DOI: 10.14107/j.cnki.kzgc.151058
文章编号：1671-7848(2018)03-0504-05

基于折半运算的带符号阶乘展开式标量乘算法

庞根明
(晋城职业技术学院 信息工程系，山西 晋城 048026)

摘 要：阶乘展开表示形式标量乘算法是椭圆曲线密码的一种快速标量乘算法。为能进一步有效提升阶乘展开表示形式的标量乘算法的计算效率，且因折半运算比倍点运算的执行效率更加高效，通过把折半运算的方法应用在带符号的阶乘展开表示形式的标量乘算法中，在预计算和主循环计算阶段分别用折半运算替代倍点运算，提出了一种基于折半运算的带符号阶乘展开式标量乘算法。算法的性能分析结果表明：与传统的阶乘展开表示形式标量乘算法相比，新算法的运算效率提高了约 60.78 %。

关键词：圆曲线密码；标量乘法；带符号阶乘展开式算法；折半运算
中图分类号：TP309　　**文献标识码**：A

The Scalar Multiplication Algorithm of Signed Factorial Expansion Based on Point Halving

PANG Gen-ming
(Department of Information Engineering, Jincheng Institute of Technology, Jincheng 048026, China)

Abstract: The scalar multiplication algorithm of factorial expansion representation is a fast scalar multiplication algorithm in elliptic curve cryptography. Due to that point halving has higher execution efficiency than double operations, point halving is applied in the signed factorial expansion scalar multiplication algorithm in order to further improve the factorial expansion scalar multiplication algorithms, and then a signed factorial expansion scalar multiplication based on point halving is proposed in which the double operation is replaced by the point halving in the stage of pre-computation and the main loop. The performance analysis results show that the new algorithm could improve the efficiency by 60.78 % compared with the traditional factorial expansion scalar multiplication algorithm.

Key words: Ellipse curve cryptography; scalar multiplication; signed factorial expansion algorithm; point halving

1 引　言

椭圆曲线密码(Elliptic Curve Cryptography, ECC)体制为Neal Koblitz[1]与Victor Miller[2]最早于1985年提出来的，它的安全性主要是构建在椭圆曲线上对数问题难解性的基础之上的，主要是通过利用有限域上的椭圆曲线有限群来替换离散对数中的有限循环群，从而实现了基于椭圆曲线的加解密运算密码体制，同其他的传统密码算法相比而言具备所需的密钥更短，存储空间更小等诸多优点[3-4]。ECC的核心计算是在椭圆曲线群上的标量乘法运算，即计算 $kP=P+P+\cdots+P$，它的运算效率为决定ECC密码系统整体实现效率的最为关键的运算。对于椭圆曲线密码的标量乘运算，国内外研究学者做了海量的相关工作，为推动椭圆曲线密码运算效率的逐步提高，加速推进椭圆曲线密码的实际应用做了非常大的贡献。为了能够进一步提升椭圆曲线密码的计算效率，目前主要采用两类较有效的方法：

一是提高底层域的运算效率，二是对标量实施重编码，通过减少点加操作和倍点操作的执行次数来提升计算效率。文献[5]的石润华等人提出的基于带符号阶乘展开式的标量乘快速算法，就是通过实现标量的重新编码，将标量表示为阶乘展开式的形式，因而使得标量乘法运算变换为多标量乘法运算，有效地提升了标量乘的计算效率，但是存在着较大的存储开销和预计算开销。而文献[6]的Kundsen E等人则提出了一种基于折半运算的快速标量乘算法，主要是通过采用更为高效的折半运算代替倍点

收稿日期：2015-11-26；**修回日期**：2016-04-20
基金项目：国家自然科学基金资助项目(51308126).
作者简介：庞根明(1981-)，男，山西晋城人，研究生，讲师，主要从事计算机网络、物联网等方面的教学与科研工作。

运算，以此能够大幅度提升标量乘运算的执行效率，相对于传统的标量乘来说运算效率提升了近一倍左右，是一种用于提升标量乘计算效率的非常有效的技术手段。因而，通过深入详细地研究基于带符号阶乘展开式的标量乘快速算法的计算特性，把折半运算的方法应用在它的预计算阶段和主循环计算阶段中，进而由此给出了一种更加高效的基于折半运算的带符号阶乘展开式标量乘算法(Algorithm of Signed Factorial Expansion Scalar Multiplication Based on Point Halving，SFEPH)，所给的SFEPH算法将能够进一步大幅有效提升椭圆曲线密码标量乘的计算效率。根据与传统的带符号阶乘展开式标量乘算法的计算效率对比分析结果可知，所给出的新快速算法计算效率能够提高约60.78 %。

2 折半运算

假定存在有一个二进制的域F_{2^m}，可在这个域上定义一个椭圆曲线E，下面式(1)给出了该域上椭圆曲线的定义：

$$E:\ y^2+xy=x^3+ax+b,\ a,b\in F_{2^m},b\neq 0 \quad (1)$$

若果在椭圆曲线E上定义一个点$P=(x_1,y_1)$，同时定义的这个点满足$P\in E(F_{2^m})$，并且存在条件$P\neq -P$。则能够在仿射坐标系下计算倍点运算可得$2P=(x_3,y_3)$，且其坐标值的计算方法如下式：

$$\begin{aligned} x_3&=\mu^2+\mu+a\\ y_3&=x_1^2+\mu x_3+x_3\\ \mu&=x_1+\frac{y_1}{x_1}\end{aligned} \quad (2)$$

下面可以根据文献[7]所给出的定理，能够得出在椭圆曲线上可以定义一种倍点操作的逆操作运算，如以下定理1所示。

定理1 (折半运算) 当$\{G\}$是椭圆曲线E上一个n阶的子群时，其中，n为奇数，倍点运算和折半运算在子群$\{G\}$上为自同构的，所以对于任一点$Q\in\{G\}$，都存在有一点$P\in\{G\}$，可使$Q=2P$。

椭圆曲线上定义的倍点操作的逆操作运算被称作折半运算，是指在先给定一点$Q=(x_3,\ y_3)$的情况下，计算得出定义在二进制域F_{2^m}中椭圆曲线E上的另一点$P=(x_1,\ y_1)$。也就是根据式(2)计算出μ，x_1，y_1，计算方法如式(3)所示：

$$\begin{aligned} \mu&=\frac{1}{2}+\frac{1}{2}\sqrt{1-4(a-x_3)}\\ x_1&=\sqrt{y_3-x_3(\mu+1)}\\ y_1&=\mu x_1-x_1^2\end{aligned} \quad (3)$$

椭圆曲线上的折半操作运算如算法1所示[8]：

算法1 椭圆曲线上折半运算算法：

输入：$Q=(x_3,\ y_3)$

输出：$P=\frac{1}{2}Q=(x_1,\ y_1)$

① 利用式$\mu^2+\mu=x_3+a$求得μ。

② 计算$\eta=y_3+\mu x_3$。

③ 若有$Tr(\eta)=0$，则$y_1=\mu$，$x_1=\sqrt{\eta+x_3}$。否则，$y_1=\mu+1$，$x_1=\sqrt{\eta}$。

④ 返回值(x_1,y_1)。

由此可知，若假设令$\sum_{i=0}^{t}k_i'2^i$为$2^t k(\mathrm{mod}\,n)$的$w-NAF$表示形式，则可将标量$k=\sum_{i=0}^{t}\frac{k_{t-i}'}{2^i}(\mathrm{mod}\,n)$的标量乘法运算变换成$Q=kP=\sum_{i=0}^{t}\frac{k_{t-i}'}{2^i}\cdot P$，因而可知计算$kP$与计算$\sum_{i=0}^{t}\frac{k_{t-i}'}{2^i}\cdot P$为等价的，其中折半运算的运算效率为$2M$[9]($M$为模乘运算，假设令$H$为折半运算，则$H=2M$)。

3 基于带符号阶乘展开式标量乘算法

3.1 整数的带符号阶乘展开式编码算法

带符号的阶乘展开式的表示形式可以说是一种比较有效的数域系统的表示方法，把其方法与椭圆曲线密码的标量乘法运算相结合起来，能够较有效地提升椭圆曲线密码的计算效率。其主要思想是将大整数标量表示成类基为$(1!,2!,\ldots,s!)$的一组小整数的标量，展开以后的小整数标量通常会较小而且个数也较少，能够在很大程度上提高大整数标量乘法的运算效率。下面式(4)给出了标量k的带符号阶乘展开式的表示方法为

$$k=\sum_{i=1}^{n-1}k_i(i!)=k_{n-1}(n-1)!+\cdots+k_2 2!+k_1 \quad (4)$$

式中，$k\in[0,n!]$，$|k_i|\leq\lfloor(i+1)/2\rfloor$，则式(2)称作整数$k$的带符号阶乘展开式，简记为$SFE(k)$。

下面算法2则给出了整数k的带符号阶乘展开式编码算法具体的描述过程[10]。

算法2 整数k的带符号的阶乘展开形式的编码算法

输入：n，k。

输出：$STE(k)$。

① 令$\omega=k$，$k_1=\omega(\mathrm{mod}\,2)$，$\omega_1=\omega/2$。

② $c=0$，$s=0$。

③ 对于i从2到$n-1$，重复执行。

④ $s=\omega_{i-1}(\mathrm{mod}(i+1))$。

⑤ $s\triangleq s+c$。

⑥ $\omega_i=\omega_{i-1}/(i+1)$。

⑦ 若 $s>((i+1)/2)$，则 $k_i=s-(i+1)$，$c=1$。

⑧ 否则，$k_i=s$，$c=0$。

⑨ 返回 $SFE(k)=(k_{n-1}\cdots k_2k_1)$。

3.2 带符号阶乘展开式快速标量乘算法

将整数 k 的带符号阶乘展开式方法应用在计算椭圆曲线密码标量乘运算中，能够把式(4)的标量乘法运算 kP 变化成如下面式(5)的表示形式：

$$\begin{aligned}kP=\sum_{i=1}^{n-1}k_i(i!)P=k_1\cdot P+k_2\cdot(2!\cdot P)+\cdots+k_{n-1}\cdot\\((n-1)!\cdot P)=k_1\cdot P_1+k_2\cdot P_2+\cdots+k_{n-1}\cdot P_{n-1}\end{aligned}\tag{5}$$

式中，标量 k 的带符号阶乘展开形式的编码长度 n 须要满足条件 $0\le k\le n!$，且同时应有 $P_i=(i!)P$。

从而能够把大整数的标量乘运算进而转换成求解一组小标量而且是多基点的标量乘法运算的累加和的形式，也即转换成了多标量乘法运算[11]。而分基点 P_i 能够利用构造预计算表的方法给出，并且具体的计算构造算法如下面的算法 3 所示：

算法 3 预计算表 P_i 的生成算法

输入：n，P。

输出：预计算表 P_i。

① 计算 i 的二进制编码 $BIN(i)=(b_0b_1\cdots b_{h-1})$。

② 令 $P_1=P$。

③ 对于 i 从 2 到 $n-1$，重复执行。

④ 令 $P_i=O$，其中，O 为无穷远点。

⑤ 对于 l 从 $d-1$ 到 0，重复执行。

⑥ $P_i=2\cdot P_i$。

⑦ 若 $b_l=1$，则 $P_i=P_i+P_{i-1}$。

⑧ 返回 P_i。

基于SFE的椭圆曲线密码标量乘算法的详细描述过程则如下面所给算法 4 所示：

算法 4 基于 SFE 的标量乘算法

输入：k，P

输出：$Q=kP$

① 计算标量 k 的阶乘展开式形式 $SFE(k)=(k_{n-1}\cdots k_2k_1)$。

② 计算小标量 k_i 的二进制编码形式 $BIN(k_i)=(k_{i0}k_{i1}\cdots k_{i(l-1)})$。

③ 构造出预计算表 P_i。

④ 令 $Q=O$，其中，O 为无穷远点。

⑤ 对于 j 从 $m-1$ 到 0，重复执行：

⑥ $Q=2Q$。

⑦ 对于 i 从 $n-1$ 到 1，若 $k_{ij}=1$，则 $Q=Q+P_i$。

⑥ 返回 Q。

4 基于 SFEPH 标量乘快速算法

所给出的基于折半运算的带符号阶乘展开式标量乘快速算法，它的基本思路是首先需要对标量进行带符号的阶乘展开式形式的编码，然后利用预计算表和折半运算的方法，以此将标量乘运算转换成计算一组基于折半运算的小标量乘法运算。

则能够把运算 $kP=\sum_{i=1}^{n-1}k_i(i!)P$ 变化成式(6)：

$$\begin{aligned}kP=\sum_{i=1}^{n-1}k_i(i!)P=\sum_{i=1}^{n-1}\left[\sum_{j_i=0}^{m-1}2^{j_i}\cdot(i!\cdot P)\right]=\sum_{i=1}^{n-1}\left[\sum_{j_i=0}^{m-1}\left(\frac{1}{2}\right)^{j_i}\cdot PH_i\right]=\\\sum_{j_1=0}^{m-1}\left(\frac{1}{2}\right)^{j_1}\cdot PH_1+\sum_{j_2=0}^{m-1}\left(\frac{1}{2}\right)^{j_2}\cdot PH_2+\cdots+\sum_{j_{n-1}=0}\left(\frac{1}{2}\right)^{j_{n-1}}\cdot PH_{n-1}\end{aligned}\tag{6}$$

式中，m 为最长二进制编码的长度，在每一个小标量的折半编码表示长度 j_i 不足 m 位时，可以将余下的 (j_i-m) 位全部补充成 0。PH_i 为应用折半运算后的预计算表。

下面则给出了基于折半运算和带符号阶乘展开式的标量乘法的具体计算步骤：

① 最开始要先计算已给标量 k 的带符号阶乘展开形式的编码表示 $SFE(k)=(k_{n-1}\cdots k_2k_1)$。

② 根据所给算法 5 的预计算表构造算法，构造出预计算表 PH_i。

③ 将折半运算应用在每一个阶乘展开后的小标量 k_i 中，转化为能够实现折半运算的形式。

④ 将每一个基于折半运算的小标量乘法通过累加和的形式进行计算，即为所求得的返回结果。

预计算表 PH_i 的构造算法的具体描述过程如算法 5 所示：

算法 5 预计算表 PH_i 的生成算法

输入：n，P。

输出：预计算表 PH_i。

① 计算 i 二进制编码形式 $BIN(i)=(b_0b_1\cdots b_{h-1})$。

② 令 $P_1=P$。

③ 对于 i 从 2 到 $n-1$，重复执行。

④ 令 $P_i=O$，其中，O 为无穷远点。

⑤ 对于 l 从 $d-1$ 到 0，重复执行。

⑥ $P_l=P_l/2$。

⑦ 若 $b_l=1$，则有 $P_i=P_i+P_{l-1}$。

⑧ 返回 PH_i。

基于 SFEPH 的椭圆曲线标量乘快速算法的具体描述过程如算法 6 所示：

算法 6 基于 SFEPH 的标量乘快速算法

输入：k，P。

输出：$Q=kP$。

① 计算出标量 k 的阶乘展开式形式 $SFE(k)=(k_{n-1}\cdots k_2k_1)$。

② 计算出每个小标量 k_i 的折半运算形式二进制编码形式 $PH(k_i)=(l_0 l_1 \cdots l_{(m-1)})$。

③ 构造出预计算表 PH_i。

④ 令 $Q=O$，其中，O 为无穷远点。

⑤ 对于 j 从 $m-1$ 到 0，重复执行。

⑥ $Q=Q/2$。

⑦ 对于 i 从 $n-1$ 到 1，若 $l_j=1$，则 $Q=Q+PH_i$。

⑥ 返回 Q。

4 性能分析

算法6中，步骤3采用算法5中基于折半运算的小标量二进制编码标量乘预计算表构造算法，其运算量是 $(n-2)\left(d_{avg}H+\lfloor d_{avg}/2\rfloor A\right)$，其中，$d_{avg}$ 是平均执行的小标量乘法运算次数，$\lfloor d_{avg}/2\rfloor$ 是执行点加操作的次数。在⑤主循环运算中，令 h_{avg} 是小标量乘采用折半运算时平均执行循环操作次数，在步骤5.2中执行点加操作的平均次数为 $\lfloor n/2\rfloor$，则主循环运算所需运算量是 $h_{avg}\left(H+\lfloor n/2\rfloor A\right)$。

算法5中所需总运算量是 $\left[(n-2)d_{avg}+h_{avg}\right]H+\left[(n-2)\lfloor d_{avg}/2\rfloor+h_{avg}\lfloor n/2\rfloor\right]A$。

假设令 t_{avg} 是小标量乘利用传统二进制编码时平均执行循环操作的次数，则基于传统的带符号阶乘展开式标量乘算法所需运算量为

$$\left[(n-2)d_{avg}+t_{avg}\right]D+\left[(n-2)\lfloor d_{avg}/2\rfloor+t_{avg}\lfloor n/2\rfloor\right]A$$

式中，H 为折半运算，D 为倍点运算，A 为点加运算。

当前，通常认为椭圆曲线密码的密钥长度为512 bit的密钥是安全的，也即 $k=2^{512}$。由于 $k\in[0,n!]$，则有 $SFE(k)$ 编码长度 $n=100$，则可令 $d_{avg}=50$。且因 $|k_i|\le\lfloor(i+1)/2\rfloor$，有 $t_{\max}=6$，则令 $t_{avg}=3$。另外由于算法4与算法5中的小标量乘法系数相同，所以也有 $h_{avg}=3$。

文献[12]中给出了在仿射坐标系的条件下，倍点操作运算 $D=24M$，点加操作运算 $A=23M$。同时根据上述的第2小节可知，折半运算 $H=2M$。其中，M 为模乘运算。

所给算法6与传统的基于带符号阶乘展开式标量乘算法性能比较，见表1。

表1 算法6与传统带符号阶乘展开式标量乘算法的性能比较

Tab. 1 Comparison between algorithm 6 and the traditional scalar multiplication algorithm based on SFE

	预计算量	主运算量	总运算量	执行时间(min)	效率提高率%
传统SFE算法	$(n-2)(d_{avg}D+\lfloor d_{avg}/2\rfloor A)$	$t_{avg}(D+\lfloor n/2\rfloor A)$	$\left[(n-2)d_{avg}+t_{avg}\right]D+\left[(n-2)\lfloor d_{avg}/2\rfloor+t_{avg}\lfloor n/2\rfloor\right]A$	177 472	-
算法6	$(n-2)(d_{avg}H+\lfloor d_{avg}/2\rfloor A)$	$h_{avg}(H+\lfloor n/2\rfloor A)$	$\left[(n-2)d_{avg}+h_{avg}\right]H+\left[(n-2)\lfloor d_{avg}/2\rfloor+h_{avg}\lfloor n/2\rfloor\right]A$	69 606	60.78

从表1能够看出，所给算法6的运算效率比传统的基于带符号阶乘展开式标量乘快速算法提高了60.78 %左右。同时，能够计算得出算法6在预计算阶段和主循环计算阶段执行时间分别为173 950 M和3 522 M，而传统基于带符号阶乘展开式算法在预计算阶段和主循环计算阶段的执行时间，则分别为66 150 M和3 456 M，可见算法6在的运算提高率主要集中在预计算阶段，运算效率大约提升61.97 %，而在主循环运算阶段的运算效率相差不大，这主要是因为传统基于带符号阶乘展开式方法要计算出所需的预计算点，需要首先耗费大量时间构建一个较为庞大预计算表，而所给算法6能够有效提升这一过程，节省大量预计算所需的时间开销。由性能分析比较结果能够看出，所给的算法6可以有效提升基于带符号阶乘展开式得标量乘算法的运算效率，因而能够很好地应用在采用椭圆曲线密码的各类加密应用系统中，尤其是对于椭圆曲线密码算法执行时间要求较为敏感的各类资源受限的系统中。

5 结　论

标量乘法运算是提高椭圆曲线密码算法的关键，主要通过反复应用点加运算和倍点运算进行计算。分析传统基于阶乘展开式标量乘算法可知，采用新的标量编码方法和预计算提高标量乘运算效率，然而通过将折半运算应用到传统的基于阶乘展开式标量乘法运算中，利用折半运算替换倍点运算，同样在预计算过程中也使用折半运算，因折半运算比倍点运算的计算效率要高，因而所给算法在椭圆曲线密码标量乘算法中可以大幅提高运算效率。性能分析可知，所给算法很好应用各种加密应用系统中，具有很好的理论研究意义和实际推广应用价值。

参考文献(References)

[1] Koblitz N. Elliptic curves cryptosystems[J]. Mathematics of Compute, 1987, 48(177): 203-209.

[2] Miller V. Uses of elliptic curves in cryptography[C]. Proceedings of Advances in Cryptology Crypto'85, Berlin: Springer-Verlag, 1986: 417-426.

[3] Antao S, Bajard J, Sousa L. Elliptic curve point multiplication on gpus [C]. Proceedings of the 21th IEEE Int. Conf. on Application-Specific Systems, Architectures and Processors, 2010: 192-199.

[4] 王正义，赵俊阁. 基于带符号双基数系统的抗功耗攻击方案算法[J]. 计算机应用, 2011, 31(11):2973-2974.
Wang Z Y, Zhao J G. Resisting power analysis attack scheme based on signed double-based number system[J]. Journal of Computer Applications, 2011, 21(11): 2973-2974.

[5] 石润华，葛丽娜，钟诚. 椭圆曲线密码体制上的一种快速算法[J]. 计算机工程与科学, 2004, 26(4): 55-58.
Shi R H, Ge L N, Zhong C. A fast algotithm for wP in the elliptic curve cryptosystem[J]. Computer Engineering & Sience, 2004, 26(4): 55-58.

[6] Knudsen E. Elliptic scalar multiplication using point halving[C]. Advances in Cryptology-ASIACRYPT'99, New Youk: Springer-Verlag, 1999, 1716(274): 135-149.

[7] Purohit G N, Rawat S A, Kumar M. Elliptic curve point multiplication using MBNR and point halving[J]. International Journal of Advanced Networking and Applications, 2012, 3(5): 1329-1337.

[8] 陈辉，鲍皖苏. 基于半点运算与多基表示的椭圆曲线标量乘法[J]. 计算机工程, 2008, 34(15): 153-155.
Chen H, Bao W S. Elliptic curve scalar multiplication based on point halving and MBNS[J]. Computer Engineering, 2008, 34(15): 153-155.

[9] Fong K, Hankerson D, Lopez J, et al. Field inversion and point halving revisited[J]. IEEE Transactions on Computers, 2004, 53(8): 1047-1059.

[10] Morain F, Olivos J. Speeding up the computations on an elliptic curve using addition-subtraction chains[J]. Theoretical Informatics and Applications, 1990, 24(6): 120-126.

[11] 陈厚友，马传贵. 椭圆曲线密码中一种多标量乘算法[J]. 软件学报, 2011, 22(4): 782-788.
Chen H Y, Ma C G. A multiple scalar multiplications algotithm in the elliptic curve cryptosystem[J]. Journal of Software, 2011, 22(4): 782-788.

[12] Barua R, Pandey S K, Pankaj R. Efficient window-based scalar multiplication on elliptic curves using double- base number system [C]. Lecture Notes in Computer Science, Berlin: Springer-Verlag, 2007: 351-360.