文章编号:1005-0523(2011)05-0052-05

基于频率的大素数高效生成算法

汤鹏志,李 彪

(华东交通大学基础科学学院,江西 南昌 330013)

摘要:公钥密码体制加解密算法基于两个大素数乘积的难分解性。为了提升大素数生成算法的效率和降低算法的报错率,提出了一种基于概率论的方法,通过优化Eratosthenes筛法构建素数库,从而通过分析素数库中素数尾数的分类频数和表达式下素数频率,再通过对素数检验算法进行分析,最后得到一种高效的大素数生成算法。在算法中,任意初始的整数都具有较高的素数概率,从根本上提升了算法的执行效率。

关键字:素数;Miller-Rabin算法;检验;生成;效率

中图分类号:TP301 **文献标志码**:A

1977年Ron Rivest、Adi Shamirh和Len Adleman提出公钥密码体制(RSA)加解密算法,其安全性在于在计算机上生成两个足够长度的大素数,其乘积具难分解性[1-2]。目前,素数确定性算法主要分为递归试除法,Eratosthenes筛法,Miller检验和多项式时间内判定的素数(AKS)算法。文献[3]分析表明,Eratosthenes筛法是一种较好的素数确定性算法。根据文献[4],任意素数(除2和3)均可表示为$6k\pm1$ $\left(k\in N^{+}\right)$的理论,可提升筛法效率。生成大素数的方法是随机产生一个大整数,然后进行素性检测。文献[5]提出一种用概率方法来研究素数的分布,是提升大整数素性的有效途径。常用的素数检验算法主要为Fermart算法、Solovay - Strassen检测法、Lehman检测法、Miller检验、Miller-Rabin检测法和AKS算法[6]。通过分析当今的大素数生成方法与原理[7-11],提升大整数的素性和选择优良的素数检验算法,从而提出一种改进的大素数的高效生成算法。

1 构建素数库

对素数的分布频率研究,需建立在素数库的基础之上。在素数的研究领域中,确定性素数算法存在很多,如试除法,Eratosthenes筛选法,Miller检验,AKS算法等。在素数算法中,Eratosthenes筛法应用非常广泛。传统Eratosthenes筛法需要对每个数依次比较,算法的效率较为低下,通过对Eratosthenes筛法进行优化,得出素数库构建算法。

定理1 除2和3之外,素数均可表示成$6k\pm1$ $(k\in N)$的形式。

定理2 如果k为素数,则$k\times j$ $\left(j\in N^{+}\right)$必为合数。

推论1 在Eratosthenes筛法中对寻求到的第一个素数k,在区间$\left(1,k^{2}\right)$没有被划掉的数均为素数。

定义一个bool型素数判定数组$p[n+1]$。根据定理1,当$i=2$或$i=3$或$i=6k+1$或$i=6k+5$ $\left(k\in N^{+}\right)$时,$p[i]=\text{true}$;否则$p[i]=\text{false}$。寻找第一个$p[i]=\text{true}$ $\left(i{:}5\mapsto\sqrt{n}\right)$的素数$i$,$i$的倍数$j$ $\left(j\in[0,n/i]\right)$均为合数。另外,数组赋值时已去除偶数,将内层循环j限定为奇数。故当素数为i $\left(i\in\left[2,\sqrt{n}\right]\right)$时,

收稿日期:2011-08-27

基金项目:国家自然科学基金项目(11061014);江西省教育厅青年科学基金项目(GJJ10129);江西省教育厅科研项目(GJJ10708)

作者简介:汤鹏志(1964—),男,教授,硕士,主要研究方向为信息系统及其安全。

$j=i*i+ik\ (k\in 2N, j<n)$ 有 $p[j]=\text{false}$ 。最后所有满足 $p[i]=\text{true}\ (2\leqslant i\leqslant n)$ 的 i 就是区间 $[2,n]$ 的所有素数。算法如下

步1　定义bool参数 $a=\text{true}$

步2　定义int参数 $b=\sqrt{n}$

步3　for $i=1$ to n

　步2.1　若 $i\neq 1(i=2\|\|i=3\|\|i\%6=1\|\|i\%6=5)$，$p[i]=\text{true}$

　　否则，$p[i]=\text{false}$

　步2.2　$i=i+1$

步4　for $i=5$ to b

　步4.1　若 $p[i]=\text{true}$

　步4.2　for $j=i*i$ to n

　　步4.2.1　$p[j]=\text{false}$

　　步4.2.2　$j=j+i+i$

　步4.2　若 $a=\text{true}$，$i=i+2$；否则，$i=i+4$

　步4.3　$a=!a$

步5　用文件流打开PrimeNumber.txt

步6　for $i=2$ to n

　步6.1　若 $p[i]=\text{true}$，将 i 写入文件中

　步6.2　$i=i+1$

步7　关闭文件流

2　素数的频率分布

素数具有无穷多个，并且随区间变大而逐渐变得稀疏。通过运行素数库构建算法，得到区间 $(0,10^9)$ 中的素数库，从而考察素数的尾数分布情况。

定义1　在某个区间范围内，用素数模100的余数进行分类，各类余数中拥有素数的个数称为素模百频数。

表1　各区间的素模百频数

Tab.1　Frequency of module in various regions

尾数	$(0,10^5)$	$(0,10^6)$	$(0,10^7)$	$(0,10^8)$	$(0,10^9)$
1	243	1 964	16 573	143 919	1 271 026
2	1	1	1	1	1
3	237	1 969	16 650	144 127	1 271 574
5	1	1	1	1	1
7	237	1 932	16 552	144 141	1 271 474
9	237	1 957	16 586	143 783	1 271 233
11	244	1 958	16 670	143 965	1 271 002
13	243	1 965	16 585	144 145	1 271 599
17	244	1 970	16 594	144 007	1 271 520
19	238	1 973	16 598	143 978	1 270 953
21	243	1 937	16 623	143 934	1 270 775
23	246	1 976	16 623	143 905	1 270 751
27	244	1 976	16 612	143 946	1 271 095
29	245	1 926	16 546	144 051	1 271 534

通过表1可知，素数尾数的分布大致相同，即任何尾数在区间中所占的素数的概率是同等的。如若一

个整数在生成之时，已经排除存在部分素因子，则它是素数的可能性会更大。

定义2 在某个区间范围内，满足表达式 $\Pi Pi\times k+1$（Pi 为小素数）的整数中，所占素数的个数称为素积频数。

表2 各区间的素积频数

Tab.2 Frequency of prime number in various regions

表达式	$(0,10^5)$	$(0,10^6)$	$(0,10^7)$	$(0,10^8)$	$(0,10^9)$
$2\times k+1$	9592	78 498	664 579	5 761 455	50 847 534
$2\times 3\times k+1$	4784	39 231	332 194	2 880 517	25 422 713
$2\times 3\times 5\times k+1$	1189	9 807	83 003	719 984	6 355 189
$2\times 3\times\cdots\times 7\times k+1$	193	1 623	13 786	120 067	1 058 918
$2\times 3\times\cdots\times 11\times k+1$	16	158	1 359	11 997	105 779
$2\times 3\times\cdots\times 13\times k+1$	0	16	111	959	8 817

通过观察表2中的数据，随着过滤的素因子的个数增加，区间内的素积频数逐渐减少。但随着表达式过滤素因子的个数增加，无法清晰得到该表达式下整数是素数的可能性变化状况。

定义3 在某个区间中，满足表达式 $\Pi Pi\times k+1$（Pi 为小素数）的素积频数与整数个数比值，称为素积频率。

表3 各区间的素积频率

Tab.3 Frequency of prime number in various regions %

表达式	$(0,10^5)$	$(0,10^6)$	$(0,10^7)$	$(0,10^8)$	$(0,10^9)$
$2\times k+1$	19.18	15.70	13.29	11.52	10.17
$2\times 3\times k+1$	28.70	23.54	19.93	17.28	15.25
$2\times 3\times 5\times k+1$	35.67	29.42	24.90	21.60	19.07
$2\times 3\times\cdots\times 7\times k+1$	40.46	34.08	28.95	25.21	22.24
$2\times 3\times\cdots\times 11\times k+1$	36.36	36.49	31.39	27.72	24.43
$2\times 3\times\cdots\times 13\times k+1$	0	47.06	33.23	28.79	26.48

通过分析表3中的数据，随着过滤的素因子的个数增加，区间内的素积频率逐渐增大。通过实践表明：采用小于512的素数可以淘汰大约82%的非素整数。

3 素数检验算法分析

素数检验算法分为确定性检验算法和概率检验算法。通过确定性检验算法的得到的数一定是素数，Miller检验和AKS算法就是常用的确定性算方法。通过概率性检验算法得到的数只是伪素数，Fermart算法、Solovay-Strassen检测法、Lehman检测法和Miller-Rabin检测法是现今流行的概率算法。确定性检验算法需要深厚的数学理论基础，算法的实现相当复杂。而概率检验算法虽然存在一定的概率得到伪素数，但经过多次测试可将报错率控制在极低的范围内。通过表4的算法分析，可知Miller-Rabin算法是素数检验算法中的最佳选择。

表4 素数检验算法分析情况

Tab.4 Analysis of prime number inspection algorithm

算法	复杂度	报错概率	算法分析
Fermat	$O(\log n)^{1+\mu}$	1/2	概率算法
Solovay - S tr assen	$O(\log n)^{1+\mu}$	1/2	概率算法
Lehman	$O(\log n)^{1+\mu}$	1/2	概率算法
Miller	$O(\log n)^{3+\mu}$	0	确定性算法
Miller-Rabin	$O(\log n)^{1+\mu}$	1/4	概率算法
AKS	$O(\log n)^{3(1+\mu)}$	0	确定性算法

注：普通乘法时，$\mu=2$；快速乘法时，$\mu=1+\varepsilon\ (0<\varepsilon<1)$。

4　大素数生成算法

通过改进ISO/IEC标准[12]，得出一种高效的大素数生成算法。在实际的工程中，如若需要生成一个1 024位（二进制数）的素数 q，则必须满足 $q\in\left[q_{\min},q_{\max}\right]$，其中 $q_{\min}=2^{1023}+1$，$q_{\max}=2^{1024}-1$。依据分析，满足表达式 $2\times3\times5\times\cdots\times509k+1(k\in N)$ 的整数已经滤掉了所有小于512的素因子，该数在生成时就已经过滤掉80%以上的非素数。在素数的生成和检验算法实施前，需要计算参数 m,n,k_j，其中 $1\leqslant j\leqslant97$ 且 $j\in N$，$m=2^{k1}\times3^{k2}\times5^{k3}\times\cdots\times503^{k96}\times509^{k97}$，$n=2\times3\times5\times\cdots\times503\times509$，并且 m 必须满足 $m+1\geqslant q_{\min}$。在算法中，每迭代一次整数 q 需要加 n，其中 $q=q+n$，保证筛选的范围内的整数的个数，必须对 n 的数量级进行分析。由于 $n=2\times3\times5\times\cdots\times503\times509<2^1\times2^2\times2^3\times\cdots\times2^{10}\times2^{10}=2^{746}$，则在筛选的范围中存在的整数个数 $i=2^{1023}/2^{746}=2^{277}$ 个。在如此大的区间中，一定能够找到若干个素数。算法如下

输入：预处理参数 m,n

输出：一个素数 $q\in\left[q_{\min},q_{\max}\right]$

步1　$q=m+1$

步2　计算 r 和 R，使得 $q-1=2^rR$，其中 R 为正奇数，r 为正整数

步3　随机选择 a，$a\in Z_n\backslash\{0\}$

步4　$y=a^R(\mathrm{mod}\,q)$

步5　若 $y=1$ 或者 $y=q-1$，跳转至步8

步6　for $i=1$ to $r-1$

　　步6.1　若 $y=n-1$，跳转至步8

　　步6.2　$y=y^2(\mathrm{mod}\,q)$

　　步6.3　若 $y=1$，跳转至步7

　　步6.4　$i=i+1$

步7　$q=q+n$，跳转至步2

步8　输出素数 q，结束

5　结论分析

本文提出了一种对素数尾数的分类频数和表达式下素数频率的分析方法，通过小素数对整数进行初次过滤，经过对素数检验算法的全面剖析，最后使用Miller-Rabin算法对形如 $\Pi Pi^m\times n+1$（Pi 为小素数）的整数进行检验。算法中通过利用1 024位和746位的两个大整数空间存储中间数据，跳跃大整数含有小素数因子的可能，降低算法的循环遍历次数，从而提升了算法的运行效率。

表5数据表明，本文的大素数生成算法可以快速的生成若干个大素数，从而为RSA加解密算法提供强有力的安全保障。

表5　生成1 024 bit素数的性能比较

Tab.5　Performance comparison of forming prime number 1024 bit

算法	平均用时/ ms
传统算法	4 420.65
本文算法	1 367.37

参考文献：

[1] ARTO SALOMAA. 公钥密码学[M]. 北京：国防工业出版社，1988：28-45.

[2] 潘峰,申军伟. 一种强素数因子分解的量子算法[J]. 计算机工程与应用,2010,46(10):73-77.
[3] HENRI COHEN. A course in computational algebraic number theory[M]. 北京:世界图书出版公司,1996:17-36.
[4] 王名利. 自然数是否为素数的两个判定方法[J]. 数学通报,2010,49(6):47-49.
[5] 宋富高. 双重概率筛法与素数分布[J]. 深圳大学学报:理工版,2003,20(4):61-107.
[6] 秦晓东,辛运帏,卢桂章. Miller-Rabin算法研究与优化实现[J]. 计算机工程,2002,28(10):55-57.
[7] 张四兰,夏静波,余荣威.可信赖的高效素数生成和检验算法[J]. 计算机工程与应用,2005,41(30):31-34.
[8] 夏静波,陈建华. 一种快速的素数生成和检验算法[J]. 武汉大学学报:理学版,2005,52(S2):25-27.
[9] 张宏,刘晓霞,张若岩. RSA公钥密码体制中安全大素数的生成[J]. 计算机技术与发展,2008,18(9):131-134.
[10] 戴经国,张韶华,易叶青,等.生成大素数的一个方法[J]. 科学技术与工程,2007,7(14):3510-3511.
[11] 游新娥. RSA算法中安全大素数生成方法研究与改进[J]. 北京电子科技学院学报,2007,15(2):14-16.
[12] ISO/IEC. WD 18032-2000 Prime number generation[S]. United States:F A Schermer,2002.

Efficient Generation Algorithm of Big Prime Number Based on Frequency

Tang Pengzhi, Li Biao

(School of Basic Sciences, East China Jiaotong University, Nanchang 330013, China)

Abstract: The security of RSA encryption algorithm is based on the decomposability of product of two big prime numbers. To ensure the security of RSA, two big prime numbers of sufficient length have been generated. A method based on probability theory is proposed in order to improve the efficiency of this algorithm and reduce the rate of error. A prime library is constructed through optimizing of Eratosthenes method, then by analyzing classification frequency of the prime mantissa in library, prime frequency of expression and the prime checking algorithm, and an efficient algorithm of big prime generation has been established. In this algorithm, arbitrary initial integer has larger prime probability, whichmay improve the efficiency of the algorithm.

Key words: prime number; Miller-Rabin; checking; generation; efficiency