椭圆曲线密码体制中点的数乘的一种快速算法 [1]

郝 林 罗 平*

(云南大学计算机科学与工程系 昆明 650091)
*(清华大学计算机科学与技术系 北京 100084)

摘 要 该文基于椭圆曲线密码体制，提出了椭圆曲线上点的数乘的一种快速算法．该算法通过引入 2^k 进制序列，缩短了乘数的相应序列长度，从而大大减少了点的数乘中的加法运算次数，并且分析了 k 的最佳选取，使得我们提出的算法比通常点的数乘算法效率提高了 60% 以上．

关键词 椭圆曲线，快速算法，密码学
中图号 TN918.1

1 引 言

自 1985 年椭圆曲线密码学出现以来，它的理论和应用发展十分迅速，现已成为公钥密码学中最为活跃的一个分支． Neal Koblitz[1] 和 Victor Miller[2] 利用椭圆曲线上点形成的 Abel 加法群构造了椭圆曲线上的离散对数问题，简称为椭圆曲线离散对数问题 (ECDLP)，提出了著名的椭圆曲线加密算法．从现有的攻击算法表明 ECDLP 的难度远远超过了乘法群上的离散对数问题．实验表明，在椭圆曲线加密算法中采用 160bit 的密钥可与 1024bit 密钥的 RSA 算法的安全性相当 [3]，且随着模数的增大，它们之间安全性的差距猛烈增大．由于椭圆曲线加密算法密钥短，并且所基于的有限域的运算位数远少于传统离散对数的运算位数，其加法运算很容易在计算机的硬、软件上实现，所以，它的应用前景非常广阔，例如可用于 16 位的 IC 卡上．然而，椭圆曲线密码算法速度仍很慢，不能满足人们的某些要求，从而或多或少影响着实际的应用．所以加强对算法的研究，提高其运算速度就成为一个重要的问题．

在基本的二进制序列中，椭圆曲线上点的数乘的运算速度的关键取决于大数 S 的二进制序列的长度及序列中 0 的个数．如果能够找到 S 的另外表示，使其序列长度缩短或序列中 0 的个数增加，则算法的运算速度有望大大加快．基于这种考虑，本文引入大数 S 的 2^k 进制序列，由此构造了一种椭圆曲线上点的数乘的快速算法，并讨论了 k 的最佳选取，给出了若干实例进行比较说明．

2 点的数乘的基本算法

在给出基本算法之前，先定义椭圆曲线 E 上的运算．为简单起见，我们设 $K = F_{2^m}$ 表示一个有限域，考虑该域 K 上的非超奇异椭圆曲线 E： $y^2 + xy = x^3 + a_4x^2 + a_6$，其中 $a, y, a_i \in K$， $(i = 4, 6)$．定义 E 上的加法运算如下：设曲线 E 上的两个点 P， $Q \neq O_k(O_k$ 为无穷远点)， $P = (x_1, y_1)$， $Q = (x_2, y_2)$， P 的逆元 $-p = (x_1, y_1 + x_1)$．

若 $Q \neq -P$， $Q \neq P$，则 $P + Q = (x_3, y_3)$，其中 $x_3 = \lambda^2 + \lambda + x_1 + x_2 + a_4$，$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$， $\lambda = (y_2 + y_1)/(x_2 + x_1)$．

若 $Q = P = (x_1, y_1)$，则 $P+Q = 2P = (x_3, y_3)$，其中 $x_3 = \lambda^2+\lambda+a_4$，$y_3 = x_1^2+(\lambda+1)x_3$，$\lambda = x_1 + (y_1/x_1)$．特别地，对 $\forall P \in E$，若 $Q = O_k$，则 $P+Q = P$．对实数 0，有 $0P = O_k$．

由上述定义可知， E 上两互异元素 (均不为 O_k) 相加，可通过域 F_{2^m} 上元素的 3 次乘法，1 次求逆和 9 次加法得到 (由于域 F_{2^m} 上元素的加法运算耗时甚少，通常将其略去，不在点的数乘运算效率中考虑)． E 上两相同元素 (不为 O_k) 相加，也称为倍点运算，可通过 F_{2^m} 上元素的 4 次乘法， 1 次求逆和 5 次加法得到．不难验证， $< E \bigcup \{O_k\}, + >$ 构成一 Abel 群．设

[1] 2001-05-28 收到， 2002-01-18 定稿
国家 973 项目 (项目编号 G1998030420)

$P \in E$ 为曲线 E 上的一个 t 阶点， S 为任意正整数， $0 \le S < t$. 那么，由椭圆曲线密钥交换协议可知. 要提高加解密速度，必须提高点的数乘 SP 的效率，为此我们先讨论它的一般情况. 显然， S 可唯一表示成 [4]

$$S = a_{n-1}m^{n-1} + a_{n-2}m^{n-2} + \cdots + a_1 m + a_0 \tag{1}$$

其中 m 是大于 1 的正整数， $a_j (j = 0, 1, \cdots, n-1)$ 是整数，满足 $0 \le a_j < m$.

当 $m = 2$ 时， (1) 式成为 $S = a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + \cdots + a_1 2^1 + a_0 2^0$ ， $0 \le a_j < 2$ ， $j = 0, 1, \cdots, n-1$. 因此，我们有

$$\begin{aligned} SP &= (a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + \cdots + a_1 2^1 + a_0 2^0)P \\ &= a_{n-1}2^{n-1}P + a_{n-2}2^{n-2}P + \cdots + a_1 2P + a_0 P \\ &= 2(2 \cdots (2(O_k + a_{n-1}P) + a_{n-2}P) \cdots + a_1 P) + a_0 P \end{aligned} \tag{2}$$

由 E 上倍点运算的定义可知，要计算 $2^i p$ ，只需计算： $2P$ ， $2^2 P = 2(2P), \cdots, 2^i P = 2(2(\cdots 2P \cdots))$. 共需 i 次倍点运算. 且

$$a_i 2^i p = \begin{cases} 2^i P, & a_i = 1, \\ O_k, & a_i = 0, \end{cases} \qquad i = 0, 1, \cdots, n-1$$

从而，由 (2) 式我们得到基本的点的数乘算法：

算法 1

```
begin
  T = O_k ;
  for i = n − 1 to 0 by −1
    begin
      if a[i] ≠ 0 then T = 2(T + P)
      else T = 2T
  end
end
```

由算法 1 ，计算点的数乘 SP ，需要 $n-1$ 次倍点运算及至多 $n-1$ 次加法运算，即需要域 F_{2^m} 上的 $7(n-1)$ 次乘法及 $2(n-1)$ 次求逆.

3 快速算法

由上面基本算法的讨论可知， SP 的运算速度取决于 n 的大小. 若选取适当正整数 $k > 1$ ，使得 $n = kN$ ，其中 N 为某一正整数，将 2^k 取代 2 构成整数 S 的基，这样 (2) 式的长度将大大缩短，从而减少了加法运算次数. 因此，我们在 (1) 式中取 $m = 2^k (k > 1)$ ，则

$S = a_{N-1}(2^k)^{N-1} + a_{N-2}(2^k)^{N-2} + \cdots + a_1 2^k + a_0$ ，其中 $0 \le a_i < 2^k$ ， $i = 0, 1, \cdots, N-1$.

从而得

$$\begin{aligned} SP &= a_{N-1}(2^k)^{N-1} + a_{N-2}(2^k)^{n-2}P + \cdots + a_1 2^k P + a_0 P \\ &= 2^k(2^k(\cdots(2^k a_{N-1}P + a_{N-2}P)\cdots) + a_1 P) + a_0 P \end{aligned} \tag{3}$$

由于 $a_i \in \{0, 1, \cdots 2^k - 1\}$ ，所以我们需计算 $2P, 3P, \cdots, (2^k - 1)P$. 由 E 上的加法运算定义可得 $2P = P + P$, $3P = 2P + P, \cdots, (2^k - 1)P = (2^k - 2)P + P$. 共需 $(2^k - 2)$ 次加法运算. 因此，我们能够构造如下点的数乘 SP 的快速算法：

算法 2

Step 1:

begin

$T[0] = O_k$;

for $i = 1$ to $2^k - 1$

$T[i] = T[i-1] + P$

end

Step 2:

begin

$T = O_k$;

for $i = N - 1$ to 0 by -1

begin

for $h = 1$ to k

$T = 2T$;

for $j = 1$ to $2^k - 1$

if $j = a[i]$ then $T = T + T[j]$

end

end

从算法 2 不难看出， Step 1 需要 $2^k - 1$ 次点的加法运算，即需要在域 F_{2^m} 上做 $3(2^k - 1)$ 次乘法及 $2^k - 1$ 次求逆运算． Step 2 需要 N 次点的加法运算及 $k \times N$ 次倍点运算 (其中 $a[i]$ 为 (3) 式中的 a_i) ，即需要在域 F_{2^m} 上做 $(3 + 4k) \times N$ 次乘法及 $(k + 1) \times N$ 次求逆运算．

4 算法讨论与实例

从算法 2 可以看到，为了加快曲线上点的数乘运算，提高效率，除了对整数 S 采用以 2^k 为基的表达外，还采用了查表技术，即在 Step 1 存储 $T(0), T(1), \cdots, T(2^k - 1)$ ．当 k 不大时，显然这种代价是微不足道的．通过对算法 1 和算法 2 的分析，我们便得到两种算法的运算步数如表 1 ：

由表 1 可知，两种算法中倍点运算次数相差甚小，当 n 很大时，二者几乎相等，区别仅在于加法运算上．以下要讨论的是在给定 S(即给定 S 的二进制序列长度 n) 后， k 取何值时，算法 2 的效率可达到最佳？即什么样的 k ，使得加法运算次数 $2^k - 1 + n/k$ 达到最小．为此，我们设 $f(x) = 2^x - 1 + n/x$ ，对 $f(x)$ 求导，则得 $f'(x) = 2^x \ln 2 - n/x^2$ ，令 $f'(x) = 0$ ．由极值理论 [5] 得到 k 的最佳选取与 S 的二进制序列长度 n 必须满足的关系式： $n = 2^k k^2 \ln 2$ ．当 $k = 3, 4, 5, 6$ 时， n 分别取值为 50 ， 177 ， 555 ， 1597 ．即当 $40 \le n \le 2000$ 时， k 可以试取值 3 ， 4 ， 5 ， 6 ，使算法效率达到最佳．

从表 2 可以看出，所得结果与理论分析一致，我们提出的算法 2 比基本的算法 1 平均效率提高了 60% 以上．

下面给出实例，随机地选取 6 个大整数进行计算，并将结果列于表 2 ．

表 1　两种算法的运算步数

	加法运算 (次)	倍点运算 (次)
算法 1	n-1	n-1
算法 2	$2^k - 1 + n/k$	n

表 2 大整数的计算

序号	S 的十进制序列长度	S 的二进制序列长度	k 值	S 的 2^k 进制序列长度	算法 1 加法运算次数	算法 2 加法运算次数	运算次数减少量
1	23	76	4	19	75	34	55%
2	49	160	4	40	159	55	65%
3	106	352	5	71	351	102	71%
4	120	394	5	79	393	110	72%
5	154	512	5	103	511	134	74%
6	180	594	5	119	593	150	75%

参 考 文 献

[1] N. Kobliz, Elliptic curve cryptosystem, Mathematics of Computation, 1987, 48(177), 203–209.

[2] V. Miller, Uses of elliptic curve in cryptography, Advances in Cryptology-CRYPTO'85, LNCS, 218, Berlin, Springer-Verlag, 1986, 417–426.

[3] N. Demytko, A new elliptic curve based analogue of RSA, Advances in Cryptology—EUROCRYPT'93 Proceedings, Springer-verlag, 1994, 40–49.

[4] 卢开澄, 计算机密码学 (第二版), 北京, 清华大学出版社, 1998, 241–243.

[5] 四川大学数学系高等数学教研室, 高等数学 (第一册), 北京, 高等教育出版社, 1995, 170–172.

A FAST ALGORITHM FOR THE POINT MULTIPLICATION IN ELLIPTIC CURVE CRYPTOSYSTEMS

Hao Lin Luo Ping*

(*Dept. of Computer Science and Engineering, Yunnan University, Kunmin* 650091, *China*)

*(*Dept. of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*)

Abstract In this paper, a n w fast algorithm for the numerical multiplication of the points on elliptic curves is presented. By introducing 2^k sequence representation for number, the length of numerical multiplication is s ortened, so that the number of addition operation on elliptic curves is decreased greatly. Mor over, the optimal choice of k is analyzed and the efficiency of the algorithm presented is impro ed about 60

Key words Elliptic curve, Fast lgorithm, Cryptography

郝 林: 男, 1955 年生, 副教授, 主要从事密码学方面的研究.

罗 平: 男, 1960 年生, 博士, 副教授, 主要从事计算数学、密码学、网络安全等方面的研究.