j不变量等于1728的GLS椭圆曲线上四维GLV方法*

宋承根,徐茂智,周正华
(北京大学 数学科学学院,北京 100871)

摘 要:为了实现椭圆曲线的快速倍乘,Gallant-Lamber-Vanstone(GLV)方法被推广到四维的一般情形。文章中回答了Galbraith,Lin和Scott(J. Cryptol. DOI: 10.1007/s00145-010-9065-y)提出的一个公开问题:研究$\mathbb{F}_{p^2}$上j不变量等于1728的GLS椭圆曲线上的四维GLV方法,并给出时间周期。尤其指出GLV的四维分解能够在很大的概率上实现,给出了一些结果和例子。特别指出在同一类曲线上,四维GLV方法的时间周期大概是二维GLV方法的70%~73%。

关键词:椭圆曲线; 点的倍乘;GLV方法
中图分类号:TN918.1 **文献标志码**:A **文章编号**:1001-2486(2012)02-0025-04

4-dimensional GLV method on GLS elliptic curves with j-invariant 1728

SONG Chenggen, XU Maozhi, ZHOU Zhenghua
(School of Mathematical Sciences, Peking University, Beijing 100871, China)

Abstract: In order to obtain a fast multiplication on elliptic curves, the Gallant-Lambert-Vanstone(GLV) method is introduced to the general situation in dimension 4, one of the open problems in Galbraith, Lin and Scott's work (J. Cryptol. DOI: 10.1007/s00145-010-9065-y) is answered, that is, studying the performance of 4-dimensional GLV method for faster point multiplication on some GLS curves over $\mathbb{F}_{p^2}$ with j-invariant 1728. Finally some results and examples are presented, showing that the 4-dimensional GLV method runs in between 70% and 73% the time of the 2-dimensional GLV method which Galbraith et al. did in their work.

Key words: elliptic curve; point multiplication; GLV method

令E是域$\mathbb{F}_q$上椭圆曲线,并假设$P\in E(\mathbb{F}_q)$的阶r为偶数。一个研究热点就是如何在椭圆曲线上快速计算点的倍乘$[k]P$。

2001年,Gallant等[1]提出了一种新的倍乘加速方法。如果E有一个有效可计算的自同构ψ满足$\psi(P)=\lambda P\in <P>$,那么我们可以用$[k_1]P+[k_2]\psi(P)$来替代计算$[k]P$,这里$|k_1|,|k_2|\approx\sqrt{r}$。

2002年,Park等第一次给出了$|k_1|,|k_2|$的边界[2]。随后Sica等给出了一个更细致的边界[3]。遗憾的是,他们的工作都仅仅局限于二维的情形。

实际上,Iijima等[4]针对$E(\mathbb{F}_{p^2})$上的椭圆曲线构造了一类自同构。在2009年,Galbraith等[5]把他们的结果应用到了GLV方法。

2010年,Zhou等[6]针对一类特殊的GLS曲线利用LLL算法得到三维GLV方法的一组基,并且得到很好的加速效果。

Galbraith等的一个公开问题[5]是:研究$\mathbb{F}_{p^2}$上j-不变量等于1728的GLS椭圆曲线上的四维GLV方法,并给出时间周期。我们回答了这个问题,并利用LLL算法得到一组四维GLV方法的基。同时我们指出GLV的四维分解能够在很大的概率上实现,给出了一些结果和例子。我们指出在同一类曲线上,四维GLV方法的时间周期大概是二维GLV方法的70%~73%。

1 快速倍乘方法

我们总是考虑定义在$\mathbb{F}_q$上的椭圆曲线E。P是E上一个有素数阶的点。

1.1 GLV方法

Gallant等[1]提出只要椭圆曲线E存在一个有效可计算的自同构Ψ满足$\Psi(P)=\lambda P\in <P>$,那么计算$[k]P$会变得容易,其中k是$[1,r-1]$间的一个随机整数。

首先,定义向量$\Lambda=(1,\lambda)$,以及格$L=\{k_0,k_1\in\mathbb{Z}^2:(k_0,k_1)\cdot\Lambda\equiv 0 \bmod r\}$。

* **收稿日期**:2011-07-28
基金项目:国家自然科学基金资助项目(10990011)
作者简介:宋承根(1987—)男,贵州锦屏人,博士研究生,E-mail:cgsong@pku.edu.cn;
徐茂智(通信作者),男,教授,博士,博士生导师,E-mail:mzxu@pku.edu.cn

然后,选择格 L 的一组基$\{v_0,v_1\}$。令 $s_0,s_1\in\mathbf{Z}$,$v=s_0\cdot v_0+s_1\cdot v_1\in L$,$(k,0)=m_0\cdot v_0+m_1\cdot v_1$,其中 $m_0,m_1\in\mathbf{Q}$。那么

$$e=(k,0)-v=(m_0-s_0)v_0+(m_1-s_1)v_1$$

只要选择合适的数值使得$|m_i-s_i|\leqslant 1/2$,$i=0,1$,那么由三角不等式,可得

$$\|e\|\leqslant 1/2(\|v_0\|+\|v_1\|)$$

以上的推论说明只要选择合适的向量$\{v_0,v_1\}$,那 e 就可能很短。通常都是利用一些格基约化算法来得到这组向量(例如 LLL 算法[7])。GLV[5]也给出了一个计算优化格基的方法。随后不少工作都推进了这个结果[2-3]。

1.2 本文的方法

Galbraith,Lin 和 Scott 在文献[5]中实现了$\mathbb{F}_{p_2}$椭圆曲线上的二维 GLV 方法。

令 $p>3$ 是一个素数,E 是$\mathbb{F}_p$ 上的椭圆曲线,且有$\#E(\mathbb{F}_p)=p+1-t$。

令 π 是 E 上 p 阶 Frobenius 映射。令 $E'(\mathbb{F}_{p^2}):y^2=x^3+a'x+b'$是 $E(\mathbb{F}_{p^2})$的纽曲线,那么$\#E'(\mathbb{F}_{q^2})=(p-1)^2+t^2$。用 $O_{E'}$表示 E'的单位元。

定理 1[5] 假设存在定义在$\mathbb{F}_{p^{2d}}$上的纽同态 $\phi:E'\to E$。定义 $\psi=\phi^{-1}\pi\phi$。那么 $\psi\in End_{\mathbf{F}_{q^d}}(E')$。令 $r|\#E'(\mathbb{F}_{q^d})|$是一个素数并满足 $r>2p^{d-1}$,那么对于 $P\in E'(\mathbb{F}_{q^d})[r]$,我们有 $\psi^d(P)+P=O_{E'}$。

推论 1 令 $p\equiv 1 \bmod 6$,并令 $A\in\mathbb{F}_p^*$。定义 $E:y^2=x^3+Ax$。选择 $u\in\mathbb{F}_{p^8}^*$满足 $u^4\in\mathbb{F}_{p^2}$并且定义$\mathbb{F}_{p^2}$上椭圆曲线 $E':y^2=x^3+u^4Ax$。那么同态 $\phi:E\to E'$由 $\phi(x,y)=(u^2x,u^3y)$给出,它是定义在$\mathbb{F}_{p^8}$上的。令 $\psi=\phi\pi\phi^{-1}$,那么对于 $P\in E'(\mathbb{F}_{p^2})$,我们有 $\psi^4(P)+P=O_{E'}$。

令 E',ψ 如推论 1 中定义,并且假设 $r|\#E'(\mathbb{F}_{p^2})|$是一个素数。在实现四维 GLV 方法过程中最困难的就是如何把 k 分解成 4 个 $O(r^{1/4})$大小的数。

令 $P\in E'(\mathbb{F}_{p^2})$是一个阶为 r 的点,并假设 $\psi(P)=[\lambda]P$ 满足 $\lambda\in\mathbb{Z}/r\mathbb{Z}$。定义向量 $\boldsymbol{\psi}=(1,\psi,\psi^2,\psi^3)$,$\boldsymbol{\Lambda}=(1,\lambda,\lambda^2,\lambda^3)$,以及格 $L=\{(k_0,k_1,k_2,k_3)\in\mathbb{Z}^4:(k_0,k_1,k_2,k_3)\cdot\Lambda\equiv 0 \bmod r\}$。

令$\{v_0,v_1,v_2,v_3\}$是格 L'的一组基,其中 $0\subset L'\subseteq L$。可以分解$(k,0,0,0)=\beta_0v_0+\beta_1v_1+\beta_2v_2+\beta_3v_3$,其中 $\beta_i\in\mathbb{Q}$,$i=0,\cdots 3$。用 $b_i=[\beta_i]$表示最接近 β_i 的整数。那么 k 可以分解成

$$\begin{aligned}&(k_0,k_1,k_2,k_3)\\&=(k,0,0,0)-(b_0v_0+b_1v_1+b_2v_2+b_3v_3)\\&=(\beta_0-b_0)v_0+(\beta_1-b_1)v_1+(\beta_2-b_2)v_2+(\beta_3-b_3)v_3.\end{aligned}$$

既然$|\beta_i-b_i|\leqslant 1/2$,由三角不等式得

$$\max_i\{|k_i|\}\leqslant\|(k_0,k_1,k_2,k_3)\|\leqslant 2\max_i\{\|v_i\|\}.$$

在四维 GLV 方法中,可用 NAFs 来加速点的倍乘[8]。接下来的文中,$(u_{d-1},\cdots,u_1,u_0)NAF_\omega$表示整数 u 的 NAF_ω 表示,ω 表示宽度。

算法 1 四维方法中使用 *NAFs* 方法。

输入:$w,u=(u_{d-1},\cdots,u_1,u_0)_{NAFw}$,$v=(v_{d-1},\cdots,v_1,v_0)_{NAFw}$,$x=(x_{d-1},\cdots,x_1,x_0)_{NAFw}$,$y=(y_{d-1},\cdots,y_1,y)_{NAFw}$,$P,Q,R,T$。

输出:$uP+vQ+xR+yT$。

(1)对所有的 $i\in\{3,5,\cdots,2^{w-1}-1\}$,计算 iP,iQ,iR,iT。

(2) $S\leftarrow O$。

(3)从 $d-1$ 到 0 计算 i:

1) $S\leftarrow 2S$;

2) $2S\leftarrow S+(u_iP+v_iQ+x_iR+y_iT)$。

(4)返回 S。

1.3 我们方法的分析

令 r 是一个素数,$\mathbb{F}_r=GF(r)$,$u=(1,\lambda,\lambda^2,\lambda^3)^{\mathrm{T}}$。$L$ 如上文中定义是一个格,$<\cdot,\cdot>$表示两个向量的内积。

主要的目的就是找出格 L 的一组有合适长度的格基。考虑下面这个问题:

定理 2 如上定义的 u,如果存在一组线性无关的向量 $x_0,x_1,x_2,x_3\in\mathbf{Z}^4$ 满足 $\|x_i\|=O(\sqrt[4]{r})$,$<x_i,u>=0 \bmod r$,$i=0,1,2,3$,那么就能在确定多项式时间内找到 4 个线性无关的向量 $b_0,b_1,b_2,b_3\in L$ 满足 $\|b_i\|=O(\sqrt[4]{r})$,$<b_i,u>=0 \bmod r$,$i=0,1,2,3$。

证明 这是文献[6]中定理 3 的一个简单推广。

附注:既然 L 的格基存在,可利用 LLL 算法得到一组输出向量 b_0,b_1,b_2,b_3。

2 生成算法及例子

2.1 生成算法

对于在我们关注的椭圆曲线上面点的个数,有

定理 3[9] 令 $p\neq 2$ 是一个素数,并且 $p\nmid D$。考虑椭圆曲线 $E:y^2=x^3-Dx$ on $\mathbb{F}_1$。如果 $p\equiv 3 \bmod 4$,那么有$\#E(\mathbb{F}_p)=p+1$。如果 $p\equiv 1 \bmod 4$,令 $p=\pi\bar{\pi}$ 其中 $\pi\in\mathbb{Z}[i]$和 $\pi\equiv 1 \bmod (2+2i)$,那

么有

$$\#E(\mathbb{F}_p) = p + 1 - \left(\frac{\bar{D}}{\pi}\right)_4 \pi + \left(\frac{D}{\pi}\right)_4 \bar{\pi}$$

利用下面这个算法来生成需要的椭圆曲线以及相关参数。这个算法是 Galbraith 等生成算法[5]的一个推广。

算法 2　椭圆曲线生成。

输出: p, E, r, w

(1) 重复

选择一个素数 $p \equiv 5 \bmod 8$;

令 $u_0 = \sqrt{-2} \in \mathbb{F}_{p^2}$;

随机选择 $A \in \mathbb{F}_p$ 并令 $E: y^2 = x^3 + A * u_0 x$;

计算$\#E(\mathbb{F}_{p^2})$;

直到 $r = \#E$ 或者 $r = (\#E/2)$是素数。

(2) 计算 $w = (-2)^{(-3-p)/8} u_0$。

(3) 返回 p, E, r, w。

令 $u \in \mathbb{F}_{p^8}$满足 $u^4 = u_0$,那么在仿射坐标下 E' 的自同构可以写成 $\psi: E' \longrightarrow E': (x, y) \longmapsto (u^{2-2p} x^p, u^{3-3p} y^p)$。令 $w = u^{1-p}$,那么 $\psi(x, y) = (w^2 x^p, w^3 y^p)$。

2.2　例子

在本小节中简单描述如何实现四维 GLV 方法。

选择的例子为一个 127 - bit 伪梅生素数:

$$p = 0x7fffffffffffffffffffffffffffffbc25,$$

满足 $p \equiv 5 \bmod 8$。椭圆曲线方程如下给出:

$$E'(\mathbb{F}_{p^2}): y^2 = x^3 - 3u_0 x;\ u_0 = u^4$$

其中 $u \in \mathbb{F}_{p^8}$的极小多项式是 $x^8 + 2$。$r = \#E'(\mathbb{F}_{p^2})$ 是一个 255bit 长素数

$$r = 0x1fffffffffffffffffffffffffffffde$$
$$127da30fc946b49a6b476a4691e8017009$$

仿射坐标下 E'的一个自同构可以表示成:

$$\psi(x, y) = (w^2 x^p, w^3 y^p) = [\lambda](x, y)$$

其中

$$w = 0x43948367c3a87981362eb117c8d11c5 \times u_0$$
$$\lambda = 0x1f8916ad355a0de12c288bc1abcb$$
$$4efe836bb99a49d427a6a7af6f96ed4eedd8$$

利用 LLL 算法和 Babai Rounding 方法可完成四维 GLV 方法的分解。

为了比较,用 $\psi_1: E' \longrightarrow E', (x, y) \longmapsto (-x, iy)$ $i^2 \equiv -1 \bmod p$,来实现二维 GLV 方法。这是来自 Galbraith 等的技术[5]。

参考 Galbraith 等[5]的工作,使用 MIRACL Library 来实现代码。测试环境是 64-bit Linux system + Intel Core i5 750 processor / 2.66GHz。用系统时间 TSC(timestamp counter)来计算时间周期,实际中省略了时间周期的后 3 位。

2.3　点的倍乘

利用 INT 方法[8]来实现 m 维 GLV 方法中的点的倍乘,$m = 2,4$。为了计算 $[k]P = \sum_i [k_i]\psi^i(P)$,其中 P 是一个动点,需要确定每个 k_i 的 NAF 表示的宽度 w,其中 $\log_2 | k_i | \approx \log_2 | k | / m$,并计算 $P_j = [j]P$ 对于 $j = \{1,3,5, \cdots, 2^{w-1} - 1\}$。令 A 和 D 分别表示 $E'(\mathbb{F}_{p^2})$ 中点的加法以 2 倍乘的消耗。令 H 表示操作 ψ 需要的消耗。那么计算 $[k]P$ 期望的消耗可以近似表示成:

$$1D + (2^{w-2} - 1)A + (m-1) \cdot 2^{w-2} H + \log_2 |k| \left(\frac{1}{w+1} A + \frac{1}{m} D\right)$$

表 1 给出了计算倍乘的时间周期。用 mGLV + INT 表示滑动窗口宽度为 5 的 NAFs 方法和 m 维 GLV 方法。在实现中,平均了 10^5 次点的倍乘的数据。

表 1　点的倍乘时间比较——127 - 比特 p

Tab. 1　Point multiplication—127-bit p

Method	$\mathbb{F}_p$ muls	$\mathbb{F}_p$ adds/subs	Clock cycles
2GLV + INT	4188	12 965	2 671 000
4GLV + INT	3069	8463	1 863 000

2.4　签名认证

在一些签名验证算法中(比如 ECDSA 和 Schnorr 签名)需要计算 $[k]P + [n]Q$,其中 P 是一个固定点。

既然 P 是固定点,可以预先计算关于 P 的一些变量。在计算固定点 P 时,用宽度为 6 的滑动窗口 NAFs 方法,同时用宽度为 5 滑动窗口 NAFs 方法来计算动点 Q。

在表 2 中针对一些曲线比较了四维 GLV 方法和二维 GLV 方法。如上所示,依然用 mGLV + INT 表示 m 维的倍乘方法。

表 2　签名验证时间比较——127 - 比特 p

Tab. 2　Signature verification—127-bit p

Method	$\mathbb{F}_p$ muls	$\mathbb{F}_p$ adds/subs	Clock cycles
2GLV + INT	5501	16 333	3 425 000
4GLV + INT	4230	11 408	2 528 000

3　结　论

研究了一类 j 不变量维 1728 的 GLS 曲线上四维 GLV 方法。具体例子实现表面四维 GLV 方

法所需要的时间是 Galbraith 等[5]提出的二维 GLV 方法的 0.70 和 0.73。

参考文献(References)

[1] Gallant R P, Lambert R J, Vanstone S A. Faster point multiplication on elliptic curves with efficient endomorphisms, [C]// Proc of CRYPTO 2001, LNCS 2139, Springer, Heidelberg, 2001:190 – 200.

[2] Park Y H, Jeong S, Kim C H, et al. An alternate decomposition of an integer for faster ponit multiplication on certain elliptic curves[C]// Proc of PKC 2002, LNCS 2274, Springer, Heidelberg, 2001:323 – 334.

[3] Sica F, Ciet M, Quisquater J J. Analysis of gallant-lambert-vanstone method based on efficient endomophisms: elliptic and hyperelliptic curves. [C]// Proc of SAC 2002, LNCS 2595, Springer, Heidelberg, 2003:21 – 36.

[4] Iijma T, Matsuo K, Chao J, et al. Construction of frobenius maps of twist elliptic curves and its application to elliptic scalar multiplication. [C]// Proc of SCIS 2002, IEICE, Japan, 2002:699 – 702.

[5] Galbraith S D, Lin X B, Scott M. Endomorphisms for faster elliptic curve cryptogrpahy on a large class of curves. [C]// Proc of EUROCRYPT 2009, LNCS 5479, Springer, Heidelberg, 2009:518 – 535.

[6] Zhou Z H, Hu Z, Xu M Z, et al. Efficient 3-dimensional GLV method for faster point multiplication on some GLS elliptic curves[J]. Information Processing Letters, 2010, 110: 1003 – 1006

[7] Cohen H. A course in computational algebraic number theory[M]. Springer-Verlag, 1996.

[8] Hankerson D, Menezes A J, Vanstone S. Guide to elliptic curve cryptography[M]. Springer, Heidelberg, 2004.

[9] Ireland K, Rosen M. A classical introduction to modern number theory [M]. 2nd ed. GTM, Springer, New York, 1990.

[10] Galbraith S D, Lin X B, Scott M. Endomorphisms for faster elliptic curve cryptogrpahy on a large class of curves[J]. J. Cryptol, 2010.

(上接第 20 页)

[3] Li N, Qi W. Construction and analysis of Boolean functions of 2t + 1 variables with maximum algebraic immunity[C] // Proc of Advances in Cryptology-ASIACRYPT 2006, LNCS 4284: 84 – 98.

[4] Carlet C, Feng K. An infinite class of balanced functions with optimal Algebraic immunity, good immunity to fast algebraic attacks and good nonlinearity [C] // Proc of Advances in Cryptology-ASIACRYPT 2008, LNCS 5350: 425 – 440.

[5] Qu L, Feng K, Liu F, et al.. Constructing symmetric Boolean functions with maximum algebraic immunity [J]. IEEE Transactions on Information Theory, 2009, 55(5): 2406 – 2412.

[6] Wang Q, Peng J, Kan H, et al. Constructions of cryptographically significant Boolean functions using primitive polynomials [J]. IEEE Transactions on Information Theory, 2010, 56(6): 3048 – 3053.

[7] Tu Z R, Deng Y P. A conjecture on binary string and its applications on constructing Boolean functions of optimal algebraic Immunity [EB/OL]. [2011 – 05 – 12]. Cryptology ePrint Archive, Report 2009/272, http://eprint.iacr.org.

[8] Tu Z R, Deng Y P. A conjecture about binary strings and its applications on constructing Boolean functions with optimal algebraic immunity [J]. Designs, Codes and Cryptography, 2011, 60(1): 1 – 14.

[9] Tu Z R, Deng Y P. Boolean functions with all main cryptographic properties [EB/OL]. [2011 – 06 – 05]. Cryptology ePrint Archive, Report 2010/518, http://eprint.iacr.org.

[10] Zeng X, Carlet C, Shan J, et al. More balanced Boolean functions with optimal algebraic immunity and good nonlinearity and resistance to fast algebraic attacks [J]. IEEE Transactions on Information Theory, 2011, 57(9): 6310 – 6320.

[11] Dillon J F. Elementary hadamard difference sets [D]. Baltimore University of Maryland, 1974.

[12] Carlet C, Dalai D K, Gupta K C, et al.. Algebraic immunity for cryptographically significant Boolean functions: analysis and construction [J]. IEEE Transactions on Information Theory, 2006, 52(7): 3105 – 3121.