30卷 第1期
2009年2月

暨南大学学报(自然科学版)
Journal of Jinan University(Natural Science)

Vol. 30 No. 1
Feb. 2009

一类求模逆元的算法

黄伟亮[1]， 王 震[2]， 黄 勇[1]

(1. 广州大学计算机科学与教育软件学院，广东 广州 510006；2. 河南大学数学与信息科学学院，河南 开封 475001)

[摘 要] 对于同余方程 $p \cdot x \equiv 1 \bmod a^p$ 求解 $p^{-1} \bmod a^p$ 的问题，给出了一个求解公式，且编程实现了求解公式的代码算法. 与传统求解方法相比，此算法更为直观、简洁. 实际测试结果显示该算法是可靠有效的.

[关键词] 模逆元； 扩展欧几里得算法； 费马小定理

[中图分类号] TP301.6 [文献标识码] A [文章编号] 1000-9965(2009)01-0057-04

An algorithm to solve some class of modular inverses

HUANG Wei-liang[1], WANG Zhen[2], HUANG Yong[1]

(1. Institute of Educational Software, Guanzhuo University, Guangzhou 510006, China;
2. College 1 of Mathematics and Information Sciences Henan University, Kaifeng 475001, China)

[**Abstract**] A brief formula to solve the modular inverse, x satisfying the congruence equation $p \cdot x \equiv 1 \bmod a^p$ is given. The programming code algorithm for solving this formula is also presented. Comparing with the traditional methods, the algorithm is more intuitive and simple. The actual test results indicate that the algorithm is reliable and effective.

[**Key words**] modular inverse; extended euclidean algorithm; fermat little theorem

求解模逆元算法是计算机代数中最重要的一个问题. 随着计算机网络技术和通信技术的发展，求解模逆元算法问题在通信理论、信息安全、密码学等领域有着广泛的应用. 在RSA公钥密码体制中，加密和解密的密钥都涉及模逆元的运算问题[1]. 另外，在有限数域上的椭圆曲线公钥密码系统和数字签名方案中，在选择仿射坐标系的情况下，需要频繁地用到模逆运算[2-3]. 因此，研究求解模逆元问题既具有重要的理论意义，又具有很强的现实意义.

1 问题的提出

设整数 a,x,p 满足 $a \cdot x \equiv 1 \bmod p$. 则称 x 为 a 模 p 的乘法逆元，即 $x = a^{-1} \bmod p$. 扩展欧几里得算法是求模逆元的一个重要工具[4]. 其基本思想是：利用两整数的最大公约数可用整系数线性组合的方式表示的性质，重复利用带余除法，并记下相应的系数，直至余数为零时止. 其计算过程可用下组式子来描述[5]，

[收稿日期] 2008-01-04

[基金项目] 广州市科技计划科技攻关重点项目(2006Z2-D0181)；广州市属高校科技计划项目(62010)

[作者简介] 黄伟亮(1974-)，男，讲师，硕士，研究方向：计算机自动推理与应用软件. 通讯作者：黄勇，教授，博士

$$\begin{cases}\gcd(p,a)=\gcd(a,r_1)=\cdots=\gcd(r_{N-2},r_{N-1})=r_N\\ r_i=s_i\cdot a+t_i\cdot p(1\leqslant i\leqslant N)\end{cases} \tag{1}$$

这里 r_i 为每次相除所得的余数,s_i,t_i 为其相应的整数系数. 尤其当

$$r_N=\gcd(p,a)=s_N\cdot a+t_N\cdot p=1.$$

则 $a^{-1}=s_N=(1-t_N\cdot p)/a.$

这里 s_N 就是所求 a 模 p 的逆元.

对形如 $a^p\cdot x\equiv 1\bmod p(a\in\mathbf{Z},p$ 是质数)同余方程,求 a^p 模 p 的逆元,由费马小定理[4-5],很容易求得其逆元值为 $a^{-2}\bmod p$. 但对形如 $p\cdot x\equiv 1 \bmod a^p$ 的同余方程,欲求 $p^{-1}\bmod a^p$ 的值,费马小定理就不适用了. 通常的求解方法是,先计算出 a^p 的值,然后,利用扩展欧几里得算法或是在其基础上进行某些改进而进行求解[6-9]. 除此之外,对于该类问题的逆元求解,还有没有其他的通用的计算方法? 更进一步,该类问题有没有类似费马小定理那样简洁的求解逆元的公式? 本文的研究工作,就是在这样的背景下展开的.

2 求解方法

对于形如 $p\cdot x\equiv 1\bmod a^p(a\in\mathbf{Z},p$ 是质数)的同余方程,求 x(即 $p-1\bmod a^p$)的问题与 $a^p\cdot x\equiv 1\bmod p(a\in\mathbf{Z},p$ 是质数)方程在形式上具有一定的相似性,于是我们很自然地会想到费马小定理:设 p 是素数,$a\in\mathbf{Z}$,则 $a^p\equiv a \bmod p$. 特别地当 $\gcd(a,p)=1$ 时,$a^{p-1}\equiv 1\bmod p$,即 $a^p\cdot a^{-2}\equiv 1\bmod p$. 由模逆元定义知,$a^{-2}\bmod p$ 就是 a^p 模 p 的逆元.

由于本文所要求的方程问题与费马小定理所涉及的内容有一定的相似性,所以,本文就从模逆元的存在条件、最大公约数的整系数线性组合表示等性质方面对该问题进行分析和探讨,以期找到该类问题的通用求解方法.

2.1 模逆元存在的充要条件

对任意两个整数 a,p,a 模 p 存在逆元的充要条件是,$\gcd(a,p)=1$[4-5]. 由费马小定理可推知,$a^p\cdot a^{-2}\equiv 1\bmod p$,故依据模逆元的存在条件和同余的性质知,$\gcd(a^p,p)=1$. 即 $p^{-1}\bmod a^p$ 一定存在.

2.2 $\gcd(a^p,p)$ 的线性组合表示

求任意两整数最大公约数的经典算法是辗转相除法. 其计算过程如式(1)所示,从中可得到最大公约数的一个重要性质,即两整数的最大公约数可用整系数线性组合的形式来表示. 因此,当时 $\gcd(a^p,p)=1$ 时,a^p,p 这两个数的最大公约数就可用式(2)来表示,

$$\gcd(a^p,p)=s_N\cdot a^p+t_N\cdot p=1,s_N,t_N\in\mathbf{Z} \tag{2}$$

2.3 变换处理

由同余的性质,可将式(2)两边同时模 p 并移项得,

$$s_N\cdot a^p\equiv 1\bmod p \tag{3}$$

依据费马小定理可知,$a^p\equiv a\bmod p$. 所以,就可直接计算出式(3)中的 s_N,即 $s_N=a^{-1}\bmod p$. 因此,把 s_N 代入式(2)可得,

$$t_N=(1-s_N\cdot a^p)/p \tag{4}$$

则 t_N 就是所求的 $p\bmod a^p$ 的逆元解,即 $t_N=p^{-1}\bmod a^p$.

综上所述,对形如 $p\cdot x\equiv 1\bmod a^p(a\in\mathbf{Z},p$ 是质数)的同余方程,求解 $p^{-1}\bmod a^p$ 的问题,可以分为两个步骤进行. 首先,求出 $s_N=a^{-1}\bmod p$ 的值;其次,利用式(4)就可直接计算出 $p^{-1}\bmod a^p$ 的值,即 $x=t_N=p^{-1}\bmod a^p=(1-s_N\cdot a^p)/p$.

与通常的求模逆元的方法相比,该方法不是首先计算 a^p(注:a^p 值可能是大整数如 256、512 和 1 024位的整数情况)的值,而是先计算 $a^{-1}\bmod p$ 的大小(通常情况下,a,p 都是一个机器整型所表示的 2 个字节或 4 个字节的整数);其次,就可通过本文所提供的公式直接进行计算求解.

3 算法实现

依据上述分析,对方程 $p\cdot x\equiv 1\bmod a^p(a\in\mathbf{Z},p$ 是质数),求解 $p^{-1}\bmod a^p$ 的问题,主要是先通过转换去计算 $a^{-1}\bmod p$ 的值,然后,把计算结果代入求解公式就很容易地求出 $p^{-1}\bmod a^p$ 的值. 按照这种处理问题的思路,本文给出了相应的实现代码. 下面是用 VC6.0 实现的程序代码.

```
void Solve_Inverse(int a, int p,CString &strInverse)
{  //变量的声明及初始化
int r,r1,r2,s,s1,s2,t,t1,t2,q;
CString temp;
r1=a,r2=p,s1=1,s2=0,t1=0,t2=1;
r=r1%r2;
```

```
while(r! =0)   //先计算 a^1 mod p
  {
    q = r1/r2;
    s = s1 - q*s2;
    s1 = s2;s2 = s;
    t = t1 - q*t2;
    t1 = t2;
    t2 = t;
    r1 = r2;
    r2 = r;
    r = r1%r2;
  }
  //把上步计算的结果 r_2 即 s_N,代入 s_N · a^p + t_N · p = 1,
  //就可计算 p^-1 mod a^p
temp = "(1 - ";
strInverse = temp;
temp. Format("%d",s2);
strInverse = strInverse + temp + "*";
temp. Format("%d%c%d",a,'^',b);
strInverse + = temp + ")/";
temp. Format("%d",b);
strInverse + = temp;
}
```

以上就是我们针对形如 $p \cdot x \equiv \bmod a^p$ 的同余方程定义的求解 x 的代码函数,void Solve_Inverse (int a, int p,CString &strInverse). 其使用就是在调用该函数时,向其传递相应的实参,就可得到所需计算结果的精确表达式.

4 算例测试及结果分析

4.1 算例测试

该算法的时间复杂度为 $O(\log(\max(a,p)))$. 根据复杂度表达式,从理论上可得出:随着 a,p(a,p 都在一个机器整型所表示的范围之内)增大,该算法的时间消耗量的增加幅度变化不是很大,但是在最后的具体数值计算时,会涉及大整数的计算,由于 VC 本身没提供有关大整数运算的代码模块,故本文的上述代码实现部分就只给出了所求逆元的解析表达式算法.

为说明该算法的可靠性和可行性,下面的表 1 给出了几个实际的数据测试用例,其结果的准确性可用第三方的数学软件进行验证(本文用的是 Maple 数学软件). 实验操作环境为,WinXP 操作系统,Visual C ++ 6.0, Maple10.0, Pentium (R) 4 CPU 1.8GHz, 256MB 内存.

表 1　对三组数据进行计算所得结果与用 Maple 计算所得结果的对比

测试数据	$a^{-1} \bmod p$ [1]	$p^{-1} \bmod a^p$ [1]	Maple 软件计算所得结果
$a=3$, $p=7,3^7$	5	$\frac{1-5\cdot 3^7}{7}=-1\,562$	625
$a=7$, $p=11,7^{11}$	8	$\frac{1-8\cdot 7^{11}}{11}=-1\,438\,055\,813$	539 270 930
$a=13$, $p=17,13^{17}$	4	$\frac{1-4\cdot 13^{17}}{17}=-2\,035\,391\,981\,030\,903\,043$	6 615 023 938 350 434 890

1)表中 $a^{-1} \bmod p$ 即为算法中的 s_N,$p^{-1} \bmod a^p$ 即为 t_N.

4.2 结果分析

由表 1 可以看出,对形如 $p \cdot x \equiv \bmod a^p$ 的同余方程,在求解 $p^{-1} \bmod a^p$ 时,用本文给出的方法计算所得结果都为负值;而用 Maple 软件计算求得的结果均为正值,且其绝对值的大小均不相同. 本小节将对这两种不同的结果进行深入地分析和探讨.

这里,以表中的第一行数据为例进行分析、比较. 此处 $a=3$,$p=7$. 求 $p^{-1} \bmod a^p$ 的值,用本文的方法得到的结果为 $-1\,562$,用 Maple 软件得到的计算结果为 625. 进一步比较这两个结果之间的关系,可发现这两个数值满足关系式,$|-1\,562| + |625| = 2\,187 = 3^7$,即二者的绝对值等于 a^p 很显然,用本文所给方法计算所得的结果与用 Maple 软件计算所得的结果对于模数(a^p)而言,是一对符号相反,绝对值之和等于模数的互补关系. 同理,对其他两组数据进行分析、比较,其结果也都满足上述互补关系.

更进一步,我们对任给定的两个整数 a,p(不妨假定 $p > a$),依据模逆元存在条件及最大公约数可以线性整系数组合表示的性质可知,必存在相应的整数 s_1,t_1 使得,

$$\gcd(p,a) = s_1 \cdot a + t_1 \cdot p = 1.$$

这里不妨假设 $s_1 > 0$,并令 $s_2 = s_1 - p$,在该条件下,若存在一个相应的整数 t_2,使上式成立,就能说明,两个符号相反,但其绝对值之和等于模数的这两个数都是所求的逆元解. 下面把 $s_2 = s_1 - p$,代如上式得,

$$s_2 \cdot a + (t_1 + a) \cdot p = 1$$

显然,$t_2 = t_1 + a$,因此,s_1, s_2 这两个符号相反,绝对值之和等与模数的数,都是符合要求的逆元解.

事实上,用本文算法给出的数值(即负整数)与用 Maple 软件计算所得的数值(即正整数)都是关于模数(a^p)的一对互补关系,即符号相反,绝对值之和等于模数的关系. 因此,二者在逻辑上是等价的,即用本文提供的算法计算所得到结果与用 Maple 计算所得结果都是符合要求的逆元解.

5 结语

本文给出了求 $p \cdot x = \mathrm{mod}\ a^p$ 方程解的一个精确的解析表达公式算法. 该算法主要分为两个步骤:①计算 $a^{-1} \mathrm{mod}\ p$;②把 $a^{-1} \mathrm{mod}\ p$ 的值代入给定的公式即可. 实际数据测试表明,本文所给出的公式算法是可靠有效的. 同时,与用 Maple 软件计算所得的结果相比,用该算法所得的计算结果都为负值. 若需要正数解,只需在所给公式等式的右边加上一个常数(即模数 a^p),就可得到所需结果. 此外,本文还提供了一个通用的公式求解算法,这相对于一般的求模逆元算法而言显得更为直观简洁.

[参考文献]

[1] 卢开澄. 计算机密码学—计算机网络中的数据保密与安全[M]. 2版. 北京:清华大学出版社,1998.

[2] 陈海进. 奇数模模逆算法对偶数模的推广[J]. 计算机应用软件,2005,22(5):100-101.

[3] AMKI K, FUJITA L, MORISUE M. Fast inverters over Finite Field Based on Euclid's Algorithm[J]. Trans IEICE, 1998, E-72(Ⅱ):1230-1234.

[4] VON ZUR GATHEN J, GERHARD J. Modern Computer Algebra[M]. Cambridge: Cambridge University Press, 2001.

[5] 李继光,余纯武. 信息安全数学基础[M]. 武汉:武汉大学出版社,2006.

[6] 谭丽娟,陈 运. 模逆算法的分析、改进及测试[J]. 电子科技大学学报,2004,33(4):383-386,394.

[7] 陈 军,侯紫峰. 一种快速适合嵌入式环境的求模逆元算法[J]. 计算机工程,2006,32(16):163-164.

[8] 刘文江,董 威. 有限域逆元算法的实现[J]. 计算机工程,2004,30(17):184-185.

[9] 袁丹寿,戎蒙恬. 基于改进欧几里德算法的可重构性逆元结构[J]. 上海交通大学学报,2006,40(1):36-40.

[责任编辑:王景周]