椭圆曲线快速点乘算法优化*

周 梦，周海波
（北京航空航天大学 数学与系统科学学院，北京 100191）

摘 要：转换乘法为平方运算，是一种快速计算椭圆曲线密码点乘的代数方法。利用此方法，提出了素域 F_p 上雅可比坐标系下的 $3P$ 和 3^kP 算法，其运算量分别为 6[M]+10[S]和(6k)[M]+(10k)[S]，与已有的最好算法相比，算法效率分别提升了 11.8% 和 10.5%。另外，还在文献[1,2]基础上，对素域 F_p 上仿射坐标系下的 2^kP 和 3^kP 的算法进行了改进，其算法效率比文献[1,2]分别提高了 6.3% 和 3.3%。

关键词：椭圆曲线密码体制；点乘；仿射坐标；雅可比坐标；乘法
中图分类号：TP309 **文献标志码**：A **文章编号**：1001-3695(2012)08-3056-03
doi:10.3969/j.issn.1001-3695.2012.08.066

Optimizing fast scalar multiplication in elliptic curves

ZHOU Meng, ZHOU Hai-bo
(*School of Mathematics & System Science*, *Beihang University*, *Beijing* 100191, *China*)

Abstract: Trading field multiplications for field squaring is an algebraic method to improve the performance of scalar multiplication in ECC. This paper gave the algorithms of $3P$ and 3^kP over F_p in terms of Jacobian coordinates. And their computational complexity were 6[M]+10[S] and (6k)[M]+(10k)[S] respectively, which was improved to 11.8% and 10.5% respectively than the best algorithms at present. In addition, this paper improved the algorithms of 2^kP and 3^kP over F_p in terms of affine coordinates on the basis of literature [1,2]. And their computational complexity was improved to 6.3% and 3.3% respectively than literature [1,2].

Key words: elliptic curve cryptosystems(ECC); scalar multiplication; affine coordinate; Jacobian coordinate; multiplications

椭圆曲线密码系统(ECC)[3]是分别由 Miller 等人和 Koblitz 等人独立提出来的。该密码体制是基于椭圆曲线离散对数难解问题(ECDLP)，与其他公钥密码体制相比(如 RSA、DSA)，椭圆曲线密码具有如下技术优势[4]：

a)ECC 安全性能高。RSA 是基于大整数因子分解问题(IFP)，具有亚指数级的复杂度，而基于椭圆曲线离散对数难解问题(ECDLP)的 ECC 具有指数级的复杂度。图 1 是 ECC 和 RSA 的安全性对照图。从图 1 可知，在同样的攻破时间内，ECC 需要的密钥比 RSA 的要短得多，如 160 bit 的 ECC 和 1 024 bit 的 RSA 有相同的安全性。

b)ECC 实现性能优。ECC 的实现优势主要为实现成本低、存储占用空间小、宽带要求低、运算处理时间短、生成密钥对方便等优势。

因此，ECC 在信息安全中有广泛的用途，近些年来已普遍受到世界密码学专家、数学家和计算机科学专家的密切关注。而在椭圆曲线密码体制中，椭圆曲线群的点乘运算占了整个运算的很大比例，它决定了椭圆曲线加密协议的执行时间，因此通过代数技巧改进椭圆曲线的运算效率就具有重要意义。目前文献中，对提高运算效率的主要代数算法可以归纳为：

a)通过坐标变换，避免求逆。Mahdavi 等人[5]对点 P 进行坐标变换(标准投影坐标、Jacobian 坐标、改进的雅可比坐标等)，通过改变点的坐标形式，继而不用求逆，提高了运算效率。

b)求递推公式，减少求逆。Guajardo 等人[6]利用转换求逆为乘法的思想，提出了仿射坐标系下直接计算 $4P$、$8P$、$16P$ 的算法。Sakai 等人[1]在此基础上推导出了 F_p 上的 2^kP 的递推公式，将求逆转换为乘法，极大地提高了效率。刘连浩等人[2]也计算出了 F_p 上的仿射坐标下的 3^kP 的递推公式，大量减少了求逆运算。Dimitroy 等人[7]也提出了雅可比坐标下的直接计算 3^kP 的方法。

c)通过算法改进，转换求逆为乘法或平方，或转换乘法为平方。Eisentrager 等人[8]把 $2P+Q$ 分解为 $(P+Q)+P$ 运算，减少了一次乘法运算。Ciet 等人[9]利用分母的最小公倍数思想，计算仿射坐标系下 $3P+Q$，同样减少了一次求逆运算。之后刘连浩又对 $3P+Q$ 算法进行了改进，又减少了一次求逆运算。Joye[10]利用转换乘法为平方的思想，改进雅可比坐标下的点乘和倍点运算，分别将 1[M]转换为 1[S]和 2[M]转换为 2[S]。

本文利用文献[10]的转换乘法为思想，提出了素域 F_p 上雅可比坐标下的 $3P$ 和 3^kP 算法。

1 背景知识介绍

1.1 椭圆曲线的定义

定义 1 定义在域 K 上的 Weierstrass 方程为

收稿日期：2011-12-19；**修回日期**：2012-01-30 **基金项目**：国家自然科学基金资助项目(10871017)；北京市自然科学基金资助项目(1102026)

作者简介：周梦(1958-)，男，江西吉安人，教授，博导，主要研究方向为代数学及其应用、代数与符号计算、密码学理论(zm1613@sina.com)；周海波(1988-)，男，湖南常德人，硕士研究生，主要研究方向为信息安全.

$$E: y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{1}$$

其中:$a_1, a_2, a_3, a_4, a_6 \in K$,$K$ 为有限域,称 E 是域 K 上的椭圆曲线。在实际中,式(1)可以利用相容性变换进行简化。当 $\mathrm{char}(K) = 2$ 时,若 $a_1 \neq 0$,则 E 可以简化为

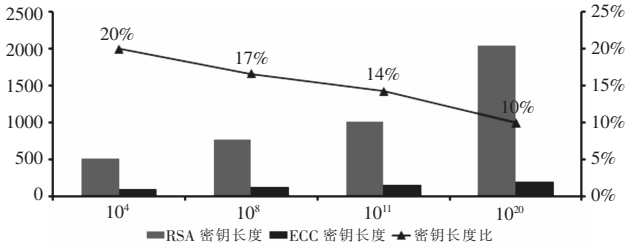
$$y^2 + xy = x^3 + ax^2 + b \tag{2}$$

其中:$a, b \in K, \Delta = b \neq 0$。

当 $\mathrm{char}(K) \neq 2, 3$ 时,E 可以简化为

$$y^2 = x^3 + ax + b \tag{3}$$

其中:$a, b \in K, \Delta = -16(4a^3 + 27b^4) \neq 0$。本文基于式(3)的情形进行讨论。



攻破时间	10^4	10^8	10^{11}	10^{20}	10^{78}
RSA 密钥长度	512	768	1024	2048	21000
ECC 密钥长度	106	132	160	210	600

图 1　RSA 和 ECC 的安全性对比

1.2　椭圆曲线的运算法则

$E(K)$ 表示 E 上的 K 有理点的集合,它关于"弦和切线"法则构成一个加法交换群,并且无穷远点 O 为其单位元。

椭圆曲线 E 的群运算是按照椭圆曲线群运算法则完成的,其中所涉及的运算有加、减、乘、求逆和平方五种基本运算,用 M、S、I 分别表示域 K 上的乘法、平方和求逆运算五种基本运算中,加法和减法相对其他运算来说,所用时间可以忽略不计。这里假设 1[I]≈8[M],1[S]≈0.6[M]。

1.3　仿射坐标下的运算法则

1.3.1　点加运算

令 $P = (x_1, y_1) \in E(K)$,$Q = (x_2, y_2) \in E(K)$,$P \neq \pm Q$,则 $P + Q = (x_3, y_3) \in E(K)$,从而

$$\begin{cases} x_3 = \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \\ y_3 = \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1 \end{cases} \tag{4}$$

仿射坐标下计算 $P + Q$ 所需运算量为 1[I] + 2[M] + 1[S]。

1.3.2　倍点运算

令 $P = (x_1, y_1) \in E(K)$,$P \neq -P$,则 $2P = (x_3, y_3) \in E(K)$,从而

$$\begin{cases} x_3 = \left(\dfrac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1 \\ y_3 = \left(\dfrac{3x_1^2 + a}{2y_1}\right)(x_1 - x_3) - y_1 \end{cases} \tag{5}$$

仿射坐标下计算 $2P$ 所需运算量为 1[I] + 2[M] + 2[S]。

1.4　雅可比坐标下的运算法则

令 $c = 2, d = 3$,则雅可比投影坐标 $P = (X, Y, Z)$ 与仿射坐标 $P' = (x, y) = (X/Z^2, Y/Z^3)$ 相对应。所以在雅可比坐标下的方程为:$Y^2 = X^3 + aXZ^4 + bZ^6$,则其运算法则如下所示。

1.4.1　点加运算

令 $P = (X_1, Y_1, Z_1) \in E$,$Q = (X_2, Y_2, Z_2) \in E$,$P \neq \pm Q$,则 $P + Q = (X_3, Y_3, Z_3)$,从而

$$\begin{cases} X_3 = u^2 - v^3 - 2X_1Z_2^2v^2 \\ Y_3 = u(X_1Z_2^2v^2 - X_3) - Y_1Z_2^3v^3 \\ Z_3 = vZ_1Z_2 \end{cases} \tag{6}$$

其中:$u = Y_2Z_1^3 - Y_1Z_2^3$;$v = X_2Z_1^2 - X_1Z_2^2$。通过存储中间结果:$A = Z_1^2$,$B = Z_2^2$,$C = Z_1A$,$D = Z_2B$,$u_1 = Y_1D$,$u_2 = Y_2C$,$u = u_2 - u_1$,$v_1 = X_1B$,$v_2 = X_2A$,$v = v_2 - v_1$,$E = v^2$,$F = vE$,$G = v_1E$,$I = u^2$,则 $X_3 = I - F - 2G$,$Y_3 = u(G - X_3) - Fu_1$,$Z_3 = vZ_1Z_2$,从而雅可比坐标下计算 $P + Q$ 所需的运算量为 12[M] + 4[S]。

1.4.2　倍点运算

令 $P = (X_1, Y_1, Z_1) \in E$,$P \neq -P$,则 $2P = (X_3, Y_3, Z_3)$,从而

$$\begin{cases} X_3 = (3X_1^2 + aZ_1^4)^2 - 8X_1Y_1^2 \\ Y_3 = (3X_1^2 + aZ_1^4) \times (4X_1Y_1^2 - X_3) - 8Y_1^4 \\ Z_3 = 2Y_1Z_1 \end{cases} \tag{7}$$

通过存储中间结果:$A = X_1^2$,$B = Z_1^2$,$C = Y_1^2$,$D = AB^2$,$u = 3A + D$,$E = 4X_1C$,则 $X_3 = u - 2E$,$Y_3 = u(E - X_3) - 8C^2$,$Z_3 = 2Y_1Z_1$,从而雅可比坐标下计算 2P 所需运算量为 4[M] + 6[S]。

文献[10]利用转换乘法为平方的思想,即令 $Z_3 = (Y_1 + Z_1)^2 - Y_1^2 - Z_1^2$,$4X_1Y_1^2 = 2[(X_1 + Y_1^2)^2 - X_1^2 - Y_1^4]$,则避免了两个乘法的计算,但同时多了两个平方运算,故 $2P$ 的运算量减少到了 2[M] + 8[S]。

2　算法改进

2.1　雅可比坐标下 $3P$、3^kP 算法改进

令 $3P = (X_3, Y_3, Z_3)$,则由式(1)(2)可得仿射坐标下的 $3P = (X'_3, Y'_3, 1)$,即

$$\begin{cases} X_3' = \dfrac{8Y_1^2(D - C) + X_1B^2}{(Z_1B)^2} \\ Y_3' = \dfrac{Y_1[4(C - D)(2D - C) - B^3]}{(Z_1B)^3} \end{cases}$$

其中:$A = 3X_1^2 + aZ_1^4$;$B = 12X_1Y_1^2 - A^2$;$C = AB$,$D = 8Y_1^4$。取 $Z_3 = Z_1B$,从而求得 $3P = (X_3, Y_3, Z_3)$ 为

$$\begin{cases} X_3 = 8Y_1^2(D - C) + X_1B^2 \\ Y_3 = Y_1[4(C - D)(2D - C) - B^3] \\ Z_3 = Z_1B \end{cases}$$

其中:$A = 3X_1^2 + aZ_1^4$;$B = 12X_1Y_1^2 - A^2$;$C = AB$,$D = 8Y_1^4$。通过存储中间结果,可知该算法的运算量为 10[M] + 6[S]。

为了应用转换乘法为平方的思想,由仿射坐标转换为雅可比坐标时,取 $Z_3 = 2Z_1B$,从而

$$\begin{cases} X_3 = 16Y_1^2(2D - 2C) + 4X_1B^2 \\ Y_3 = 8Y_1[(2C - 2D)(4D - 2C) - B^3] \\ Z_3 = (Z_1 + B)^2 - Z_1^2 - B^2 \end{cases}$$

其中:$A = 3X_1^2 + aZ_1^4$;$B = 12X_1Y_1^2 - A^2$;$2C = 2AB$,$2D = 16Y_1^4$。

这时,令 $A = 3X_1^2 + aZ_1^4$,$B = 6[(X_1 + Y_1^2) - X_1^2 - Y_1^4] - A^2$,$2C = (A + B)^2 - A^2 - B^2$,$2D = 16Y_1^4$,则将四个乘法运算转换成了平方运算。通过存储中间结果,可知该算法的运算量为:6[M] + 10[S],相比原算法的 10[M] + 6[S],该算法效率提升了 11.8%。

用此算法计算 3^kP 所花费的时间为:$(6k)[M]+(10k)[S]$,与文献[7]的结果$(11k-1)[M]+(4k+2)[S]$相比,增加了$(6k-2)[S]$,减少了$(4k-1)[M]$,算法效率提升了10.5%。

2.2 仿射坐标下 2^kP 的算法改进

令 $P_1=(x_1,y_1)\in E(K)$,$P_{2k}=2^kP_1=(x_{2^k},y_{2^k})\in E(F_p)$,利用转换乘法为平方的思想和新算法,对文献[1]的算法进行改进。在改进 2^kP 算法之前,先推导出 $2P$、$4P$、$8P$、$16P$ 的改进算法。

2.2.1 计算 $2P$

本文采用先直接求出分母中公因式 $2C_1$ 的逆,再采取平方和乘法运算,改进算法如下:

令 $A_1=x_1$,$B_1=3x_1^2+a$,$C_1=-y_1$,$D_1=12A_1C_1^2-B_1^2$,$E_1=(2C_1)^{-1}$,$x_2=(B_1^2-8A_1C_1^2)\times(E_1)^2$,$y_2=(8C_1^4-B_1D_1)\times(E_1)^2\times E_1$。该算法的运算量为:$1[I]+5[M]+5[S]$,与文献[1]结果 $1[I]+6[M]+4[S]$ 相比,增加了 $1[S]$,减少了 $1[M]$,算法效率提升了2.4%。

2.2.2 计算 $4P$

由于 A_1^2、C_1^2 和 C_1^4 都是已知项,这里利用转换乘法为平方的思想,用$(A_1+C_1^2)^2-A_1^2-C_1^4$ 来替换原来算法中的 $2A_1C_1^2$,这样就用 $1[S]$ 替换了 $1[M]$,减少了运算量。而且可先对分母的公因式 $4C_1C_2$ 求逆,再采取乘法和平方运算。由于在计算 $2C_1C_2$ 时,已知 C_1^2 和 C_2^2,从而可令 $4C_1C_2=2[(c_1+c_2)^2-c_1^2-c_2^2]$,这样就用 $1[S]$ 替换了 $1[M]$。改进算法如下:

令 $A_2=B_1^2-4[(A_1+C_1^2)^2-A_1^2-C_1^4]$,$B_2=3A_2^2+16aC_1^4$,$C_2=-8C_1^4-B_1(A_2-4A_1C_1^2)$,$D_2=12A_2C_2^2-B_2^2$,$E_2=\{2[(c_1+c_2)^2-c_1^2-c_2^2]\}^{-1}$,$P_4=(x_4,y_4)$有

$$\begin{cases}x_4=(B_2^2-8A_2C_2^2)\times(E_2)^2\\ y_4=(8C_2^4-B_2D_2)\times(E_2)^2\times E_2\end{cases}$$

从而该算法的运算量为:$1[I]+7[M]+11[S]$,与文献[1]结果 $1[I]+10[M]+9[S]$ 相比,增加了 $2[S]$,减少了 $3[M]$,算法效率提升了7.7%。

2.2.3 计算 $8P$

同样地,利用上述思想,但此时由于要求逆的部分为 $8C_1C_2C_3$,不能再转换一个乘法为平方。从而改进算法如下:

令 $A_3=B_2^2-4[(A_2+C_2^2)^2-A_2^2-C_2^4]^2$,$B_3=3A_3^2+256aC_1^4C_2^4$,$C_3=-8C_2^4-B_2(A_3-4A_2C_2^2)$,$D_3=12A_3C_3^2-B_3^2$,$E_3=(8C_1C_2C_3)^{-1}$,则 $P_8=(x_8,y_8)$有

$$\begin{cases}x_8=(B_3^2-8A_3C_3^2)\times(E_3)^2\\ y_8=(8C_3^4-B_3D_3)\times(E_3)^2\times E_3\end{cases}$$

该算法的运算量为:$1[I]+11[M]+15[S]$,与文献[1]结果 $1[I]+14[M]+13[S]$ 相比,增加了 $2[S]$,减少了 $3[M]$,算法效率提升了6.0%。

2.2.4 计算 $16P$

令 $A_4=B_3^2-4[(A_3+C_3^2)^2-A_3^2-C_3^4]^2$,$B_4=3A_4^2+4096aC_1^4C_2^4C_3^4$,$C_4=-8C_3^4-B_3(A_4-4A_3C_3^2)$,$D_4=12A_4C_4^2-B_4^2$,$E_4=(16C_1C_2C_3C_4)^{-1}$,则 $P_{16}=(x_{16},y_{16})$有

$$\begin{cases}x_{16}=(B_4^2-8A_4C_4^2)\times(E_4)^2\\ y_{16}=(8C_4^4-B_4D_4)\times(E_4)^2\times E_4\end{cases}$$

该算法的运算量为:$1[I]+14[M]+20[S]$,与文献[1]结果 $1[I]+18[M]+17[S]$ 相比,增加了 $3[S]$,减少了 $4[M]$,算法效率提升了7.1%。

2.3 计算 2^kP 算法

在仿射坐标系下直接计算 2^kP,其中 $k\geqslant1$ 且 $P\in E(F_p)$。其具体算法如下。

算法1 仿射坐标下改进的 2^kP 算法

输入:$P_1=(x_1,y_1)\in E(F_p)$;

输出:$P_{2k}=2^kP_1=(x_{2^k},y_{2^k})\in E(F_p)$。

a)计算 A_1、B_1 和 C_1:

$$A_1\leftarrow x_1,B_1\leftarrow 3x_1^2+a,\ C_1\leftarrow -y_1$$

b)i 从 2 到 k 循环计算 A_i、B_i 和 C_i:

$$A_i\leftarrow B_{i-1}^2-4[(A_{i-1}+C_{i-1}^2)^2-A_{i-1}^2-C_{i-1}^4]^2$$

$$B_i\leftarrow 3A_i^2+16^{i-1}a(\prod_{j=1}^{i-1}C_j)^4$$

$$C_i\leftarrow -8C_{i-1}^4-B_{i-1}[A_i-2[(A_{i-1}+C_{i-1}^2)^2-A_{i-1}^2-C_{i-1}^4]^2]$$

c)计算 D_k、E_k:

$$D_k\leftarrow 12A_kC_k^2-B_k^2,E_k\leftarrow(2^k\prod_{i=1}^{k-1}C_i)^{-1}$$

d)计算 x_{2^k}、y_{2^k}:

$$x_{2^k}\leftarrow(B_k^2-8A_kC_k^2)\times(E_k)^2,y_{2^k}\leftarrow(8C_k^4-B_kD_k)\times(E_k)^2\times E_k$$

该算法的运算量为:$1[I]+(3k+2)[M]+(5k)[S]$,与文献[1]结果 $1[I]+(4k+2)[M]+(4k+1)[S]$ 相比,增加了 $(k-1)[S]$,减少了 $k[M]$,算法效率提升了6.3%。

2.4 仿射坐标下 3^kP 算法改进

令 $P_1=(x_1,y_1)\in E(K)$,$P_{3^k}=3^kP_1=(x_{3^k},y_{3^k})\in E(F_p)$,由于2.3节已经给出了 2^kP 的详细改进过程,因此不再赘述详细的推导过程,而直接给出文献[2]的改进算法。

算法2 仿射坐标下改进的 3^kP 算法

输入:$P=(x_1,y_1)\neq0$;

输出:$3^k=(x_{3^k},y_{3^k})$。

a)计算 A_0、B_0、C_0:

$$A_0\leftarrow x_1,B_0\leftarrow y_1,C_0\leftarrow 1$$

b)计算 T_i、N_i、D_i、A_i、B_i、C_i:

$$T_i\leftarrow 8B_{i-1}^4$$

$$N_i\leftarrow 3B_{i-1}^4+aC_{i-1}^4$$

$$D_i\leftarrow 12A_{i-1}B_{i-1}^2-N_i^4$$

$$A_i\leftarrow 4B_{i-1}^2[2T_i-(N_i+D_i)^2+N_i^2+D_i^2]+A_{i-1}D_i^2$$

$$B_i\leftarrow B_{i-1}[4(N_iD_i-T_i)(2T_i-N_iD_i)-D_i^3]$$

$$C_i\leftarrow D_iC_{i-1}$$

c)计算 x_{3^k}、y_{3^k}:

$$x_{3^k}\leftarrow A_k/C_k^2,y_{3^k}\leftarrow B_k/C_k^3$$

该算法的运算量为:$1[I]+(7k+2)[M]+(8k-1)[S]$,与文献[2]的结果 $1[I]+(8k+2)[M]+(7k-1)[S]$ 相比,增加了 $k[S]$,减少了 $k[M]$,算法效率提升了3.3%。

3 结束语

本文利用转换乘法为平方的思想,研究特征不等于2和3的域上的椭圆曲线,给出了雅可比坐标下 $3P$ 和 3^kP 的算法,与文献[7]的结果相比,分别将 $4[M]$ 转换成了 $4[S]$ 和 $(4k-1)[M]$ 转换成了 $(6k-2)[S]$,运算效率分别提升了11.8%和10.5%。另外,利用此思想和新方法计算了仿射坐标下的 $2P$、$4P$、$8P$、$16P$,比文献[1]的算法效率分别提升了2.4%、7.7%、6.0%、7.1%。同时,利用相同的原理,本文给出了仿射坐标下计算 2^kP 和 3^kP 的改进算法,计算时间与文献[1,2]相比,计算效率分别提升了6.3%和3.3%。 (下转第3061页)

```
    return Area(cid + cid_X)
    else
    return CLOAKING(k, A_min, (cid + cid_Y))
  end if
  if (cid.N ≥ k and cid.Area < A_min)
    return CLOAKING(k, A_min, (cid + cid_Y))
  end if
end function
```

4　实验结果与分析

该算法采用 Java 实现,在处理器 Intel Core 2 Duo T6400 2.0 GHz、内存 2 GB 的平台上运行。实验采用著名的 Thomas Brinkhoff 基于路网的移动对象生成器[8],以城市 Oldenburg 的交通路网作为输入,生成输入城市路线图上移动的移动对象,目标对象均匀分布在空间。本实验取 5 000 个注册移动用户,为每个用户随机产生一个隐私协议,k 取值范围为[1,50],A_{min} 取值范围[1,50],本实验取 A_{min} 的单位为一个单元格。

为验证本文方法的有效性,实验将 Bottom-up 匿名算法和本文所提出的 Grid-divide 位置匿名算法使用相同的数据集进行比较。实验对算法的匿名时间和匿名区域两个指标进行评测。匿名时间是指触发查询提出请求到匿名成功的时间。取 $A_{min}=4$ 不变,如图 3 所示,匿名时间随着 k 值的增加而增长,这是因为随着 k 值的增加,每一个用户位置需要更多的时间处理、等待才能匿名成功。匿名区域是指用户位置匿名成功后得到的空间区域的大小。取 $A_{min}=2$ 不变,如图 4 所示,匿名区域随着 k 值的增加而增长,这是因为随着 k 值的增加,需要更大的空间才能满足用户的隐私需求。由图 4 中可以看到随着 k 值的增加,本文提出的 Grid-divide 位置匿名算法得到的匿名区域明显减小,因为在用户当前所在网格不符合条件的情况下,Bottom-up 匿名算法递归返回四倍于当前网格的父网格,而改进后的 Grid-divide 位置匿名算法递归合并和当前网格同样大小的邻居网格。由这两个指标可以看出,本文提出的算法可以有效减少匿名时间,减小匿名空间区域,从而为用户查询的服务质量提供了保障。

5　结束语

本文针对位置服务中的用户位置隐私保护问题,采用网格结构划分空间方法,在 Bottom-up 匿名算法的基础上进行改进。实验结果表明,改进后的算法匿名时间更短,匿名区域较小,从而有利于提高用户的位置查询服务质量。

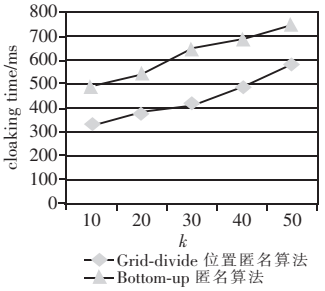


图 3　本文算法与 Bottom-up 匿名算法匿名时间比较

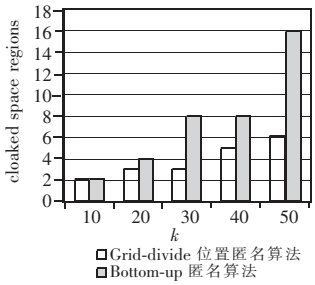


图 4　本文算法与 Bottom-up 匿名算法匿名区域比较

参考文献:

[1] MOKBEL M F, CHOW C Y, AREF W G. The new casper: ouery processing for location services without compromising privacy[C]//Proc of the 32nd International Conference on Very Large Data Bases. 2006: 763-774.

[2] CHOW C, MOKBEL M F. Enabling privacy continuous queries for revealed user location[C]//Proc of International Symposium on Advances in Spatial and Temporal Databases. Berlin: Springer-Verlag, 2007.

[3] 潘晓,肖珍,孟小峰. 位置隐私研究综述[J]. 计算机科学与探索, 2007, 1(3):268-281.

[4] SWEENEY L. K-anonymity: a model for protecting privacy[J]. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 2002, 10(5):557-570.

[5] GRUTESER M, GRUNWAL D. Anonymous usage of location-based services through spatial and temporal cloaking[C]//Proc of International Conference on Mobile Systems, Applications, and Services. New York: ACM Press, 2003:163-168.

[6] GEDIK B, LIU L. Location privacy in mobile systems: a personalized anonymization model[C]//Proc of International Conference on Distributed Computing System. 2005:620-629.

[7] XIAO Z, MENG X, XU J. Quality-aware privacy protection for location-based services[C]//Proc of International Conference on Database System for Advanced Applications. 2007.

[8] BRINKHOFF T. A framework for generating network-based moving objects[J]. GeoInformatica, 2002, 6(2):153-180.

(上接第 3058 页)

参考文献:

[1] SAKAI Y, SAKURAI K. Efficient scalar multiplications on elliptic curves without repeated doublings and their practical performance [C]//Proc of the 5th Australasian Conference. 2000:59-73.

[2] 刘连浩,申勇. 椭圆曲线密码体制中标量乘法的快速算法[J]. 计算机应用研究, 2009, 26(3):1104-1108.

[3] KOBLITZ N. Elliptic curve cryptosystems[J]. Mathematics of Computation, 1987, 48(177):203-209.

[4] 旺朝晖. ECC 在信息安全领域内的应用现状[J]. 中国集成电路, 2010, 19(6):22-26.

[5] MAHDAVI R, SAIADIAN A. Efficient scalar multiplications for elliptic curve cryptosystems using mixed coordinates strategy and direct computations[C]//Lecture Notes in Computer Science, vol 6467. 2010:184-198.

[6] GUAJARDO J, PAAR C. Efficient algorithms for elliptic curve cryptosystems[C]//Proc of the 17th Annual International Cryptology Conference. Berlin: Springer-Verlag, 1997:342-356.

[7] DIMITROY V, IMBERT L, MISHRA P K. Efficient and secure elliptic curve point multiplication using double-base chains[C] //Proc of ASIACRYPT. Berlin: Springer-Verlag, 2005:59-78.

[8] EISENTRAGER K, LAUTER K, MONTGOMERY P L. Fast elliptic curve arithmetic and improved weil pairing evaluation[C]//Proc of RSA Conference on the Cryptographers' Track. 2003:343-354.

[9] CIET M, JOYE M, LAUTER K, *et al*. Trading inversions for multiplications in elliptic curve cryptography[J]. Designs Codes and Cryptography, 2006, 39(2):189-206.

[10] JOYE M. Fast point multiplication on elliptic curves without precomputation[C]//Proc of the 2nd International Workshop on Arithmetic of Finite Fields. 2008:36-46.