软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(10):2275−2288 [doi: 10.3724/SP.J.1001.2013.04370]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

超椭圆曲线上 Montgomery 标量乘的快速计算公式*

李 明[1,3], 孔凡玉[2], 朱大铭[1]

[1](山东大学 计算机科学与技术学院,山东 济南 250100)

[2](山东大学 网络信息安全研究所,山东 济南 250100)

[3](国网山东省电力公司,山东 济南 250100)

通讯作者: 李明, E-mail: luaming@msn.com

摘 要: 超椭圆曲线密码体制与椭圆曲线密码体制相比,具有安全性高、密钥短的特点.标量乘计算是这两个密码体制中最为核心和重要的计算,其中,Montgomery 阶梯算法是计算标量乘的一种重要算法,且因为其可以抵抗简单的边带信道攻击,而被广泛研究和应用.近几年,椭圆曲线上的 Montgomery 阶梯算法和相应的点运算公式一直在不断改进,但是在超椭圆曲线上,直接设计快速运算公式来提高 Montgomery 阶梯算法的速度,却一直没有太大的进展. Lange 曾经探讨过这种快速公式存在的可能性,但却并没有得到一个实用、有效的计算公式.在特征为 2 的域上,通过改进超椭圆曲线上的除子类加法公式来提高超椭圆曲线上的 Montgomery 阶梯标量乘计算,提出了一种新的思路来改进多种坐标系下的加法公式.分析和仿真结果表明,在特征为 2 的域上,新的运算公式的运行速度比之前的标准公式均有所提高.在某类常用曲线上,新的公式比之前的公式快了 4%~8.3%.这说明,直接设计快速除子运算公式来提高 Montgomery 阶梯算法的速度是可行的.同时,使用新的公式实现的 Montgomery 阶梯算法可以抵抗简单边带信道攻击.

关键词: 超椭圆曲线;标量乘;Montgomery Ladder;运算公式;除子类

中图法分类号: TP301 **文献标识码**: A

中文引用格式: 李明,孔凡玉,朱大铭.超椭圆曲线上 Montgomery 标量乘的快速计算公式.软件学报,2013,24(10):2275−2288. http://www.jos.org.cn/1000-9825/4370.htm

英文引用格式: Li M, Kong FY, Zhu DM. Fast addition formulae for Montgomery Ladder scalar multiplication on hyperelliptic curves. Ruan Jian Xue Bao/Journal of Software, 2013,24(10):2275−2288 (in Chinese). http://www.jos.org.cn/1000-9825/4370.htm

Fast Addition Formulae for Montgomery Ladder Scalar Multiplication on Hyperelliptic Curves

LI Ming[1,3], KONG Fan-Yu[2], ZHU Da-Ming[1]

[1](School of Computer Science and Technology, Shandong University, Ji'nan 250100, China)

[2](Institute of Network Security, Shandong University, Ji'nan 250100, China)

[3](State Grid Shandong Electric Power Company, Ji'nan 250100, China)

Corresponding author: LI Ming, E-mail: luaming@msn.com

Abstract: Comparing with elliptic curve (EC) cryptosystem, hyperelliptic curve (HEC) cryptosystem offers high level of security with shorter key size. Scalar multiplication is the most important and key operation in cryptosystems built on HEC and EC. Montgomery Ladder algorithm is an efficient and important algorithm to implement scalar multiplications for defending against side channel attacks. While Montgomery Ladder algorithm on elliptic curve is being improved in recent years, there is not much advance on hyperelliptic

* 基金项目: 国家自然科学基金(61070019, 60703089); 山东省自然科学基金(ZR2010FQ015); 山东省优秀中青年科学家科研奖励基金(2008BS01011)

收稿时间: 2011-10-26; 定稿时间: 2012-12-27

curves. Lange proposed a way to design faster addition formula on hyperelliptic curves but did not result in a practical solution. This paper improves the addition for divisor classes for the first time to implement faster Montgomery Ladder algorithm. New technique is applied for improving the formulae on various coordinates. The analysis and experimental results show that the new formulae are faster than previous ones. Over fields of character two and Type II curves, the new formulae is 4%~8.4% faster than the ones known before. And the Montgomery Ladder algorithm implemented in this paper is secure against side channel attacks.

Key words: hyperelliptic curve; scalar multiplication; Montgomery ladder; addition formulae; divisor classes

超椭圆曲线(HEC)与椭圆曲线(EC)类似,也可以用来建立密码体制.自从 Koblitz[1]于 1988 年提出了在超椭圆曲线上建立密码体制以来,超椭圆曲线上的运算公式和标量乘计算方法也在不断地发展和改进.与椭圆曲线密码体制相比,超椭圆曲线密码体制的安全性更高,其上的运算参数及密钥长度都更短.对应于参数为 1 024bits 长度的 RSA 密码体制,现在推荐在椭圆曲线和超椭圆曲线上建立阶为 2^{160} 大小的群,就可以满足我们的安全需求[2].在使用椭圆曲线时,操作数的长度为 160bits 即可满足要求.在亏格 g=2 的超椭圆曲线上,操作数的长度约为 160/g=80bits;在亏格为 3 的超椭圆曲线上,操作数的长度约为 55bits.已经证明,在亏格大于 3 的超椭圆曲线上建立密码体制,安全性有所降低.当使用亏格为 3 的超椭圆曲线时,也要小心地选择安全曲线以避免攻击.本文讨论亏格为 2 的超椭圆曲线上的运算.

虽然超椭圆曲线上的操作数大多很短,但其运算公式却很复杂.随着研究的深入,各种高效的除子类的运算公式陆续被提出[3−5],并且使得超椭圆曲线上的标量乘运算速度越来越接近于椭圆曲线上的标量乘计算.

Montgomery 阶梯算法[2,6]是计算标量乘的一种重要算法,且因为其一定程度上可以抵抗简单边带信道攻击[2]而被广泛应用.这种算法在加解密计算和利用椭圆曲线(超椭圆曲线)分解大素数等应用中,都有重要的作用和意义.近几年,椭圆曲线上的 Montgomery 阶梯算法和相应的点运算公式一直在不断地得以改进,比如 2010 年的密码年会上,Bernstein 等人[7]在 binary Edwards 曲线上使用 Montgomery 阶梯算法实现了一般椭圆曲线上至今最快的标量乘计算.在超椭圆曲线上,人们利用 Kummer surface 设计 Montgomery 计算方案[2,8].直接在一般曲线上设计快速除子类的运算公式来实现较为高效的 Montgomery 阶梯算法,是另一条改进途径,但在这个方向上却一直没有太大进展.Lange 曾经探讨过这种快速公式存在的可能性[9],却并未得到一个实用、有效的计算公式.

本文提出了更为快速和高效的除子类加法公式来加快超椭圆曲线上的 Montgomery 阶梯算法的计算.首先给出了新的思路来进行公式上的代数变换,然后在多种坐标系下优化设计出了新的公式.对于特征为 2 的域上的超椭圆曲线,根据不同的参数取值[2]可将其分类,分为类型 I、类型 II、类型 III 曲线.分析和比较结果表明,新的公式在投影坐标系(projective coordinate)、N 坐标系(new coordinate)和 R 坐标系(recent coordinate)下,在类型 II 和类型 III 的曲线上,速度与之前的公式相比都有明显提高.在类型 II 的曲线上提高最为明显,而类型 II 曲线正好也是我们在构建超椭圆曲线密码体制时最为常用的一种曲线.当域上的平方和乘法之间的运算量的比率满足 0.8 比 1 时,类型 II 的曲线上新的公式比之前的公式快了 7.7%~7.8%.当我们用正规基来表示域上的元素时,平方的计算量可以忽略不计,这时新的公式快了 4%~8.3%.当然,新的技术也可以进一步应用于参数更为特殊的曲线上,以得到更快的算法.本文的研究结果表明,超椭圆曲线上的 Montgomery 阶梯算法可以直接在公式层面提高效率.本文最后使用新的公式实现了超椭圆曲线上的 Montgomery 阶梯算法,在保持高效的同时可以抵抗简单边带信道攻击.

本文第 1 节介绍相关基本概念和基础知识,主要是已有的运算公式.第 2 节提出新的代数变换方法和思路,并设计出新的运算公式.第 3 节是性能分析和仿真实验.最后一节对本文进行总结,并对进一步的工作进行探讨.

1 基础知识

1.1 超椭圆曲线上的运算

本节对亏格为 2 的超椭圆曲线进行简要介绍,具体内容可参考文献[2,6,10,11].设 F_q 是一个特征为 p 的有限域,其中,$q=p^v$, $\bar{F}_q$ 是 F_q 的代数闭包,下面定义了亏格为 g 的超椭圆曲线:

$$C:y^2+h(x)y=f(x),$$

其中,$f(x)\in F_q[x]$首项系数为 1,次数为 $2g+1$;$h(x)\in F_q[x]$是一个次数最大为 g 的多项式,对于二元组 $(a,b)\in \bar{F}_q\times\bar{F}_q$,没有合适的解满足 $b^2+h(a)b=f(a)$和偏微分等式 $2b+h(a)=0,h'(a)b-f'(a)=0$.这是为了保证超椭圆曲线不是超奇异的.曲线 C/F_q 就叫作定义在 F_q 上的亏格为 g 的超椭圆曲线.

设 K 是 $\bar{F}_q$ 的一个子域,超椭圆曲线 C 在 K 上所有的点构成了一个点集:

$$C(K)=\{(x,y):x,y\in K,y^2+h(x)y=f(x)\}\cup\{\infty\},$$

其中,∞表示无穷远点.$C(K)$上是没有合适的群运算的,我们构建超椭圆曲线上的 Jacobian 来获得一个实用的离散对数系统.首先定义超椭圆曲线 C 上的一个除子(divisor): $D=\sum m_pP$, 其中,$m_p\in Z,P\in C(K)$,而除子的次数 $\deg(D)=\sum m_p$. 其中,所有次数为 0 的除子构成了一个集合 $Div_0C(D)$.我们再通过函数域 $F_q[x,y]/(y^2+h(x)-f(x))$,构造出主除子群(group of principle divisors):$P_C(F_q)=\{Div(F):F\in F_q(C)\}$.最后,即得到了 C 在域 F_q 上的 Jacobian $J_C(F_q)=Div_0C(F_q)/P_C(F_q)$.超椭圆曲线的安全性即建立在 Jacobian 的离散对数问题上.

下一步,Jacobian 上的每一个元素,都可以用唯一的 reduced divisor 来表示,Mumford[12]给出了表示方法:

(1) u 是首项系数为 1 的;

(2) $\deg v<\deg u\leqslant g$;

(3) $u|v^2+vh-f$.

Cantor[13]在此基础上提出了如何计算两个除子类的加法,但是 Cantor 算法中含有求多项式的最大公约数的运算以及多个求逆运算.之后,有人使用各种技巧改进了 Cantor 算法.最近的导出公式是 Lange[6]提出的,如下所示:

$$k=(f-v_2h-v_2^2)/u^2,\ s\equiv(v_1-v_2)/u_2 \bmod u_1,\ l=su_2,\ u=(k-s(l+h+2v_2))/u_1,$$
$$u'=u \text{ made monic},\ v'=-h-(l+v_2) \bmod u'.$$

这就将 Cantor 算法中计算 semi-reduced divisor 的过程和计算 reduced divisor 的过程合并且简化了,并去掉了计算最大公约数的步骤.其中,s 的计算可以利用 Sylvester resultant 来实现.具体的计算公式如算法 1 所示.算法中的 M 表示域上的乘法,S 表示平方计算,而 I 表示元素求逆.

算法 1. 一般情况下除子类加法公式[2,5]

Addition, deg u_1=deg u_2=2		
Input	$[u_1,v_1]$, $[u_2,v_2]$, $u_i=x^2+u_{i1}x+u_{i0}$, $v_i=v_{i1}x+v_{i0}$;	
Output	$[u',v']=[u_1,v_1]+[u_2,v_2]$.	
Step	Expression	Operations
1	Compute resultant r of u_1, u_2: $z_1=u_{11}-u_{21}$, $z_2=u_{20}-u_{10}$, $z_3=u_{11}z_1+z_2$; $r=z_2z_3+z_1^2u_{10}$;	1S+3M
2	Compute almost inverse of u_2 modulo u_1($inv=r/u_2 \bmod u_1$): $inv_1=z_1$, $inv_0=z_3$;	
3	Compute $s'=rs\equiv(v_1-v_2)inv \bmod u_1$: $w_0=v_{10}-v_{20}$, $w_1=v_{11}-v_{21}$, $w_2=inv_0w_0$, $w_3=inv_1w_1$; $s_1'=(inv_0+inv_1)(w_0+w_1)-w_2-w_3(1+u_{11})$, $s_0'=w_2-u_{10}w_3$; if $s_1'=0$ see below	5M
4	Compute $s''=x+s_0/s_1=x+s_0'/s_1'$ and s_1: $w_1=(rs_1')^{-1}(=1/r^2s_1)$, $w_2=rw_1(=1/s_1')$, $w_3=s_1'^2w_1(=s_1)$; $w_4=rw_2(=1/s_1)$, $w_5=w_4^2$, $s_0''=s_0'w_2$;	1I+2S+5M
5	Compute $l'=s''u_2=x^3+l_2'x^2+l_1'x+l_0'$: $l_2'=u_{21}+s_0''$, $l_1'=u_{21}s_0''+u_{20}$, $l_0'=u_{20}s_0''$;	2M
6	Compute $u'=(s(l+h+2v_2)-k)/u_1=x^2+u_1'x+u_0'$: $u_0'=(s_0''-u_{11})(s_0''-z_1+h_2w_4)-u_{10}+l_1'+(h_1+2v_{21})w_4+(2u_{21}+z_1-f_4)w_5$; $u_1'=2s_0''-z_1+h_2w_4-w_5$;	3M
7	Compute $v'\equiv-h-(l+v_2) \bmod u'=v_1'x+v_0'$: $w_1=l_2'-u_1'$, $w_2=u_1'w_1+u_0'-l_1'$, $v_1'=w_2w_3-v_{21}-h_1+h_2u_1'$; $w_2=u_0'w_1-l_0'$, $v_0'=w_2w_3-v_{20}-h_0+h_2u_0'$;	4M
Total		1I+3S+22M

算法 1. 一般情况下除子类加法公式[2,5](续)

Special case $s=s_0$		
4′	Compute s: $inv=1/r$, $s_0=s_0' inv$;	1I+1M
5′	Compute $u'=(k-s(l+h+2v_2))/u_1=x+u_0'$: $u_0'=f_4-u_{21}-u_{11}-s_0^2-s_0h_2$;	1S
6′	Compute $v'\equiv -h-(l+v_2) \bmod u'=v_0'$: $w_1=s_0(u_{21}+u_0')+h_1+v_{21}+h_2u_0'$, $w_2=s_0+v_{20}+h_0$, $v_0'=u_0'w_1-w_2$;	2M
Total		1I+2S+11M

我们这里只讨论绝大多数情况下的加法公式,也就是 u_1 和 u_2 的次数均为 2 且互素时的公式.其他情况出现的概率非常小,且计算过程简单很多,这里不再赘述,具体过程参考文献[2,9].

1.2 Montgomery阶梯算法

Montgomery 阶梯算法如算法 2 所示.实际上,算法中的 D_1 总是存储了目前为止所求出的结果,而 D_2 中总是存储了 D_1+D 的值.这样,最后输出的 D_1 就是要求的结果.在算法的循环体中,由于每次循环都需要做一次二倍运算和一次除子类的加法运算,这就使得每次执行循环体时的运算量是一样的.所以,Montgomery 阶梯标量乘算法一定程度上可以抵抗简单边带信道攻击.

算法 2. Montgomery 标量乘算法

输入:reduced divisor D 和整数 $s=(s_{l-1},...,s_0)$;
输出:sD.

1. $D_1=0$, $D_2=D$
2. For $i=l-1$ down to 0 do
3. If $s_i=0$ then $D_1=2D_1$, $D_2=D_1+D_2$
4. Else $D_1=D_1+D_2$, $D_2=2D_2$
5. Return D_1

可以看到,算法 2 中的加法运算是特殊的加法,其中的变量总是满足 $D_2-D_1=D$,且 D 是固定、已知的.在椭圆曲线密码体制中也有类似的性质,人们利用这一点设计出了高效的点运算公式[2,7].2010 年的密码年会上,Bernstein 等人[7]使用 Montgomery 阶梯算法,在一般的 binary Edwards 曲线上实现了较快的椭圆曲线标量乘计算.上述 binary Edwars 曲线其实是 Bernstein 等人仿照大素数域上的 Edwards 曲线设计出来的,但是二元域上所有的椭圆曲线方程都可以有理变换成为 binary Edwards 曲线的形式.

在超椭圆曲线上如何应用 Montgomery 阶梯算法,也是人们长久以来不断考虑的问题.Lange[9]设计适用于 Montgomery 标量乘算法的快速除子类加法公式,但是没有得到最终的公式.在文献[2]中,Lange 从另外一个角度,仿照椭圆曲线上的 Montgomery 形式,设计出一种特殊的超椭圆曲线的 Montgomery 形式.而后,她利用 Kummer surface 设计出较为高效的除子类运算公式.但是只有某类超椭圆曲线可以等价变换到这种特殊的 Montgomery 形式,并且算法的空间复杂性较大.在计算中,还需要额外的预计算和预存储,进行坐标表示的转换.

我们在下一节介绍设计出的新的超椭圆曲线上的快速加法公式.新的技术适用于普通的超椭圆曲线,特别是在特征为 2 的域上,可以明显提高公式的运算效率.

1.3 域 F_{2^n} 上的同态曲线

根据文献[2]中的描述,在特征为偶数的域上,超椭圆曲线中的 $f(x)$和 $h(x)$就定义在 F_{2^d} 上,这样的超椭圆曲线可分为如下 3 类:

I. deg $h=2$;

II. deg $h=1$;

III. deg $h=0$.

对于所有形如类型 1 的曲线,均可以变换成如下两种形式:

Ia. $y^2+(x^2+h_1x+t\,h_1^2)y=x^5+\varepsilon tx^4+f_1x+f_0$;

Ib. $y^2+x(x+h_1)y=x^5+\varepsilon tx^4+f_1x+f_0$.

其中,$\varepsilon\in F_2$,t 表示一个 trace 为 1 的元素(当 n 为奇数时,t=1).

对于类型 II,曲线均可以有理变换为如下两种形式:

- $y^2+xy=x^5+f_3x^3+\varepsilon x^2+f_0$,$n$ 为奇数时;
- $y^2+h_1xy=x^5+\varepsilon' x^3+\varepsilon t\,h_1^2\,x^2+f_0$,$n$ 为偶数时.

类型 III 中的曲线是超奇异的,在 Frey-Ruck 攻击下其安全强度会有所降低,所以不能直接用来构建基于离散对数的超椭圆曲线密码体制.但是,这种曲线可以用来构建基于双线性对的密码体制.所以,设计这种曲线上的快速加法公式依然具有重要意义.

我们在下一节针对不同的情况,设计快速的加法公式.本文主要是对类型 II 和类型 III 推导快速加法公式.为了方便起见,在下一节中始终假设 h_2=0.由于通过等价变化,所有的超椭圆曲线都可以变换到 h_2=0,1 的形式,所以类型 II 和类型 III 的曲线可以占总曲线数目的一半.而为了更方便地得到阶很大的 Jacobian[2],通常选取域 F_{2^d},其中,d 为奇数,而这种情况也是我们着重研究的对象.此外,下一节中的代数变换技术在 h_2=1 时也是适用的,只是在这里不再专门对这种情况进行优化了.

2 新的运算公式

2.1 设计快速加法公式

由于$-(u_1,v_1)=-(u_1,-h-v_1 \bmod u_1)$且 $h_2=\{0,1\}$,所以当 h 的次数为 2 时,我们有

$$-(u_1,v_1)=-(u_1,-h-v_1+u_1);$$

而当 h 的次数小于 2 时,就有

$$-(u_1,v_1)=-(u_1,-h-v_1).$$

我们先来讨论最为通用的加法公式,这时 h 为 2 次多项式.

对于 $D=D_2-D_1=-(u_1,v_1)+(u_2,v_2)=(u_1,-h-v_1)+(u_2,v_2)$:

- $k=(f-v_2h-v_2^2)/u_2$, $s\equiv(-h-v_1-v_2)/u_2 \bmod u_1$, $l=su_2$, $u=(k-s(l+h+2v_2))/u_1$;
- $u'=u$ made monic, $v'=-h-(l+v_2) \bmod u'$.

可以得到 $u_1u=k-s(l+h+2v_2)$.

我们需要计算 $D_3=D_2+D_1=(u_1,v_1)+(u_2,v_2)$:

- $k_3=(f-v_2h-v_2^2)/u_2$, $s_3\equiv(v_1-v_2)/u_2 \bmod u_1$, $l_3=s_3u_2$, $u_3=(k-s_3(l_3+h+2v_2))/u_1$;
- $u_3'=u$ made monic, $v_3'=-h-(l_3+v_2) \bmod u_3'$.

可以得到 $u_1u_3=k_3-s_3(l_3+h+2v_2)$.

由于有 $k=k_3$,所以,

$$u_1(u-u_3)=s_3(l_3+h+2v_2)-s(l+h+2v_2)=s_3l_3-sl+(h+2v_2)(s_3-s)=(s_3-s)((s_3+s)u_2+h+2v_2).$$

因为 $s_3+s=(-h-2v_2)e_2 \bmod u_1$, $s-s_3=(-h-2v_1)e_2 \bmod u_1$,所以有,

$$u_1(u-u_3)=(s_3-s)(((-h-2v_2)e_2+iu_1)u_2+h+2v_2),$$

其中,$i=-h_2e_{21}x+e_{21}(h_2u_{11}-h_1-2v_{21})-h_2e_{20}$.

于是,

$$u_1(u-u_3)=(s_3-s)((-h-2v_2)e_2u_2+iu_1u_2+h+2v_2).$$

因为 $e_2u_2=1-e_1u_1$,所以,

$$u_1(u-u_3)=(s_3-s)((h+2v_2)e_1u_1+iu_1u_2).$$

最后可得:

$$u-u_3=(s_3-s)((h+2v_2)e_1+iu_2).$$

因为 $\deg((h+2v_2)e_1+iu_2)=1$,所以,

$$(h+2v_2)e_1+iu_2=(h+2v_2)e_1 \bmod u_2.$$

这样,$u-u_3=((-h-2v_1)e_2 \bmod u_1)((h+2v_2)e_1 \bmod u_2)$.

我们现在来推导其在仿射坐标系下的导出公式以求出 D_3:

- 首先,计算 $e_2 \bmod u_1$ 和 $e_1 \bmod u_2$:计算 u_1 和 u_2 的 resultant r 和 re_2, re_1:
  - $z_1=u_{11}-u_{21}$, $z_2=u_{20}-u_{10}$, $z_3=u_{11}z_1+z_2$;
  - $re_{11}=-z_1$, $re_{10}=u_{21}(u_{21}-u_{11})+u_{10}-u_{20}$, $re_{21}=z_1$, $re_{20}=z_3$.
- 其次,我们来计算 $s'=rs$ 和 $s'_3=rs_3$.
- 然后,我们计算 $c_1=c_{11}x+c_{10}=s'+s'_3$ 和 $c_2=c_{21}x+c_{20}=s'-s'_3$:
  1. 注意到 $r(s+s_3)=(-h-2v_2)re_2 \bmod u_1$,而我们想要求 $r(ax+b)=(h+2v_2)re_1 \bmod u_2$.这两个等式分别可以写为 $-r(s+s_3)=(h+2v_2)re_2+iu_1$ 和 $r(ax+b)=(h+2v_2)re_1+ju_2$.其中,

     $$i=h_2e_{21}x+h_2e_{20}+e_{21}(h_1-h_2u_{11}+2v_{21}),\ j=h_2e_{11}x+h_2e_{10}+e_{11}(h_1-h_2u_{21}+2v_{21}).$$

  2. 两个等式的等号两边分别乘以 u_2 和 u_1,可以得到:

     $$-r(s+s_3)u_2=(h+2v_2)re_2u_2+iu_1u_2,\ r(ax+b)u_1=(h+2v_2)re_1u_1+ju_2u_1.$$

     因为 $e_1u_1+e_2u_2=1$ 并且 $i+j=0$,所以两个等式相加可以得到 $r(ax+b)u_1-r(s+s_3)u_2=(h+2v_2)r$.

     于是有 $r(ax+b)u_1=(h+2v_2)r+r(s+s_3)u_2$,也即 $r(ax+b)=((h+2v_2)r+r(s+s_3)u_2)/u_1$.

     可以得到 $ra=c_{11}$,$rb=h_2+c_{10}+a(u_{21}-u_{11})$.
- 最后,我们计算 $u'_3=(c_{21}x+c_{20})(ax+b+s'^2_1u')/s'^2_{31}$.由于上面等式的参数中均有 r^2 这个因子,所以最后除以 s'^2_{31} 的时候就约掉了.具体运算公式见算法 3.至于 v'_3 的计算,与算法 1 中的一样.

算法 3. 通用的加法公式

Addition, deg u_1=deg u_2=2		
Input Output	$[u,v]$, $[u_1,v_1]$, $[u_2,v_2]$,其中,$[u',v']=[u_2,v_2]-[u_1,v_1]$, $u_i=x^2+u_{i1}x+u_{i0}$, $v_i=v_{i1}x+v_{i0}$; $[u'_3,v'_3]=[u_1,v_1]+[u_2,v_2]$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $z_1=u_{21}-u_{11}$, $z_2=u_{10}-u_{20}$, $z_3=z_1^2$; $e_{21}=-z_1$, $e_{20}=u_{11}e_{21}-z_2$, $e_{11}=z_1$, $e_{10}=z_3-e_{20}$; $r=z_2e_{10}+z_3u_{20}$;	3M+1S
2	Compute $s'_3=rs_3=(v_1-v_2)e_2 \bmod u_1$: $w_0=e_{21}(v_{11}-v_{21})$, $w_1=e_{20}(v_{10}-v_{20})$, $w_2=(v_{11}-v_{21}+v_{10}-v_{20})(e_{21}+e_{20})$, $w_3=w_2-w_0-w_1$; $s'_{31}=w_3-w_0u_{11}$, $s'_{30}=w_1-u_{10}w_0$;	5M
3	Compute $s'=rs=(-h-v_1-v_2)e_2 \bmod u_1$: $w_0=e_{21}(-h_1-v_{11}-v_{21})$, $w_1=e_{20}((-h_0-v_{10}-v_{20})$, $w_2=(-h_1-v_{11}-v_{21}-h_0-v_{10}-v_{20})(e_{21}+e_{20})$, $w_3=w_2-w_0-w_1$; $s'_1=w_3-w_0u_{11}$, $s'_0=w_1-u_{10}w_0$; $c_1=c_{11}x+c_{10}=s'+s'_3$, $c_2=c_{21}x+c_{20}=s'-s'_3$;	5M
4	Compute $1/r$ and $1/s_1$: $w_0=(rs'_1)^{-1}$, $w_1=w_0r$, $w_1=w_1^2$, $w_2=w_0s'_1$;	1I+3M+1S
5	Compute $u'_3=((c_{21}x+c_{20})(c_{11}x+c_{10}+c_{11}(u_{21}-u_{11}))+s_1^2u')/s_{31}^2=x^2+u'_{31}x+u'_{30}$: $w_4=c_{11}c_{21}$, $w_5=c_{20}(c_{10}+c_{11}(u_{21}-u_{11}))$, $w_6=((c_{21}+c_{20})(c_{11}+c_{10}+c_{11}(u_{21}-u_{11}))$, $u'_{31}=(w_6-w_4-w_5+s_1^2u'_1)w_1$; $u'_{30}=(w_5+s_1^2u'_0)w_1$;	8M
6	Compute $s_3=s'_3/r$: $s_{31}=s'_{31}w_2$, $s_{30}=s'_{30}w_2$;	2M
7	Compute $v'_3\equiv-h-(s_3u_2+v_2) \bmod u'_3=v'_{31}x+v'_{30}$: $w_1=s_{31}(u_{21}-u_{31})$, $w_2=s_{30}(u_{20}-u_{30})$, $w_3=(s_{31}+s_{30})(u_{21}-u_{31}+u_{20}-u_{30})$; $v'_{30}=-w_2-h_0-v_{20}$, $v'_{31}=-w_3-v_{21}-h_1$;	5M
Total		1I+2S+31M

上面的推导都是等价的代数变换,所以推导过程也就是算法 3 的正确性证明,于是我们得到如下定理:

定理 1. 算法 3 的输出结果与算法 1 是相同的.

注意到,当 $s=s_0$ 时,我们仍然将其作为特殊情况,如算法 1 所示进行处理,所以我们就不再将其包括在算法 3 中了.特殊情况出现的概率很低,一次标量乘计算中平均最多出现一次特殊情况;且如果出现了,按照已有的公式可以很高效地进行计算.所以本文中设计其他算法时也这样处理,只讨论普通情况下的算法公式.

与算法 1 相比,算法 3 不足够快,但在特征为 2 的域上,我们可以改进其中某几步运算来推导除子类的快速加法公式,这是原有的算法 1 所不具有的特点.我们首先给出类型 II 的曲线上的加法运算公式.为了方便寻找合适阶的群,我们都是在域 F_{2^n} (n 为奇数)上建立密码体制,所以只考虑这种域上的公式即可.

在这种情况下,算法 3 的第 3 步中有 $c_1=c_2=c_{11}x+c_{10}=s'+s'_3=(u_{10}+u_{20})x+u_{10}(u_{11}+u_{21})$,

而第 5 步有 $u'_3=((c_1+c_{11}(u_{11}-u_{21}))c_2-u)/s'^2_{31}=(((u_{10}+u_{20})x+u_{10}(u_{11}+u_{21}))((u_{10}+u_{20})x+u_{20}(u_{11}+u_{21}))+u)/s'^2_{31}$.

因为 $u'_3+u'=ax+b$,可以得到:

$$u'_3=((u_{10}+u_{20})^2x^2+(u_{10}+u_{20})^2(u_{11}+u_{21})x+u_{10}u_{20}(u_{11}+u_{21})^2)+(u_{10}+u_{20})^2u)/s'^2_{31}+u'.$$

也就是说,我们可以先计算 $u'_3+u'=(ax+b)/s'^2_{31}$,然后计算 $u'_3=(ax+b)/s'^2_{31}+u'$.

综上所述,我们有 $u'_{31}=(u_{10}+u_{20})^2(u_{11}+u_{21}+u_1)w_1+u_1, u'_{30}=(u_{10}u_{20}(u_{11}+u_{21})^2+(u_{10}+u_{20})^2u_0)w_1+u_0$.

具体计算过程见算法 4.

算法 4. 在域 F_{2^n} 上的 Montgomery 阶梯加法公式(类型 II,n 为奇数)

Addition, deg u_1=deg u_2=2		
Input Output	$[u,v]$, $[u_1,v_1]$, $[u_2,v_2]$,其中,$[u',v']=-[u_1,v_1]+[u_2,v_2]$, $u_i=x^2+u_{i1}x+u_{i0}$, $v_i=v_{i1}x+v_{i0}$; $[u'_3,v'_3]=[u_1,v_1]+[u_2,v_2]$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $z_1=u_{21}+u_{11}$, $z_2=u_{10}+u_{20}$, $z_3=z_1^2$, $z_4=z_2^2$, $z_5=z_3u_{10}$; $e_{21}=z_1$, $e_{20}=u_{11}e_{21}+z_2$, $e_{11}=z_1$, $e_{10}=z_3+e_{20}$; $r=z_2e_{20}+z_5$;	3M+2S
2	Compute $s'_3=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(v_{11}+v_{21})$, $w_1=e_{20}(v_{10}+v_{20})$, $w_2=(v_{11}+v_{21}+v_{10}+v_{20})(e_{21}+e_{20})$, $w_3=w_2+w_0+w_1$; $s'_{31}=w_3+w_0u_{11}$, $s'_{30}=w_1+u_{10}w_0$;	5M
3	Compute $w_2=1/r$ and $w_1=1/s'^2_{31}$: $w_0=(rs'_{31})^{-1}$, $w_1=w_0r$, $w_1=w_1^2$, $w_2=w_0s'_{31}$;	1I+3M+1S
4	Compute $u'_3=((c_1+(s_{31}+s_1)(u_{11}+u_{21}))c_2+u')/s'^2_{31}$: $u'_{31}=z_4(u_{11}+u_{21}+u'_1)w_1+u'_1$; $u'_{30}=(u_{20}z_5+z_4u'_0)w_1+u'_0$;	5M
5	Compute $s_3=s'_3/r$: $s_{31}=s'_{31}w_2$, $s_{30}=s'_{30}w_2$;	2M
6	Compute $v'_3\equiv h+(s_3u_2+v_2) \bmod u'_3=v'_{31}x+v'_{30}$: $w_1=s_{31}(u_{21}+u'_{31})$, $w_2=s_{30}(u_{20}+u'_{30})$, $w_3=(s_{31}+s_{30})(u_{21}+u'_{31}+u_{20}+u'_{30})$; $v'_{30}=w_2+w_1u'_{30}+v_{20}$, $v'_{31}=w_3+w_1+w_2+v_{21}+w_1u'_{31}+1$;	5M
Total		1I+3S+23M

当在域 F_{2^n} (n 为偶数)上进行计算时,有 $h=h_1x$,这时,$c_1=c_2=c_{11}x+c_{10}=s'+s'_3=h_1x((u_{10}+u_{20})x+u_{10}(u_{11}+u_{21}))$.

于是,我们只需将第 3 步中 w_1 的运算变为 $w_1=h_1^2w_0^2$ 即可保证算法正确.同时,将第 6 步中 v'_{31} 的计算变为 $v'_{31}=w_3+v_{21}+h_1$.当我们选取的 h_1 是非零比特非常稀疏的随机数时,有关的 h_1 计算是可以忽略不计的.这样,算法的运算量依然是 1I+3S+23M.

当 $h=h_0$ 时,超椭圆曲线的形式为类型 III,这时也有高效的公式,见算法 5.注意到,我们也选取 h_0 是稀疏的二进制数.

算法 5. 在域 F_{2^n} 上的 Montgomery 阶梯加法公式(类型 III)

Addition, deg u_1=deg u_2=2		
Input Output	$[u,v]$, $[u_1,v_1]$, $[u_2,v_2]$,其中,$[u,v]=-[u_1,v_1]+[u_2,v_2]$, $u_i=x^2+u_{i1}x+u_{i0}$, $v_i=v_{i1}x+v_{i0}$; $[u'_3,v'_3]=[u_1,v_1]+[u_2,v_2]$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $z_1=u_{21}+u_{11}$, $z_2=u_{10}+u_{20}$, $z_3=z_1^2$, $z_4=z_2^2$; $e_{21}=z_1$, $e_{20}=u_{11}e_{21}+z_2$, $e_{11}=z_1$, $e_{10}=z_3+e_{20}$; $r=z_2e_{10}+z_3u_{20}$;	3M+1S
2	Compute $s'_3=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(v_{11}+v_{21})$, $w_1=e_{20}(v_{10}+v_{20})$, $w_2=(v_{11}+v_{21}+v_{10}+v_{20})(e_{21}+e_{20})$, $w_3=w_2+w_0+w_1$; $s'_{31}=w_3+w_0u_{11}$, $s'_{30}=w_1+u_{10}w_0$;	5M
3	Compute $w_2=1/r$ and $w_1=h_0/s'^2_{31}$: $w_0=(rs'_{31})^{-1}$, $w_1=w_0r$, $w_1=w_1^2$, $w_2=w_0s'_{31}$;	1I+3M+1S
4	Compute $u'_3=((c_1+(s_{31}+s_1)(u_{11}-u_{21}))c_2-u)/s'^2_{31}$: $u'_{31}=z_3(u_{11}+u_{21}+u'_1)w_1+u'_1$; $u'_{30}=(e_{10}e_{20}+(u_{11}+u_{21})u'_0)w_1+u'_0$;	5M
5	Compute $s_3=s'_3/r$: $s_{31}=s'_{31}w_2$, $s_{30}=s'_{30}w_2$;	2M
6	Compute $v'\equiv -h-(s_3u_2+v_2) \bmod u'=v'_1x+v'_0$: $w_1=s_{31}(u_{21}+u_{31})$, $w_2=s_{30}(u_{20}+u_{30})$, $w_3=(s_{31}+s_{30})(u_{21}+u_{31}+u_{20}+u_{30})$; $v'_{30}=w_2+v_{20}+h_0$, $v'_{31}=w_3+v_{21}$;	5M
Total		1I+2S+23M

2.2 在投影坐标系上的加法公式

求逆运算是以上公式中最为耗时的运算,且一般情况下是普通乘法运算的 8 倍(大素数域)或者 15 倍(特征为 2 的域)[2].在椭圆曲线和超椭圆曲线密码体制中,我们为了避免求逆运算,经常采用一些特殊的坐标系,如投影坐标系、Jacobian 坐标系、LD 坐标系等等[6].在超椭圆曲线密码体制上,出现的可以避免求逆运算的坐标系有投影坐标系和 Lange 提出的 N 坐标系、R 坐标系.我们在本节设计算法 4 和算法 5 在投影坐标系下的公式.

算法 6 的正确性也很好验证,只要满足 $U_3/Z_3=u_3$ 和 $V_3/Z_3=v_3$ 即可.注意到算法中的 D 是仿射坐标系下的,也就说有 $Z=1$,且在计算过程中 D 的值一直未变.以下定理就是算法 6 的正确性证明.

定理 2. 算法 6 的输出结果正是算法 4 输出的除子类的投影坐标系上的形式.

证明:我们需要验证 $U_3/Z_3=u'_3$ 和 $V_3/Z_3=v'_3$,其中,U_3,V_3 和 Z_3 是算法 6 的输出,而 u'_3 和 v'_3 是算法 4 的输出.首先验证 $U_{31}/Z_3=u'_{31}$ 和 $U_{30}/Z_3=u'_{30}$.由于算法 6 与算法 4 中的 e_{21} 相比满足 $e_{21}/Z_1Z_2=u_{21}+u_{11}$,则称算法 6 中的 e_{21} 与算法 4 中的 e_{21} 相比包含有因子 Z_1Z_2.我们可以依次确定 e_{20},r 和 s_{31} 的因子分别为 $Z_1^2Z_2$,$Z_1^3Z_2^2$ 和 $Z_1^3Z_2^2$.s_{30} 中包含的因子与 s_{31} 相等.算法 6 第 4 步中求出 $Z_3=Z'_3R$.下面来验证 U_{3i} 和 V_{3i} 的值(i=0,1),只需将每步的变量逐步代入即可:

- $U_{31}/Z_3=(z_1R_3+U_1(Z'_3+R_3R_2))/Z'_3=z_4R_1(z_1+u'_1R_2)/Z'_3+u'_1=(u_{10}+u_{20})^2(u_{11}+u_{21}+u'_1)/s_{31}^2+u'_1=u'_{31}$;
- $U_{30}/Z_3=(U_{20}z_5R_2+(R_3+Z'_3)U'_0)=(u_{20}z_5+z_4u'_0)/s_{31}^2+u'_0=u'_{30}$.

分别对 V_{31} 和 V_{30} 中的每一项进行处理:

- 首先看 $w_1U_{30}/Z_3=((s_{31}w_1)/(s_{31}Z_3))u'_{30}=s_{31}(u_{21}+u'_{31})u'_{30}$,其中,等式最右边是算法 4 的第 6 步里的一项;
- 同样,$(w_2+R'V_{20})/Z_3=w_2+v_{20}$.

这样,我们验证完每一项后,可以得到 $v'_{31}=V_{31}/Z_3$ 和 $v'_{30}=V_{30}/Z_3$.

综上,定理 2 得证. □

Lange 提出的在投影坐标系上的 F_2^n 上的加法公式为 49M+4S.这样,我们的公式就比之前的公式少用了 4 个乘法.

算法 6. 在投影坐标系上的 Montgomery 阶梯加法公式(类型 II,n 为奇数)

Addition, deg u_1=deg u_2=2		
Input Output	$D_1=[U_{11},U_{10},V_{11},V_{10},Z_1]$, $D_2=[U_{21},U_{20},V_{21},V_{20},Z_2]$, $D=D_2-D_1=[U_1,U_0,V_1,V_0,Z]$; $D_3=D_1+D_2$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $z_1=U_{11}Z_2+U_{21}Z_1$, $z_2=U_{20}Z_1+U_{10}Z_2$, $z_3=z_1^2$, $z_4=z_2^2$, $z_5=z_3U_{10}$; $e_{21}=z_1$, $e_{20}=U_{11}z_1+z_2Z_1$; $r=z_2e_{20}+z_5$;	8M+2S
2	Compute $s_3=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(V_{11}Z_2+V_{21}Z_1)$, $w_1=e_{20}(V_{20}Z_1+V_{10}Z_2)$; $w_2=(V_{11}Z_2+V_{21}Z_1+V_{20}Z_1+V_{10}Z_2)(Z_1e_{21}+e_{20})$; $s_{31}=w_2+w_1+w_0(Z_1+U_{11})$, $s_{30}=w_1+U_{10}w_0$;	10M
3	Compute $u_3'=((c_1+(s_{31}+s_1)(u_{11}+u_{21}))c_2+s_1^2u)/s_{31}^2$: $R_2=Z_1Z_2$, $Z_3'=s_{31}^2$, $R_1=Z_1^2$, $R_1=R_1R_2$, $R_3=z_4R_1$; $R'=rs_{31}$, $R=R'Z_2$, $R'=rZ_3'$; $U_{31}=z_1R_3+U_1(Z_3'+R_3R_2)$; $U_{30}=U_{20}z_5R_1+U_0(Z_3'+R_3R_2)$;	12M+2S
4	Compute $V_3\equiv h+(s_3u_2+v_2) \bmod u_3=v_{31}'x+v_{30}'$: $w_1=U_{21}Z_3'+U_{31}Z_2$, $w_2=s_{30}(U_{20}Z_3'+U_{30}Z_2)$, $w_3=(s_{31}+s_{30})(U_{21}Z_3'+U_{31}Z_2+U_{20}Z_3'+U_{30}Z_2)$; $Z_3=Z_3'R$, $V_{30}=(w_2+V_{20}R')s_{31}+w_1U_{30}$, $V_{31}=(w_3+w_1+w_2+V_{21}R')s_{31}+w_1U_{31}+Z_3$;	13M
5	Adjust: $U_{31}=U_{31}R$, $U_{30}=U_{30}R$;	2M
Total		4S+45M

此外,对于类型 III 的曲线上的加法运算公式,其在投影坐标系下的算法形式列在算法 7 中.这个公式的计算量也比 Lange 给出的公式少用了 1 个乘法.由于算法 7 与算法 6 在大多数步骤中是相同的,因此只需验证第 3 步中 $U_{30}=u_{30}'$ 即可验证算法的正确性.

算法 7. 在投影坐标系上的 Montgomery 阶梯加法公式(类型 III)

Addition, deg u_1=deg u_2=2		
Input Output	$D_1=[U_{11},U_{10},V_{11},V_{10},Z_1]$, $D_2=[U_{21},U_{20},V_{21},V_{20},Z_2]$, $D=D_2-D_1=[U_1,U_0,V_1,V_0,Z]$; $D_3=D_1+D_2$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $z_1=U_{11}Z_2+U_{21}Z_1$, $z_2=U_{20}Z_1+U_{10}Z_2$, $z_3=z_1^2$; $e_{21}=z_1$, $e_{20}=U_{11}z_1+z_2Z_1$, $e_{11}=z_1$, $e_{10}=U_{21}z_1+z_2Z_2$; $r=z_2e_{20}+z_3U_{10}$;	10M+1S
2	Compute $s_3=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(V_{11}Z_2+V_{21}Z_1)$, $w_1=e_{20}(V_{20}Z_1+V_{10}Z_2)$; $w_2=(V_{11}Z_2+V_{21}Z_1+V_{20}Z_1+V_{10}Z_2)(Z_1e_{21}+e_{20})$; $s_{31}=w_2+w_1+w_0(Z_1+U_{11})$, $s_{30}=w_1+U_{10}w_0$;	10M
3	Compute $u_3'=((c_1+(s_{31}+s_1)(u_{11}+u_{21}))c_2+s_1^2u)/s_{31}^2$: $R_2=Z_1Z_2$, $Z_3'=s_{31}^2$, $R_1=Z_1^2$, $R_1=R_1R_2$, $R_3=z_3R_1$; $R'=rs_{31}$, $R=R'Z_2$, $R'=rZ_3'$; $U_{31}=R_3(z_1+R_2U_1)+U_1Z_3'$, $U_{30}=e_{10}e_{20}R_1+U_0(Z_3'+R_2R_3)$;	13M+2S
4	Compute $V_3\equiv h+(s_3u_2+v_2) \bmod u_3=v_{31}'x+v_{30}'$: $w_1=U_{21}Z_3'+U_{31}Z_2$, $w_2=s_{30}(U_{20}Z_3'+U_{30}Z_2)$, $w_3=(s_{31}+s_{30})(U_{21}Z_3'+U_{31}Z_2+U_{20}Z_3'+U_{30}Z_2)$; $Z_3=Z_3'R$, $V_{30}=(w_2+V_{20}R')s_{31}+w_1U_{30}$, $V_{31}=(w_3+w_1+w_2+V_{21}R')s_{31}+w_1U_{31}+Z_3$;	13M
5	Adjust: $U_{31}=U_{31}R$, $U_{30}=U_{30}R$;	2M
Total		4S+48M

2.3 在N坐标系下的加法公式

Lange 为了得到更高效的除子类运算公式,提出了 N 坐标系(new coordinate).在这种坐标系下,二倍运算和

加法运算的计算量变为 34M+7S 和 48M+4S[2,5].N 坐标系的形式为一个六元组$[U_1,U_0,V_1,V_0,Z_1,Z_2]$,其对应于仿射坐标系下的坐标值为$[U_1/Z_1^2,U_0/Z_1^2,V_1/(Z_1^3Z_2),V_0/(Z_1^3Z_2)]$.当在特征为偶数的域上进行计算时,我们需要几个辅助的变量来提高计算效率.这时,可将 New 坐标系上的一个点表示为$[U_1,U_0,V_1,V_0,Z_1,Z_2,z_1,z_2,z_3,z_4]$,其中,

$$z_1=Z_1^2,\ z_2=Z_2^2,\ z_3=Z_1Z_2,\ z_4=z_1z_3.$$

当然,在下面的计算中可以看到,公式中并没有用到变量 Z_1 和 Z_2,所以我们在实际计算时可以将这两个变量去掉以节省空间.而为了保证算法的清晰和可读,我们在本节的推导和分析中将这两个变量保留下来.算法中的输入$[U_1,U_0,V_1,V_0,1,1,1,1]$不发生改变,也就是说一直有 $U_i=u_i'$ 和 $V_i=v_i'$ $(i=0,1)$.

定理 3. 算法 8 的输出结果相对应的仿射坐标是算法 4 的输出.

证明:对于 z_{31},z_{32},z_{33} 和 z_{34} 的正确性,在第 4 步的计算过程中已经表示了出来.下面只需验证有 $u_{31}'=U_{31}/Z_{31}^2,u_{30}'=U_{30}/Z_{31}^2,v_{31}'=V_{31}/(Z_{31}^3Z_{32})$ 和 $v_{30}'=V_{30}/(Z_{31}^3Z_{32})$ 即可,其中,u_{31}',u_{30}',v_{31}' 和 v_{30}' 是算法 4 中的输出.我们首先来确定几个中间变量中的因子.由于算法 8 中的 $e_{21}/z_{11}z_{21}=u_{21}+u_{11}$,则称为算法 8 中的 e_{21} 与算法 4 中的 e_{21} 相比包含有因子 $z_{11}z_{21}$.我们可以依次确定 e_{20},r 和 s_{31} 的因子分别为 $z_{11}^2z_{21},z_{11}^3z_{21}^2$ 和 $z_{11}^2z_{21}z_{14}z_{24}$.$s_{30}$ 中包含的因子与 s_{31} 是相等的.算法中求出 $Z_{31}=s_{31},Z_{32}=rz_{13}z_{24}$.下面来验证 U_{3i} 和 V_{3i} 的值$(i=0,1)$:

- $U_{31}/z_{31}=(R_3y_1+(R_3+z_{31})u_1')/z_{31}=y_4R_2(y_1+u_1')/z_{31}+u_1'=(u_{10}+u_{20})^2(u_{11}+u_{21}+u_1')/s_{31}^2+u_1'=u_{31}'$;
- $U_{30}/z_{31}=U_{20}y_5R_2+(R_3+z_{31})U_0'=(u_{10}z_5+z_4u_0')/s_{31}^2+u_0'=u_{30}'$.

分别对 V_{31} 和 V_{30} 中的每一项进行处理:

- 首先来看 $w_1U_{30}/z_{34}=((s_{31}w_1)/(s_{31}Z_{32}))u_{30}'=s_{31}(u_{21}+u_{31}')u_{30}'$, 等式最右边是算法 4 中第 6 步里的一项;
- 同样,$(w_2+R_1V_{20})Z_{31}=w_2+v_{20}$.

这样,我们验证完每一项后可以得到 $v_{31}'=V_{31}/(Z_{31}^3Z_{32})$ 和 $v_{30}'=V_{30}/(Z_{31}^3Z_{32})$.

综上,定理得证. □

算法 8. 在 N 坐标系下的 Montgomery 阶梯加法公式(类型 II,n 为奇数)

Addition, deg u_1=deg u_2=2		
Input	$D_1=[U_{11},U_{10},V_{11},V_{10},Z_{11},Z_{12},z_{11},z_{12},z_{13},z_{14}]$, $D_2=[U_{21},U_{20},V_{21},V_{20},Z_{21},Z_{22},z_{21},z_{22},z_{23},z_{24}]$, $D=D_2-D_1=[U_1,U_0,V_1,V_0,Z_1,Z_2,z_1,z_2,z_3,z_4]$;	
Output	$D_3=D_1+D_2$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $y_1=U_{21}z_{11}+U_{11}z_{21}$, $y_2=U_{10}z_{21}+U_{20}z_{11}$, $y_3=y_1^2$, $y_4=y_2^2$, $y_5=y_3U_{10}$; $e_{21}=y_1$, $e_{20}=U_{11}e_{21}+y_2z_{11}$; $r=y_2e_{10}+y_5$;	8M+2S
2	Compute $s_3'=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(V_{11}z_{24}+V_{21}z_{14})$, $w_1=e_{20}(V_{10}z_{24}+V_{20}z_{14})$, $w_2=(V_{11}z_{24}+V_{21}z_{14}+V_{10}z_{24}+V_{20}z_{14})(e_{21}z_{11}+e_{20})$; $s_{31}=w_2+w_1+w_0(U_{11}+z_{11})$, $s_{30}=w_1+U_{10}w_0$;	10M
3	Compute $u_3'=((c_1+(s_{31}+s_1)(u_{11}+u_{21}))c_2+u')/s_{31}'^2$: $Z_{31}=s_{31}$, $z_{31}=s_{31}^2$, $R_1=z_{11}z_{21}$, $R_2=z_{14}z_{23}$, $R_2=R_2^2$, $R_2=R_1R_2$, $R_3=y_4R_2$; $U_{31}=R_3y_1+(R_3+z_{31})U_1'$; $U_{30}=U_{20}y_5R_2+(R_3+z_{31})U_0'$;	9M+2S
4	Compute $v_3'\equiv h+(s_3u_2+v_2) \bmod u_3'=v_{31}'x+v_{30}'$: $R_1=rz_{13}$, $Z_{32}=z_{24}R_1$, $z_{32}=Z_{32}^2$, $z_{33}=Z_{31}Z_{32}$, $R_1=z_{31}R_1$, $z_{34}=z_{31}z_{33}$; $w_1=U_{21}z_{31}+U_{31}z_{21}$, $w_2=s_{30}(U_{20}z_{31}+U_{30}z_{21})$, $w_3=(s_{31}+s_{30})(U_{21}z_{31}+U_{31}z_{21}+U_{20}z_{31}+U_{30}z_{21})$; $V_{30}=(w_2+R_1V_{20})Z_{31}+w_1U_{30}$, $V_{31}=(w_3+w_2+R_1V_{21})Z_{31}+w_1(U_{31}+z_{31})+z_{34}$;	17M+1S
Total		4S+44M

在 N 坐标系下,我们设计类型 III 的曲线上的加法公式如算法 9 所示.算法 9 与算法 8 的区别在于第 1 步中多计算了 e_{11} 和 e_{10},且第 3 步中 U_{30} 的计算发生了变化.由于有 $U_{30}=(e_{10}e_{20}+(u_{11}+u_{21})^2u_0')w_1+u_0'=u_{30}'$,而等号最右边为算法 5 中的一项,所以算法 9 也是正确的,其输出变换为仿射坐标系后正是算法 5 的输出.

算法 9. 在 N 坐标系下的 Montgomery 阶梯加法公式(类型 III,n 为奇数)

Addition, deg u_1=deg u_2=2		
Input	$D_1=[U_{11},U_{10},V_{11},V_{10},Z_{11},Z_{12},z_{11},z_{12},z_{13},z_{14}]$, $D_2=[U_{21},U_{20},V_{21},V_{20},Z_{21},Z_{22},z_{21},z_{22},z_{23},z_{24}]$, $D=D_2-D_1=[U_1,U_0,V_1,V_0,Z_1,Z_2,z_1,z_2,z_3,z_4]$;	
Output	$D_3=D_1+D_2$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $y_1=U_{21}z_{11}+U_{11}z_{21}$, $y_2=U_{10}z_{21}+U_{20}z_{11}$, $y_3=y_1^2$, $y_4=y_2^2$, $y_5=y_3U_{10}$; $e_{21}=y_1$, $e_{20}=U_{11}e_{21}+y_2z_{11}$, $e_{11}=y_1$, $e_{10}=U_{21}e_{11}+y_2z_{21}$; $r=y_2e_{10}+y_5$;	10M+2S
2	Compute $s_3'=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(V_{11}z_{24}+V_{21}z_{14})$, $w_1=e_{20}(V_{10}z_{24}+V_{20}z_{14})$, $w_2=(V_{11}z_{24}+V_{21}z_{14}+V_{10}z_{24}+V_{20}z_{14})(e_{21}z_{11}+e_{20})$; $s_{31}=w_2+w_1+w_0(U_{11}+z_{11})$, $s_{30}=w_1+U_{10}w_0$;	10M
3	Compute $u_3'=((c_1+(s_{31}+s_1)(u_{11}+u_{21}))c_2+u')/s_{31}'^2$: $Z_{31}=s_{31}$, $z_{31}=s_{31}^2$, $R_1=z_{11}z_{21}$, $R_2=z_{14}z_{23}$, $R_2=R_2^2$, $R_2=R_1R_2$, $R_3=y_4R_2$; $U_{31}=R_3y_1+(y_4R_2+z_{31})U_1'$; $U_{30}=e_{10}e_{20}R_2+(y_3R_2+z_{31})U_0'$;	10M+2S
4	Compute $v_3'\equiv h+(s_3u_2+v_2) \bmod u_3'=v_{31}'x+v_{30}'$: $R_1=rz_{13}$, $Z_{32}=z_{24}R_1$, $z_{32}=Z_{32}^2$, $z_{33}=Z_{31}Z_{32}$, $R_1=z_{31}R_1$, $z_{34}=z_{31}z_{33}$; $w_1=U_{21}z_{31}+U_{31}z_{21}$, $w_2=s_{30}(U_{20}z_{31}+U_{30}z_{21})$, $w_3=(s_{31}+s_{30})(U_{21}z_{31}+U_{31}z_{21}+U_{20}z_{31}+U_{30}z_{21})$; $V_{30}=(w_2+R_1V_{20})Z_{31}+w_1U_{30}$, $V_{31}=(w_3+w_2+R_1V_{21})Z_{31}+w_1(U_{31}+z_{31})+z_{34}$;	17M+1S
Total		4S+47M

2.4 在R坐标系下的加法公式

本文主要研究的是 $h_2=0$ 的情况,在文献[2]中,Lange 提出了一种 R 坐标系,我们称其为 R 坐标系,以加快这种情况下的二倍运算;但同时,加法计算会大大变慢.由于二倍计算加快的速率比加法计算变慢的速率要高,所以最后还是得到了一种较快的计算标量乘的方法.下面我们使用本文中提出的技术来改进 R 坐标系下的加法计算.在 R 坐标系下,一个点的坐标可以表示为$[U_1,U_0,V_1,V_0,Z,z]$,其相对应的仿射坐标系上的点为$(U_1/Z,U_0/Z,V_1/z,V_0/z)$,其中,$z=Z^2$.具体的运算公式见算法 10 和算法 11.

算法 10. 在 R 坐标系下的加法公式(类型 II)

Addition, deg u_1=deg u_2=2		
Input	$D_1=[U_{11},U_{10},V_{11},V_{10},Z_1,z_1]$, $D_2=[U_{21},U_{20},V_{21},V_{20},Z_2,z_2]$, $D=D_2-D_1=[U_1,U_0,V_1,V_0,Z,z]$;	
Output	$D_3=D_1+D_2$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $y_1=U_{21}Z_1+U_{11}Z_2$, $y_2=U_{10}Z_2+U_{20}Z_1$, $y_3=y_1^2$, $y_4=y_2^2$, $y_5=y_3U_{10}$; $e_{21}=y_1$, $e_{20}=U_{11}e_{21}+y_2Z_1$; $r=y_2e_{10}+y_5$;	8M+2S
2	Compute $s_3'=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(V_{11}z_2+V_{21}z_1)$, $w_1=e_{20}(V_{10}z_2+V_{20}z_1)$, $w_2=(V_{11}z_2+V_{21}z_1+V_{10}z_2+V_{20}z_1)(e_{21}z_1+e_{20})$; $s_{31}=w_2+w_1+w_0(U_{11}+z_1)$, $s_{30}=w_1+U_{10}w_0$;	10M
3	Compute $u_3'=((c_1+(s_{31}+s_1)(u_{11}+u_{21}))c_2+u')/s_{31}'^2$: $Z_3'=s_{31}^2$, $R_1=Z_1Z_2$, $R_2=z_1^2z_2$, $R_3=R_1R_2$, $R_5=y_4R_3$, $R_4=R_5R_1$; $U_{31}=R_5y_1+(R_4+Z_3')U_1'$; $U_{30}=U_{20}y_5R_3+(R_4+Z_3')U_0'$;	10M+2S
4	Compute $v_3'\equiv h+(s_3u_2+v_2) \bmod u_3'=v_{31}'x+v_{30}'$: $R=rR_1$, $R_2=rZ_1$, $R_3=R_2Z_3'$, $Z_3=RZ_3'$, $z_3=Z_3^2$; $w_1=s_{31}(U_{21}z_{31}+U_{31}z_{21})$, $w_2=s_{30}(U_{20}z_{31}+U_{30}z_{21})$, $w_3=(s_{31}+s_{30})(U_{21}z_{31}+U_{31}z_{21}+U_{20}z_{31}+U_{30}z_{21})$; $R_4=w_1R_2$, $V_{30}=(w_2+Z_3V_{20})R_3+U_{30}R_4$, $V_{31}=(w_1+w_3+w_2+Z_3V_{21})R_3+U_{31}R_4+z_3$;	18M+1S
5	$U_{31}=U_{31}R$, $U_{30}=U_{30}R$;	2M
Total		5S+48M

算法 11. 在 R 坐标系下的加法公式(类型 III)

Addition, deg u_1=deg u_2=2		
Input Output	$D_1=[U_{11},U_{10},V_{11},V_{10},Z_1,z_1]$, $D_2=[U_{21},U_{20},V_{21},V_{20},Z_2,z_2]$, $D=D_2-D_1=[U_1,U_0,V_1,V_0,Z,z]$; $D_3=D_1+D_2$.	
Step	Expression	Operations
1	Compute $e_1=r/u_1 \bmod u_2$ and $e_2=r/u_2 \bmod u_1$: $y_1=U_{21}Z_1+U_{11}Z_2$, $y_2=U_{10}Z_2+U_{20}Z_1$, $y_3=y_1^2$, $y_4=y_2^2$, $y_5=y_3U_{10}$; $e_{21}=y_1$, $e_{20}=U_{11}e_{21}+y_2Z_1$, $e_{11}=y_1$, $e_{10}=U_{21}e_{11}+y_2Z_2$; $r=y_2e_{10}+y_5$;	8M+2S
2	Compute $s_3'=(v_1+v_2)e_2 \bmod u_1$: $w_0=e_{21}(V_{11}z_2+V_{21}z_1)$, $w_1=e_{20}(V_{10}z_2+V_{20}z_1)$, $w_2=(V_{11}z_2+V_{21}z_1+V_{10}z_2+V_{20}z_1)(e_{21}z_1+e_{20})$; $s_{31}=w_2+w_1+w_0(U_{11}+z_1)$, $s_{30}=w_1+U_{10}w_0$;	10M
3	Compute $u_3'=((c_1+(s_{31}+s_1)(u_{11}+u_{21}))c_2+u')/s_{31}'^2$: $Z_3'=s_{31}^2$, $R_1=Z_1Z_2$, $R_2=z_1^2z_2$, $R_3=R_1R_2$, $R_5=y_4R_3$, $R_4=R_5R_1$, $R_2=R_1R_3$; $U_{31}=R_5y_1+(R_4+Z_3')U_1'$; $U_{30}=e_{10}e_{20}R_3+(y_3R_2+Z_3')U_0'$;	12M+2S
4	Compute $v_3'\equiv h+(s_3u_2+v_2) \bmod u_3'=v_{31}'x+v_{30}'$: $R=rR_1$, $R_2=rZ_1$, $R_3=R_2Z_3'$, $Z_3=RZ_3'$, $z_3=Z_3^2$; $w_1=s_{31}(U_{21}z_{31}+U_{31}z_{21})$, $w_2=s_{30}(U_{20}z_{31}+U_{30}z_{21})$, $w_3=(s_{31}+s_{30})(U_{21}z_{31}+U_{31}z_{21}+U_{20}z_{31}+U_{30}z_{21})$; $R_4=w_1R_2$, $V_{30}=(w_2+Z_3V_{20})R_3+U_{30}R_4$, $V_{31}=(w_1+w_3+w_2+Z_3V_{21})R_3+U_{31}R_4+z_3$;	18M+1S
5	$U_{31}=U_{31}R$, $U_{30}=U_{30}R$;	2M
Total		5S+50M

对定理 4 的证明,也即证明了算法 10 的正确性.

定理 4. 算法 10 的输出结果相对应的仿射坐标是算法 4 的输出.

证明:通过验证下面的等式来证明:

- $U_{31}/Z_3=(R_5y_1+(R_4+Z_3')U_1')/Z_3=(u_{10}+u_{20})^2(u_{11}+u_{21}+u_1')/s_{31}^2+u_1'=u_{31}'$;
- $U_{30}/Z_3=(U_{20}y_5R_3+(R_4+Z_3')U_0')/Z_3=(u_{10}z_5+z_4u_0')/s_{31}^2+u_0'=u_{30}'$;
- $V_{31}/z_3=((w_2+R_4V_{20})R_3+w_1U_{30}R_2)/z_3=v_{31}'$;
- $V_{30}/z_3=((w_3+w_2+R_4V_{21})R_3+w_1(U_{31}R_2+z_3)+z_3)/z_3=v_{30}'$. □

同样,对于算法 11 的验证也只需要验证第 3 步中的 U_{30} 就可以了.

3 算法分析和仿真

本节首先比较分析各种算法的效率,然后使用算法实现超椭圆曲线上的 Montgomery 阶梯算法.各个坐标系下的加法公式的运算量列在表 1 中.表中的 M,S 和 I 分别表示域上的乘法、平方和求逆.

Table 1 The computational complexity of Montgomery Ladder algorithm over F_{2^n}

表 1 在域 F_{2^n} 上的 Montgomery 阶梯加法公式的运算量

	传统算法[2,5]	类型 II 的曲线	类型 III 的曲线
投影坐标系	49M+4S	45M+4S	48M+4S
N 坐标系	48M+4S	44M+4S	47M+4S
R 坐标系	50M+8S	48M+5S	50M+5S
仿射坐标系	I+3S+22M	I+3S+23M	I+2S+23M

可以看到,本文中提出的新的公式在投影坐标系、N 坐标系和 R 坐标系下都比之前的公式计算速度要快.提高幅度最大的是在类型 II 的曲线上,在投影坐标系和 N 坐标系下分别比之前的公式少用了 4 个乘法,在 R 坐标系下少用了 2 个乘法和 3 个平方.由于在不同的实现函数库中,域上的平方和乘法计算之间的速度比值是不同的,这样就造成新的算法运行效率提高的比率也会有所不同[2].在一些实现中,S=0.8M,那么新的加法公式在类型 II 的曲线上,在投影、N 坐标系和 R 坐标系下分别比之前的公式快了 7.7%,7.8%和 7.8%.当 1S=0.5M 时,本文中的公式在类型 II 的曲线上比之前的公式快了 7.8%,8%和 6.5%.当我们用正规基来表示域 F_2^n 上的元素时,

一次平方计算只是一次移位操作,这样,平方计算的运算量与乘法相比可以忽略不计.这时,我们的公式在类型 II 的曲线上比之前的公式快了 8.2%,8.3%和 4%.本文提出的公式在类型 III 的曲线上的效率与之前的相比也有明显的提高,在投影坐标系和 N 坐标系下分别少用了 1 个乘法,而在 R 坐标系下少用了 3 个平方.

我们对各个公式进行了仿真实验.最终的运算结果证明了算法的正确性,各种算法的运行时间列在了表 2 中.我们的仿真平台是在 Tinkpad T500(CPU P8800,DDR2 4G)上,使用了 NTL5.4.2 和 Magma online 两个函数库分别进行了实验,选择的域是 $F_{2^{255}}$.在这两个函数库中,在特征为 2 的域上的乘法和平方计算之间的比值满足 1S=0.8M–0.75M.每次仿真选择 100 000 个随机参数,然后取运行时间的平均值.

Table 2 The sumulation of algorithms of addition formula(n=255bits, Unit: ms)

表 2 加法公式的仿真实验结果(n=255bits,单位:ms)

	已有算法[2,5]		类型 II 的曲线上		类型 III 的曲线上	
	NTL	Magma	NTL	Magma	NTL	Magma
投影坐标系	1.06	0.029 1	0.98	0.026 2	1.04	0.028 1
N 坐标系	1.04	0.029 0	0.96	0.026 1	1.02	0.276
R 坐标系	1.15	0.031 0	1.05	0.028 8	1.09	0.292

本文的加法公式是为了实现超椭圆曲线上的 Montgomery 阶梯算法而设计的.Montgomery 阶梯算法实现的标量乘计算,可以抵抗简单边带信道攻击.而当使用之前的常规公式时,只有“总是二倍加法”标量乘算法可以实现这个功能.在特征为偶数的域上,投影坐标系下计算二倍除子类的运算量为 38M+7S,但是,当 h_2=0 时降低为 25M+6S.这也是为什么 h_2=0 的曲线特别适合 Montgomery 阶梯算法的实现.在域 $F_{2^{255}}$ 上实现 Montgomery 阶梯算法时,需要做 254 次二倍和加法运算.如果在类型 II 的曲线上实现 Montgomery 阶梯标量乘算法,我们将各种坐标系下的运算量列入表 3.根据文献[2],类型 II 的曲线上 N 坐标系和 R 坐标系下的二倍计算的运算量为 39M+6S 和 23M+8S.表 4 中是不同情况下标量乘计算的仿真结果,该结果也证明了我们的算法的正确性和有效性.可以看到,在投影坐标系下,类型为 II 的超椭圆曲线上的 Montgomery 阶梯算法最为高效.

Table 3 The computational complexity of Montgomery Ladder algorithm over $F_{2^{255}}$ (Type II)

表 3 在域 $F_{2^{255}}$ 上的 Montgomery 阶梯算法的运算量(类型 II)

投影坐标系	N 坐标系	R 坐标系
17780M+2540S	21082M+2540S	18034M+3302S

Table 4 The simulation of Montgomery Ladder algorithm over $F_{2^{255}}$ (Type II, Unit: s)

表 4 在域 $F_{2^{255}}$ 上对 Montgomery 阶梯算法的仿真结果(类型 II,单位:s)

投影坐标系		N 坐标系		R 坐标系	
NTL	Magma	NTL	Magma	NTL	Magma
0.41	0.011 1	0.48	0.012 6	0.44	0.011 5

4 结 论

本文研究了如何在超椭圆曲线上为 Montgomery 阶梯算法设计高效的除子类加法计算公式.本文通过代数变换技术,使得超椭圆曲线上的 Montgomery 阶梯算法在公式层面上有了改进和提高.分析和比较结果表明,新的公式在投影坐标系、N 坐标系和 R 坐标系下,在类型 II 和类型 III 的曲线上,速度都有明显提高.在类型 II 曲线上的提高最为明显.当域上的平方和乘法之间运算速度的比率满足 1S=0.8M 时,类型 II 的曲线上新的公式比之前的公式快了 7.7%~7.8%;当 1S=0.5M 时,新的公式快了 6.5%~8%.当我们用正规基来表示域上的元素时,新的公式快了 4%~8.3%.

本文中提出的技术还可以与其他技术结合使用,以进一步提高公式的运行速度.比如,文献[14]中将二倍和加法结合起来,而减少乘法个数的技术.当然,椭圆曲线上出现的很多其他变换技术,比如有理变换曲线来设计公式的技术[15],也可以拿来运用到超椭圆曲线上.至于如何应用,有待于我们进一步的研究.

致谢 非常感谢 Ali Miri 教授对本文提出的建议和意见.

References:

[1] Koblitz N. Hyperelliptic cryptosystems. Journal of Cryptology, 1989,1:139−150. [doi: 10.1007/BF02252872]

[2] Avanzi R, Cohen H, Doche C, Frey G, Lange T, Nguyen K, Vercauteren F. Handbook of Elliptic and Hyperelliptic Curve Cryptography. Boca Raton: Chapman & Hall, 2005.

[3] Lange T. Efficient doubling on genus two curves over binary fields. In: Haddad H, Omicini A, Wainwright RL, Liebrock LM, eds. Proc. of the Selected Areas in Cryptography (SAC 2004). LNCS 3357, 2005. 170−181. [doi: 10.1007/978-3-540-30564-4_12]

[4] Wollinger T, Pelzl J, Paar C. Cantor versus Harley: Optimization and analysis of explicit formulae for hyperelliptic curve cryptosystems. IEEE Trans. on Computers, 2005,54(7):861−872. [doi: 10.1109/TC.2005.109]

[5] Lange T. Formulae for arithmetic on genus 2 hyperelliptic curves. Applicable Algebra in Engineering, Communication and Computing, 2005,15(5):295−328. [doi: 10.1007/s00200-004-0154-8]

[6] Blake I, Seroussi G, Smart N, eds. Advances in Elliptic Curve Cryptography. London Mathematical Society 317, Cambridge University Press, 2005.

[7] Bernstein D. Batch binary Edwards. In: Halevi S, ed. Proc. of the Advances in Cryptology—CRYPTO 2009. LNCS 5677, Springer-Verlag, 2009. 317−336. [doi: 10.1007/978-3-642-03356-8_19]

[8] Duquesne S. Montgomery Ladder for all genus 2 curves in characteristic 2. LNCS 5130 (Arithmetic of Finite Fields), 2008. 174−188. [doi: 10.1007/978-3-540-69499-1_15]

[9] Lange T. Montgomery addition for genus two curves. LNCS 3076 (Algorithmic Number Theory), 2004. 309−317. [doi: 10.1007/978-3-540-24847-7_23]

[10] Menezes A, Wu YH, Zuccherato R. An elementary introduction to hyperelliptic curves. In: Proc. of the Algebraic Aspects of Cryptography. 1998. http://cacr.uwaterloo.ca/techreports/1997/corr96-19.ps

[11] Wollinger T. Software and hardware implementation of hyperelliptic curve cryptosystems [Ph.D. Thesis]. Bochum: Ruhr-University of Bochum, 2004.

[12] Mumford D. Tata Lectures on Theta II. Birkhäuser Boston: Springer-Verlag, 1993.

[13] Cantor DG. Computing in the Jacobian of a hyperelliptic curve. Mathematics of Computation, 1987,48:95−101.

[14] Fan XX, Gong G. Efficient explicit formulae for genus 2 hyperelliptic curves over prime fields and their implementations. In: Adams C, Miri A, Wiener M, eds. Proc. of the Selected Areas in Cryptography 2007. LNCS, Springer-Verlag, 2007. 155−172. [doi: 10.1007/978-3-540-77360-3_11]

[15] Bernstein D, Lange T, Farashahi R. Binary Edwards curves. In: Proc. of the 10th Int'l Workshop on Cryptographic Hardware and Embedded Systems (CHESS 2008). LNCS 5154, 2008. 244−265. [doi: 10.1007/978-3-540-85053-3_16]

李明(1982－),男,山东肥城人,博士,工程师,主要研究领域为算法分析与设计,密码学.
E-mail: luaming@msn.com

孔凡玉(1978－),男,博士,副教授,CCF 会员,主要研究领域为密码学,信息安全.
E-mail: fanyukong@sdu.edu.cn

朱大铭(1963－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为算法分析与设计,生物计算.
E-mail: dmzhu@sdu.edu.cn