Bisección-y-adición para curvas supersingulares de género 2 en característica 2

Ricard Garra, Josep M. Miret, Jordi Pujolàs
Dept. de Matemàtica
Universitat de Lleida
Email: {rgarra, miret, jpujolas}@matematica.udl.cat

Thomaz Oliveira
Instituto de Computação
Universidade Estadual de Campinas, Campinas, Brasil
Email: thomaz.oliveira@students.ic.unicamp.br

***Resumen*—Damos un método para el cálculo de múltiplos kD de divisores en Jacobianas de curvas supersingulares de género 2 en característica 2 basado en algoritmos de bisección. Nuestro método es competitivo comparado con los tradicionales de duplicación-y-adición, y también comparado con los análogos para curvas no supersingulares en característica 2.**

I. Introducción

En el contexto de criptografía basada en el problema del logaritmo discreto, la operación fundamental es el cálculo de múltiplos escalares de elementos del grupo donde se sustenta el sistema. Para que esta operación sea eficiente, el método más común es el basado en sumas y duplicaciones (el llamado algoritmo de duplicación-y-adición), que usa la representación binaria del escalar por el que se multiplica.

En criptografía de curva elíptica sobre cuerpos binarios, Knudsen [7] y Schroeppel [10] propusieron un método alternativo que sustituye las duplicaciones de puntos por bisecciones. El método, llamado de bisección-y-adición, resultó ser igual o más eficiente que el de duplicación-y-adición (véase un análisis detallado de Fong, Hankerson, López y Menezes en [5]).

Para curvas de género 2 sobre cuerpos binarios, la alternativa al algoritmo de duplicación-y-adición adquiere más interés dado que el algoritmo de Cantor [3] para doblar divisores es menos eficiente que el correspondiente para curvas elípticas. Algoritmos de bisección de divisores en género 2 se pueden encontrar en [6], [1] para característica par y en [9] en característica impar.

El caso de las curvas supersingulares en característica 2 es especialmente simple dado que, al ser la 2-torsión trivial, cada divisor tiene una única bisección. En este trabajo damos una versión del método de bisección-y-adición adaptado a estas curvas, y mostramos que es mucho más eficiente que para curvas no supersingulares.

II. Preliminares

Sea $q = 2^m$ para algún natural m, y sea C una curva de género 2 definida sobre $\mathbb{F}_q$ con un punto en el infinito $P_{\infty|\mathbb{F}_q}$, de modo que C tiene un modelo imaginario

$$\mathrm{C} : y^2 + h(x)y = f(x),$$

donde

$$f(x) = f_5x^5 + f_4x^4 + ... + f_0 \in \mathbb{F}_q[x],\ \deg(f) \leq 5,$$
$$h(x) = h_2x^2 + h_1x + h_0 \in \mathbb{F}_q[x],\ h(x) \neq 0.$$

Los elementos del grupo $\mathrm{Jac}(\mathrm{C})(\mathbb{F}_q)$ son divisores de grado 0 módulo divisores principales, y se pueden representar en coordenadas de Mumford por:

$$D = \big(u(x), v(x)\big), \quad u(x), v(x) \in \mathbb{F}_q[x],\ \deg v(x) < \deg u(x).$$

Para un divisor reducido de peso dos $D = P_1 + P_2 - 2P_\infty$, $P_i \in \mathrm{Jac}(\mathrm{C})(\overline{\mathbb{F}_q})$, las coordenadas de Mumford son de la forma

$$D = (x^2 + u_1x + u_0, v_1x + v_0),$$

donde las raíces de $u(x)$ son las abscisas de los puntos P_i, mientras que un divisor reducido de peso uno $P_1 - P_\infty$ tiene coordenadas de Mumford $(x + u_0, v_0)$. Los coeficientes u_1, u_0, v_1, v_0 pertenecen a $\mathbb{F}_q$, y la segunda coordenada interpola las coordenadas (x_i, y_i) de los puntos $P_i \in \mathrm{C}(\overline{\mathbb{F}_q})$ del soporte del divisor. Notemos que x_1, x_2 pueden pertenecer a $\mathbb{F}_{q^2}$.

La primera coordenada de los divisores de orden 2 de $\mathrm{Jac}(\mathrm{C})(\mathbb{F}_q)$ está formada por factores de $h(x)$. De hecho, el rango del grupo de 2-torsión $\mathrm{Jac}(\mathrm{C})(\mathbb{F}_q)[2]$ sigue la distribución siguiente:

Tipo de factorización de $h(x)$	2-rango
$[0]$	0
$[1]$	1
$[2]$	1
$[1^2]$	1
$[1, 1]$	2

Por lo que se refiere al uso criptográfico de estas curvas, las variedades Jacobianas deben tener en su cardinal un factor primo grande y un cofactor pequeño. Estas condiciones permiten que el problema del logaritmo discreto (DLP) sea lo suficientemente resistente y pueda sustentar los protocolos.

En este trabajo nos restringimos al caso en que el grupo de 2-torsión es trivial tomando $h(x) = h_0$. Esto implica que la multiplicación por 2 es biyectiva. A pesar de que estas curvas tienen Jacobianas supersingulares (y por lo tanto existen ataques específicos al DLP que sustentan), se consideran adecuadas, en lo referente a su seguridad, para protocolos basados en pairings [8, pg. 173].

III. Bisección

Dado un divisor $D_2 \in \text{Jac(C)}(\mathbb{F}_q)$, queremos encontrar un divisor $D_1 \in \text{Jac(C)}(\mathbb{F}_q)$ tal que

$$2D_1 = D_2,$$

donde

$$D_2 = (x^2 + u_{21}x + u_{20}, v_{21}x + v_{20}),$$
$$D_1 = (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10}).$$

Según [6], si deshacemos el paso de reducción en el algoritmo de duplicación, llegamos a una igualdad entre coordenadas de divisores no reducidos. En el caso de $2D_1$ estas coordenadas son de la forma:

$$((x^2 + u_{11}x + u_{10})^2, s_3x^3 + s_2x^2 + s_1x + s_0)$$

donde los s_i son coeficientes indeterminados. Para determinarlos, notemos que si $D_1 = P_1 + Q_1 - 2P_\infty$ entonces $2D_1 = 2P_1 + 2Q_1 - 4P_\infty$. De aquí podemos deducir que

$$s_3 = \frac{u_{11}^2 + f_3}{h_0},$$

$$s_2 = \frac{u_{11}^4 + u_{11}^2u_{10} + u_{10}^2 + f_3(u_{11}^2 + u_{10}) + v_{11}h_0 + f_1}{u_{11}h_0},$$

$$s_1 = \frac{u_{10}^2 + f_1}{h_0},$$

$$s_0 = \frac{u_{11}^2u_{10}^2 + u_{10}^3 + u_{10}^2f_3 + u_{10}f_1 + u_{10}v_{11}h_0}{u_{11}h_0} + v_{10}.$$

El divisor no reducido $(u_2'(x), v_2'(x))$ correspondiente a D_2 se obtiene usando un polinomio auxiliar $k(x) = k_1x + k_0$. Sus coordenadas son:

$$u_2'(x) = \frac{f(x) + h_0v_2(x) + v_2(x)^2}{u_2(x)} + h_0k(x) + k(x)^2u_2(x),$$
$$v_2'(x) = v_2(x) + h_0 + k(x)u_2(x).$$

Igualando la primeras componentes de los divisores no reducidos de la igualdad inicial $2D_1 = D_2$ obtenemos

$$u_{11} = \sqrt{k_1^2u_{20} + k_0^2 + f_4 + u_{21}}/k_1,$$
$$u_{10} = \sqrt{k_0h_0 + k_0^2u_{20} + c_0}/k_1$$

donde

$$k_1 = \sqrt{1/u_{21}},$$
$$k_0 = \sqrt{(k_1h_0 + c_1)/u_{21}},$$
$$c_1 = f_3 + u_{20} + (f_4 + u_{21})u_{21},$$
$$c_0 = f_2 + v_{21}^2 + (f_4 + u_{21})u_{20} + u_{21}c_1.$$

Para la segunda componente de Mumford de D_1, igualando s_2 y s_0 en $v_2'(x)$ encontramos que

$$v_{11} = \frac{u_{11}^4 + u_{11}^2u_{10} + u_{10}^2 + f_3(u_{11}^2 + u_{10}) + f_1}{h_0} + (k_1u_{21} + k_0)u_{11},$$

$$v_{10} = \frac{u_{11}^2u_{10}^2 + u_{10}^3 + u_{10}^2f_3 + u_{10}f_1}{u_{11}h_0} + \frac{u_{10}v_{11}}{u_{11}} + k_0u_{20} + v_{20} + h_0,$$

y así se evita tener que calcular el polinomio interpolador.

IV. Multiplicación por escalares

Revisamos en esta sección los métodos de duplicación-y-adición y de bisección-y-adición en un grupo finito aditivo G.

IV-A. Duplicación-y-adición

Sea n el orden del grupo G. Dado un elemento $D \in G$ y un entero k, $0 \le k < n$, para el cálculo de kD se busca la representación binaria del escalar k. El número esperado de dígitos con el valor 1 en la representación binaria de k es $t/2$, donde

$$t = \lfloor \log_2 n \rfloor + 1.$$

De este modo, el tiempo aproximado usado en el cálculo de kP es

$$(t/2)A + tB,$$

donde A es el tiempo empleado en una adición y B el de una duplicación.

Una alternativa para mejorar este tiempo se basa en una representación diferente de k denominada Forma No Adyacente [11]:

$$\text{NAF}(k) = \sum_{i=0}^{\ell-1} k_i 2^i, \quad k_i \in \{0, 1, -1\},$$

dotada de la propiedad de no tener dos coeficintes k_i consecutivos distintos de 0. Esta forma se generaliza a la llamada Forma No Adyacente con longitud ω, $\text{NAF}_\omega(k)$, en la que cada k_i no nulo satisface

- k_i es impar y $|k_i| < 2^{\omega-1}$,
- en cada secuencia de ω dígitos a lo sumo hay uno que es distinto de cero.

Observamos que $\text{NAF}_2(k)$=NAF(k), que los k_iD pueden ser precalculados y que el tiempo de cómputo de kD es inversamente proporcional a ω.

IV-B. Bisección-y-adición

El cálculo de kP mediante bisecciones en vez de duplicaciones requiere una representación distinta del escalar k. Sean $t = \lfloor \log_2 n \rfloor + 1$ y $k' = 2^t k \pmod n$.

Si la representación NAF_ω de k' es

$$\text{NAF}_\omega(k') \sum_{i=0}^{t} k_i' 2^i, \quad |k_i| < 2^{\omega-1},$$

entonces

$$k = \sum_{i=0}^{t} \frac{k_{t-i}'}{2^i} \pmod n = \frac{k_0'}{2^t} + \cdots + \frac{k_{t-1}'}{2} + k_t' \pmod n.$$

Por lo tanto obtenemos la siguiente representación de kD en términos de bisecciones:

$$kP = \frac{k_0'}{2^t}D + \cdots + \frac{k_{t-1}'}{2}D + k_t'D.$$

Damos a continuación el algoritmo de cálculo de kD mediante bisecciones.

Algoritmo 1. ***Bisección-y-adición.***

Input*: Un elemento $D \in G$, un entero k y la representación NAF con longitud ω de $k' = 2^t k = \sum_{i=0}^{t} k'_{t-i} 2^i$.*

Output*: El elemento kD.*

1. *calcular $D_i = iD$, para $i \in \{1, 3, 5, \ldots, 2^{\omega-1} - 1\}$*

2. $D' \leftarrow \mathcal{O}$

3. *para i desde 0 hasta t hacer*

3.1 $D' \leftarrow D'/2$

3.2 *si* $k'_i > 0$ *entonces* $D' \leftarrow D' + D_{k'_i}$

3.3 *si* $k'_i < 0$ *entonces* $D' \leftarrow D' - D_{-k'_i}$

4. *devolver*(D')

En nuestro caso las bisecciones $D'/2$ del paso 3.1 se obtienen mediante las fórmulas de la sección anterior.

En cuanto a la complejidad del Algoritmo 1, cabe señalar que en la parte relativa a la bisección, la extracción de las raíces cuadradas no es el paso más caro computacionalmente, ya que calcular una raíz cuadrada mediante el método descrito en [5] y [4, pg. 228] cuesta aproximadamente la mitad que una multiplicación en el cuerpo base. En [5] también se puede encontrar un estudio detallado del costo de usar representaciones NAF.

Por otra parte, en [11] se muestra que los métodos τ-ádicos son mejores que los de bisección en el contexto más específico de las curvas de Koblitz.

V. Un ejemplo de muestra

Sea C la curva de de género 2 definida sobre el cuerpo $\mathbb{F}_q$, $q = 2^3$, de ecuación:

$$\mathrm{C} : y^2 + y = x^5 + x^3 + a^5 x^2$$

La variedad Jacobiana de esta curva tiene cardinal

$$n = \#\mathrm{Jac}(\mathrm{C})(\mathbb{F}_q) = 81.$$

Consideremos el divisor

$$D = (x^2 + a^4 x + a, a^3 x + a).$$

Para calcular kD, con $k = 13$, mediante el algoritmo de bisección-y-adición, necesitamos la representación NAF de k. Se calcula $t = \lfloor \log_2 n \rfloor + 1 = 7$ y entonces

$$k' = 2^t k = [-1, 0, -1, 0, 1, 0] = \frac{-1}{32} + \frac{-1}{8} + \frac{1}{2}.$$

Por lo tanto, usando las expresiones de la bisección obtenemos

$$13D = \frac{-1}{32} D + \frac{-1}{8} D + \frac{1}{2} D = (x^2 + a^2 x, a^4 x).$$

El cálculo de $13D$ se podría hacer también usando la representación NAF de k con amplitud $\omega > 2$.

VI. Resultados

Hemos ejecutado el algoritmo anterior en MAGMA [2]. Para las curvas no supersingulares (aquellas con $h(x)$ no constante) hemos usado los algoritmos de bisección de [6]. Para las supersingulares hemos usado las fórmulas de la sección III. Las raíces de polinomios cuadráticos han sido calculadas mediante el método de Half-Trace (ver [5]).

Con el objetivo de comparar el algoritmo de bisección propuesto con el dado en [6] para curvas no supersingulares (el cardinal de cuyas Jacobianas es un múltiplo de 2), hemos lanzado 1000 bisecciones en los dos casos para curvas definidas sobre cuerpos binarios de 80 bits y el promedio de tiempos obtenido ha sido el siguiente:

$h(x)$ curva	Tiempo
$h_2 \neq 0$ o $h_1 \neq 0$	2.17
$h_2 = h_1 = 0$	0.51

Por lo tanto, hemos verificado que el método de bisección propuesto en este trabajo para curvas de género 2 con $h_2 = h_1 = 0$ sobre cuerpos de característica 2 es unas cuatro veces más rápido que para aquellas curvas cuyo cardinal es múltiplo de 2. La diferencia se debe a que en nuestro caso extraemos raíces cuadradas mientras que en [6] se tienen que solucionar ecuaciones cuadráticas de tipo general, lo cual es más costoso. En consecuencia, el algoritmo de bisección-y-adición también será más eficiente.

Agradecimientos

Este trabajo ha sido financiado parcialmente por los proyectos MTM2010-21580-C02-01 del Ministerio de Ciencia y Educación de España y 2009SGR-442 de la Generalitat de Catalunya.

Referencias

[1] P. Birkner, N. Thériault, "Faster halvings in genus 2", Selected Areas in Cryptography 2008, LNCS **5381**, 1–17 (2008).

[2] J. Canon, W. Bosma, D. Playoust, "The Magma algebra system I. the user language", J. symbolic comp. 24, no. 3-4, pp. 235-265 (2005).

[3] D. Cantor, Çomputing in the Jacobian of a Hyperelliptic Curve", Mathematics of Computation **48**, 95–101 (1987).

[4] Cohen, H., Frey, G., Avanzi R., Doche, C., Lange, T., Nguyen, K., Vercauteren, F.: Handbook of elliptic and hyperelliptic curve cryptography. Discrete Mathematics and its Applications (Boca Raton). Chapman & Hall/CRC, Boca Raton (2005).

[5] K. Fong, D. Hankerson, J. López, A. Menezes, "Field Inversion and Point Halving Revisited", IEEE Tansactions on Computers, **53** no. 8, 1047–1059 (2004).

[6] I. Kitamura, Katagi, Takagi, "A complete divisor class halving algorithm for hyperelliptic curve cryptosystems of genus two", LNCS **3574**, 146–157 (2005).

[7] E. W. Knudsen, .Elliptic Scalar Multiplication using Point Halving", en *K. Y. Lam, E. Okamoto and C. Xing (Eds.): ASIACRYPT99*, LNCS **1716**, pp. 135–149, 1999. Springer-Verlag Berlin Heidelberg (1999).

[8] T. Lange, M. Stevens, "Efficient Doubling on Genus Two Curves over Binary Fields", SAC 2004, LNCS **3357**, 170–181 (2005).

[9] Miret, Pujolàs, Rio, "Bisection for genus 2 curves in odd characteristic", Proc. of the Japan Academy, Vol. 85, Ser. A, no. 4, pp. 55-60 (2009).

[10] R. Schroeppel, "Elliptic Curve Point Halving Wins Big", Second Midwest Arithmetical Geometry in Cryptography Workshop (2000).

[11] J. Solinas, "Efficient Arithmetic on Koblitz curves", Designs, Codes and Cryptography, **19**, 195–249 (2000).