基于 Markov 链的椭圆曲线标量乘法算法性能分析

唐 文, 唐礼勇, 陈 钟
(北京大学信息科学技术学院信息安全实验室, 北京 100871)

摘 要: 在椭圆曲线密码系统中, 采用规范重编码、滑动窗口等优化技术可以有效提高椭圆曲线上点的标量乘法 k#P 的运算性能, 但在实现中, 需要对不同优化技术的算法性能进行定量分析, 才能确定标量乘法的最优实现. 本文运用 Markov 链对标量 k 规范重编码表示的滑动窗口划分过程进行了建模, 提出了一种对椭圆曲线标量乘法的平均算法性能进行定量分析的方法, 并运用该方法分析了不同参数下标量乘法运算的平均性能, 计算了滑动窗口的最优窗口大小. 最后, 通过比较说明, 采用规范重编码和滑动窗口技术的椭圆曲线标量乘法的运算开销比用 m2ary 法少 10132 ~ 17132%, 比单纯采用滑动窗口法也要少 4153 ~ 8140%.

关键词: 椭圆曲线密码系统; 标量乘法; 规范重编码; 滑动窗口; Markov 链
中图分类号: TN309 文献标识码: A 文章编号: 0372-2112 (2004) 12-1778-04

A Markov-Chain Based Performance Analysis Method for Scalar Multiplication on Elliptic Curve

TANG Wen, TANG Li-yong, CHEN Zhong
(School of Electronic Engineering and Computer Science, Peking University, Beijing 100871, China)

Abstract: The canonical re-coding and sliding window techniques are often used in computation of scalar multiplication k#P on elliptic curves for reducing the average number of required operation. In this paper, scalar multiplication with canonical re-coding and sliding window techniques is analyzed by modeling the window partition process of canonical re-coding expression of k as Markov-chain, the average performance of scalar multiplication under different parameters are given and the optimal window sizes are computed. Finally, the comparison shows that scalar multiplication with canonical re-coding and sliding window techniques requires 10132 ~ 17132% fewer operations than m-ary method, and 4153 ~ 8140% fewer operations than simple sliding window method.

Key words: elliptic curve cryptosystem; scalar multiplication; canonical re-coding; sliding window; Markov-chain

1 引言

在椭圆曲线密码系统中, 需要对很大的 k (通常 $n = \log_2 k \setminus 128$) 计算点的标量乘法 k#P[1], 而一个椭圆曲线密码系统是否具有实际价值往往就取决于其标量乘法的算法性能.

当采用已有 2000 多年历史的二进制法[2] (又称"平方且相乘法") 时, 由于标量 k 通常是随机产生的, 可以假定 k 的二进制表示中非零位 (等于"1"的位) 的数目大约为 k 的长度的一半, 即为 (n - 1)/2, 采用二进制法实现椭圆曲线标量乘法平均只需 n - 1 次倍点运算和 (n - 1)/2 次点加运算. 对 k 比较大的应用而言, 二进制法是实现椭圆曲线标量乘法的基本算法. 但二进制法还可与规范重编码技术和滑动窗口技术结合起来, 进一步减少所需的运算量, 提高椭圆曲线标量乘法的算法性能.

在具体实现中, 椭圆曲线密码系统可以选择若干种不同的坐标系. 在不同坐标系中, 椭圆曲线的点加和倍点运算的速度是不同的. 假设大整数乘法的运算时间为 M, 大整数平方的运算时间为 S, 而大整数模逆的运算时间为 I. 在仿射坐标系中, 点的加法的运算开销即为 I + 2M + S, 倍点运算的开销为 I + 2M + 2S. 其中, S/M 的值与所采用的坐标系和具体的实现无关, 一般为 0.8 左右, 而 I/M 的值一般都在 9 以上[3]. 因此, 在仿射坐标系中模逆运算是素数域椭圆曲线密码系统的主要性能瓶颈, 在实现中常采用 Jacobian 坐标系[1] 来避免模逆运算. 由于椭圆曲线标量乘法所需的倍点运算的数量要比点的加法运算要多得多 (平均约为 2 到 3 倍), 文献[1] 提出了一种改进的 Jacobian 坐标系, 通过略为增加点加运算的开销来减少倍点运算的开销, 从而从整体上减少椭圆曲线标量乘法运算所需的运算量. 在改进的 Jacobian 坐标系中, 点加运算的开销为 13M + 6S, 倍点运算的开销为 4M + 4S. 在此选择改进的 Jacobian 坐标系中的素数域椭圆曲线标量乘法作为本文的研究对象, 并定义点的加法的运算开销为 A = 13M + 6S = 17.8M, 倍点运算的开销则为 D = 4M + 4S = 7.2M.

由于存在多种优化技术, 所以在具体实现椭圆曲线标量乘法运算时, 需要对不同实现的算法性能进行定量分析, 才能确定最优的实现算法. 文献[4, 5] 采用 Markov 链分别对规范重编码法和滑动窗口法的运算性能进行了建模和定量分析.

收稿日期: 2003-12-18; 修回日期: 2004-08-16
基金项目: 北京市自然科学基金 (No. 4042016)

但这些研究工作的不足之处在于没有对同时采用规范重编码和滑动窗口技术的情形进行研究, 而在椭圆曲线密码系统中这两种技术实际上是可以结合在一起以提高标量乘法的算法性能的. 所以, 本文在上述研究工作的基础上提出了运用Markov 链对采用了规范重编码和滑动窗口技术的椭圆曲线标量乘法的平均性能进行建模分析的方法, 计算了在标量 k 采用规范重编码表示时滑动窗口的最优大小, 比较了采用不同优化技术的椭圆曲线标量乘法的平均算法性能.

2　规范重编码技术和滑动窗口技术

无论采用哪种方法, 计算椭圆曲线标量乘法 k#P 都需要进行 n- 1(n = log$_2$ k)次倍点运算. 但采用规范重编码技术和滑动窗口技术却可以减少 k#P 所需的点加运算的数量, 从而提高整个椭圆曲线标量乘法的算法性能.

2.1　规范重编码技术

重编码技术最先在乘法算法[6]中得到应用, 其目的是获得二进制数的稀疏表示以加速乘法运算. 这类重编码技术等价于用带符号的数字表示将 $2^{m-1}+\cdots+2+1$ 转换为 2^m-1, 以消除二进制表示中由连续的- 1. 构成的位串, 例如将 15 的二进制表示- 1111. 转变为- 1000$\bar{1}$. (即 16- 1= 15). 在基底为 2 的带符号数字表示中, 数字集合由三种符号 $\{\bar{1}, 0, 1\}$ 构成, 其中在第 i 位处出现的 1 或 $\bar{1}$ 分别表示+ 2^i 或- 2^i.

采用重编码技术得到的带符号数字向量 D= (D_{n-1}, D_{n-2}, , , D_0) 中不会含有相邻的非零位, 即满足 $D_i \# D_{i-1} = 0$ (0[i[n- 1), 满足这种特征的 D 被称为规范带符号数字向量[7, 6], 可以证明对给定的 k 存在唯一的规范带符号数字向量. 重编码技术亦称规范重编码技术.

对素数域椭圆曲线密码系统而言, 设 P 为椭圆曲线上的点, 已知 k#P= (x, y), 则(- k)#P= (x, - y) = (x, p- y) (p 为素数域的基底), 即已知 k#P 求(- k)#P 仅需一次减法. 所以, 在本文的讨论中始终假设已知 k#P 求(- k)#P 所需的时间可以忽略不计. 正是由于计算(- k)#P 的时间可以忽略, 在椭圆曲线密码系统中才可采用规范重编码技术以求解标量乘法问题.

2.2　滑动窗口技术

m2ary 法[2]是一种二进制法的改进算法. m2ary 法每次扫描标量 k 的 d(m= 2^d) 位(比特) 数字. 实际上, m2ary 法是通过预计算事先保存 x^i(1[i[2^d- 1), 然后将标量 k 的二进制表示分解成多个d 位的字, 再以这些 d 位字为运算单位, 利用事先保存的预计算结果来减少实际所需的运算数量.

滑动窗口法在 m2ary 法的基础上进行了改进. 滑动窗口法对等于零的字采用可变长度, 通过增加零窗口中所包含的总的位数来减少非零窗口的数目. 滑动窗口技术首先将标量 k 分解为零字(零窗口) 和非零字(非零窗口) . 其中, 非零字的大小为 d 位, 而零字的大小则可能不等于 d 位, 所以总的窗口数可能不等于 k/ d. 由于是从最高位开始分解 k 的, 没有理由从- 0. 位开始一个非零窗口. 所以, 如果 k_i 是非零窗口, 那么 k_i 的最高位必定为- 1. . 算法 1 在预计算(第 1 步)过程中只需对大小为 d 位, 且其二进制表示的最高位为- 1. 的数计算并保存 t#P 即可.

算法 1　采用滑动窗口技术的椭圆曲线标量运算

输入: 标量 k, 椭圆曲线上的点 P
输出: G = k#P
(1) 预计算并存储 t#P, 其中 t= 2^{d-1}, $2^{d-1}+1$, , , 2^d-1
(2) 将 k 的二进制表示分解为 m 个零窗口和非零窗口 k_i, 设其窗口大小为 L(k_i) , i = 0, , , m- 1
(3) G= k_{m-1}#P
(4) For i= k- 2 down to 0
　(a) For j= 1 to L(k_i)　　G= 2#G(椭圆曲线倍点运算)
　(b) if k_i X0 then　　G= G+ (k_i#P) (椭圆曲线点加运算, k_i#P 为第 2 步保存的预计算结果)
(5) 如果剩下的窗口的大小< d, 则采用普通的二进制法进行运算
(6) Return G

在算法 1 中, 将 k 分解为非零窗口和零窗口(第 2 步) 的具体实现方法是, 从高位到低位扫描 k 的二进制表示, 如果当前最高位为- 1. , 则取从当前位开始的连续 d 位作为非零窗口 k_i, 其大小 L(k_i) = d; 如果当前最高位为- 0. , 那么就取从当前位开始, 到最近的- 1. 位(下一非零窗口的开始) 之前的连续的若干个- 0. 位作为零窗口 k_i, 其大小 L(k_i) 就等于连续的- 0. 位的个数. 如此反复处理 k 的整个二进制表示. 所以, L(k_i) 就表示对当前窗口 k_i 而言, 所需进行的倍点运算的次数, 显然有 E L(k_i) = n- 1.

3　采用规范重编码和滑动窗口技术的椭圆曲线标量乘法

规范重编码技术和滑动窗口技术都是提高椭圆曲线标量乘法算法性能的有效方法. 在椭圆曲线密码系统中, 这两种优化技术可以结合在一起使用: 先对 k 进行规范重编码得到它的规范带符号数字向量, 再采用滑动窗口技术对规范带符号数字向量进行窗口划分.

由于 k 的规范带符号数字向量中不会含有相邻的非零位, 所以 d 位大小的非零窗口中的可能取值比在二进制表示下要少. 如当 d= 4 时, 非零窗口中的可能的取值只有- 1000. , - 1001. , - 1010. , - 1001$\bar{1}$. , - 1010. 0, 以及这些值的负值(对其所有的位取反) , 由于假设已知 k#P 求(- k)#P 的开销可以忽略不计, 所需预计算量仅为 3 次倍点运算和 4 次点加运算.

更一般地, 可以计算出当 d 为任意值时, 预计算的运算开销. 首先, d 位的窗口需要 d- 1 次倍点运算. 在缓存了所有中间运算结果的情况下, 所需点加运算的数目等于 d 位规范重编码表示中所有可能取值的数目减 1(- 10, 0. 不需进行点加运算) . 由于 d 位窗口中包含的是带符号数字向量, 不含有相邻的非零位, 所以要计算 d 位窗口中所有可能取值的数目就需要先计算 d- 2, d- 3, , , 3 位窗口中所有可能取值的数目. 按此思路可以推算出窗口大小为 d 时预计算所需的乘法运算量 R(d)

$$R(d) = (d-1)\#D + h(d)\#A \qquad (1)$$

其中 h(d) 为窗口大小为 d 时预计算所需的点加运算的数量, $h(d) = 2@\left(\sum_{i=3}^{d-2} h(i) + d - 2\right)$, 且 h(0) = h(1) = h(2) = 0.

由公式(1) 和 D= 7.2M, A= 17.8M 可计算出表 1.

表 1 采用规范重编码和滑动窗口技术的椭圆曲线标量乘法的预计算开销

d	h(d)	R(d)	所需乘法运算量(M)
2	0	D	7.2
3	2	2D+2A	50
4	4	3D+4A	92.8
5	10	4D+10A	206.8
6	20	5D+20A	392
7	42	6D+40A	755.2
8	84	7D+84A	1545.6
9	170	8D+170A	3083.6

4 采用规范重编码和滑动窗口技术的椭圆曲线标量乘法算法性能分析

在对采用规范重编码和滑动窗口技术的椭圆曲线标量乘法的平均算法性能进行分析时, 首先要对规范重编码进行建模. 假定标量 k 为均匀分布在 $[1, 2^n - 1]$ 内的 n 位二进制数, 那么就可以将 k 看作是一个每次只产生一比特的随机过程的输出结果[4]. 由于此随机过程产生的每一位取 0 或 1 的可能性都是相同的, 并且所产生的任意两位之间都是相互独立的. 所以, $P(k_i = 0) = P(k_i = 1) = 1/2$ 对 $0 \leq i \leq n-1$ 成立. 在此采用有限 Markov 链对标量 k 的规范重编码进行建模, 状态变量为三元组 (k_{i+1}, k_i, C_i)[4]. (k_{i+1}, k_i, C_i) 共有 8 种可能的输入, 所以存在 8 个不同的状态 $(S_0 \sim S_7)$.

表 2 规范重编码的状态转移表

状态		输出	下一状态	
S_i	(k_{i+1}, k_i, C_i)	(D_i, C_{i+1})	$k_{i+2}=0$	$k_{i+2}=1$
S_0	(0, 0, 0)	(0, 0)	S_0	S_4
S_1	(0, 0, 1)	(1, 0)	S_0	S_4
S_2	(0, 1, 0)	(1, 0)	S_0	S_4
S_3	(0, 1, 1)	(0, 1)	S_1	S_5
S_4	(1, 0, 0)	(0, 0)	S_2	S_6
S_5	(1, 0, 1)	$(\bar{1}, 1)$	S_3	S_7
S_6	(1, 1, 0)	$(\bar{1}, 1)$	S_3	S_7
S_7	(1, 1, 1)	(0, 1)	S_3	S_7

表 2 中给出了 S_i 之间的状态转移关系, 其中 k_i 为当前位, k_{i+1} 为要处理的下一位, C_i 和 C_{i+1} 为进位位, D_i 为输出位.

在运用 Markov 链建模的过程中, 还应当考虑滑动窗口法的窗口划分过程. 根据表 2 可知, 当零窗口开始时, Markov 链的当前状态为 S_0、S_3、S_4 或 S_7(输出位 $D_i = 0$); 而当非零窗口开始时, Markov 链的当前状态为 S_1、S_2、S_5 或 S_6(输出位 $D_i \neq 0$). 由于非零窗口的大小必为 d 位, 也即必经过 d 个状态, 所以可将非零窗口中的状态定义为 $S_8, S_9, \ldots, S_{7+d}$. 当非零窗口结束时, 当前状态为 S_{7+d}, 而 S_{7+d} 将要转移到的状态则可能为 S_0、S_3、S_4、S_7(输出 $D_i = 0$ 时)或 S_8(输出 $D_i \neq 0$ 时). 因此考虑了窗口划分的规范重编码的状态转移表即为表 3, 其中 P_{ij} 为从状态 S_i 转移到状态 S_j 的转移概率.

表 3 中的 f_d 为满足生成式(2)的数列:

$$f_k = \frac{(f_{k-1} + f_{k-2})}{2} \tag{2}$$

其初始值 $f_0 = 0, f_1 = 1/4$.

表 3 考虑了窗口划分的规范重编码的状态转移表

S_i	S_j	P_{ij}	S_j	P_{ij}	S_j	P_{ij}	S_j	P_{ij}	S_j	P_{ij}
S_0	S_0	1/2	S_4	1/2						
S_1	S_8	1								
S_2	S_8	1								
S_3	S_1	1/2	S_5	1/2						
S_4	S_2	1/2	S_6	1/2						
S_5	S_8	1								
S_6	S_8	1								
S_7	S_3	1/2	S_7	1/2						
S_8	S_9	1								
S_9	S_{10}	1								
,	,	,								
S_{7+d}	S_0	f_d	S_3	f_d	S_4	f_d	S_7	f_d	S_8	$2f_{d-1}$

根据表 3 和式(2)可得 $d = 4$ 的单步转移概率矩阵 P:

$$P = \begin{bmatrix} 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 5/32 & 0 & 0 & 5/32 & 5/32 & 0 & 0 & 5/32 & 3/8 & 0 & 0 & 0 \end{bmatrix}$$

由于在单步转移概率矩阵 P 中, 状态 S_1、S_2、S_5、S_6 实际上是完全等价的, 可以将其合并为一个状态以简化计算. 原状态与合并简化后的新状态之间的对应关系如下:

$$S_0 \to Sc_0,\ S_3 \to Sc_1,\ S_4 \to Sc_2,\ S_7 \to Sc_3,$$
$$[S_1, S_2, S_5, S_6] \to Sc_4,\ S_8 \to Sc_5,\ S_9 \to Sc_6,\ S_{10} \to Sc_7,\ S_{11} \to Sc_8$$

合并简化以后的单步转移概率矩阵 Pc 为:

$$Pc = \begin{bmatrix} 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 5/32 & 5/32 & 5/32 & 5/32 & 0 & 3/8 & 0 & 0 & 0 \end{bmatrix}$$

设标量 k 的 n 位数字在产生过程中所包含的非零窗口的数目为 C, 则 C 即为椭圆曲线标量乘法所需的点加运算的数量, 而 C 的值等于从状态 Sc_4 到状态 Sc_5 的状态迁移数. 若 $Pc^{(n)}$ 为 n 步转移概率矩阵, 那么 $Pc^{(n)}$ 即为矩阵 Pc 的 n 次幂. 如果将矩阵 $Q^{(n)}$ 定义为

$$Q^{(n)} = \sum_{i=1}^{n} Pc^{(i)} = \sum_{i=1}^{n} Pc^{i} \tag{3}$$

从状态 Sc_4 到状态 Sc_5 的平均迁移数即为

$$C = Q_{4,5}^{(n)} \quad (4)$$

由于预处理所需的乘法运算量为 R(d)(式(1)), 而在椭圆曲线标量乘法运算中所需的倍点运算量为 n- d. 所以, 采用规范重编码和滑动窗口技术的椭圆曲线标量乘法运算所需的(大整数) 乘法运算量平均为

$$T(n, d) = R(d) + (n - d)\#D + (C - 1)\#A \quad (5)$$

给定正整数 $n = \log_2 k$ 和窗口大小 d, 采用上述方法可计算出椭圆曲线标量乘法所需的平均乘法运算量. 表 4 及图 1 给出了在 n 和d 的不同取值下, 椭圆曲线标量乘法所需的平均乘法运算量.

对给定的 n 而言, 不同大小(d) 的滑动窗口所需的平均乘法运算量是不同的(表 4), 所以在计算椭圆曲线标量乘法运算时应当选择最优的滑动窗口大小以保证所需的平均乘法运算量最小. 根据表 4 和图 1, 当 n 为 128、256、512、1024、2048 或 4096 比特时, 滑动窗口的最优大小 d 分别为 4、5、5、6、7、8.

表 4　采用规范重编码和滑动窗口技术的椭圆曲线标量乘法所需的平均乘法运算量

n \ d	2	3	4	5	6	7	8	9
128	155512	137712	135914	141218	155512	187516	262312	413612
256	313514	2726	267216	265418	276116	304614	377612	527114
512	6278	544114	5299	513818	517414	5388	608212	7524
1024	1256312	1087212	10534	1012416	10000	10089	1067614	12047
2048	2515114	21716	21004	2007814	19669	19491	1988216	2107512
4096	503110	4340316	4196118	39986	3898912	3827712	3825914	3913116

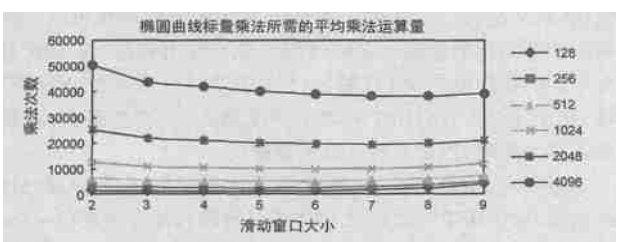

图 1　椭圆曲线标量乘法所需平均乘法运算量的变化趋势

表 5 将采用了规范重编码和滑动窗口技术的椭圆曲线标量乘法所需的乘法运算量与采用 m2ary 法或滑动窗口法的情形进行了比较, 其中 d 的取值都为最优窗口大小, 而 Tc、Td和 T 的取值则是在给定的n 和d 下椭圆曲线标量乘法所需的平均乘法运算量.

本文的分析比较(表 5)表明, 在 128[n[2048 的情况下, 采用规范重编码和滑动窗口技术实现椭圆曲线标量乘法所需的平均乘法运算量要比用 m2ary 法少 1013 2~ 1713 2%, 比单纯采用滑动窗口法也要少 413 5~ 813 4%. 这是由于标量 k 的规范重编码表示中的非零位的数目(平均为(n- 1)/3)比普通的二进制表示(平均为(n- 1)/2) 要少得多, 从而减少了滑动窗口法在窗口划分过程中所产生的非零窗口的数目. 此外, 采用规范重编码技术还可以减少滑动窗口法所需的预计算量. 所以同时采用规范重编码和滑动窗口技术可以显著提高椭圆曲线标量乘法的平均算法性能.

表 5　m2ary 法、滑动窗口法与采用规范重编码的滑动窗口法的算法开销比较

n	m2ary		滑动窗口法		规范重编码与滑动窗口法		(Tc- T)/Tc	(Td- T)/Td
	d	Tc	d	Td	d	T	%	%
128	4	1644.2	4	1484	4	1359.4	17.32	8.40
256	4	3099.8	5	2850.6	5	2654.8	14.36	6.87
512	5	5904.2	5	5459.2	6	5138.8	12.96	5.87
1024	5	11352.8	6	10516.2	6	10000	11.92	4.91
2048	6	21733.8	7	20416.6	7	19491	10.32	4.53

5　结论

在对很大的 k 计算椭圆曲线标量乘法 k# P 时, 经常会采用规范重编码技术和滑动窗口技术来减少所需的点加运算的数量. 本文采用 Markov 链对标量 k 的规范重编码表示的窗口划分过程进行了建模, 利用 n 步转移概率矩阵分析了采用规范重编码和滑动窗口技术的素数域椭圆曲线标量乘法在改进的 Jacobian 坐标系中的平均算法性能. 通过计算、比较给出了在不同的 $n = \log_2 k$ 下滑动窗口的最优窗口大小, 并将采用了规范重编码和滑动窗口技术的椭圆曲线标量乘法的平均算法性能与其它两种实现算法进行了比较, 证明了基于规范重编码和滑动窗口技术的实现算法的运算效率要优于另外两种方法.

参考文献:

[1] Cohen H, Miyaji A, Ono T. Efficient elliptic curve exponentiation using mixed coordinates[A]. Proceeding of 1998 Internation Conference on the Theory and Applications of Cryptology and Information Security[C]. Beijing, China, 1998. 51- 65.

[2] Knuth D. Seminumerical Algorithms, Volume 2 of The Art of Computer Programming[M]. Second edition. Massachusetts, Addison2Wesley Reading, 1981.

[3] J Lopez, R Dahab. Performance of Elliptic Curve Cryptosystems[OB/OL]. Technical Report, IC200208, May 2000. http://www.dcc.unicamp.br/ic2main/publication2e.html.

[4] Egecioglu O, Koc C. Exponentiation using canonical recoding[J]. Theoretical Computer Science, 1994, 129(2): 407- 417.

[5] Koc C. Analysis of sliding window techniques for exponentiation[J]. Computers and Mathematics with Application, 1995, 30(10): 17- 24.

[6] Waser S, Flynn M. Introduction to Arithmetic for Digital System Designers[M]. CBS College Publishing, 1982.

[7] Reitwiesner G. Binary arithmetic[J]. Advances in Computers, 1960, (1): 231- 308.

作者简介:

唐　文　男, 1974 年生于贵州省清镇, 1997 年在四川大学获得学士学位, 2000 年在天津大学获得硕士学位, 2003 年在北京大学获得博士学位, 现为北京大学信息科学技术学院讲师, 主要研究方向为网络信息安全. Email: tangwen@infosec. pku. edu. cn.