Treball final de grau

GRAU DE MATEMÀTIQUES

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

CRIPTOGRAFIA AMB CORBES EL·LÍPTIQUES

Autor: Ramon Pérez Garcia

Directora: Dra. Núria Vila
Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, 19 de gener de 2018

Abstract

In the last decades public key cryptosystems and digital signature algorithms have been designed based on mathematical objects that have been known and whose properties have been studied for a long time: elliptic curves.

The aim of this work is to study the basic properties of elliptic curves in order to be able to understand the behavior and to program some of the best known cryptosystems, like Diffie-Hellman key exchange and ElGamal digital signature.

Resum

En les últimes dècades s'han dissenyat sistemes criptogràfics de clau pública i mètodes per signar digitalment que aprofiten uns objectes matemàtics dels quals fa molt més temps se'n coneixen i estudien les propietats: les corbes el·líptiques.

L'objectiu d'aquest treball és estudiar les propietats bàsiques de les corbes el·líptiques per tal de poder comprendre el funcionament i programar alguns dels sistemes criptogràfics més coneguts, com l'intercanvi de claus Diffie-Hellman i la signatura digital ElGamal.

Índex

1 Introducció

Una branca important de la matemàtica aplicada i que forma part de la nostra vida diària és la criptografia. El món seria molt diferent si dues persones no es poguessin comunicar de manera segura a través de canals públics, ja que, des de fer una compra en línia a enviar un missatge amb una aplicació de missatgeria instantània requereixen encriptar informació.

Fa molt temps que es coneixen i s'utilitzen els sistemes criptogràfics de clau privada, és a dir, sistemes que requereixen que tant la persona que envia el missatge com la que el rep coneguin una clau privada. El fet que tots dos usuaris haguessin de posar-se en contacte per establir una clau feia inviable que dues persones que no es coneguessin i que no haguessin tingut contacte previ es poguessin comunicar de manera segura a través de canals públics. Aquest inconvenient va ser superat a la dècada de 1970, quan es van desenvolupar els primers criptosistemes de clau pública, entre ells el més conegut: RSA.

No va passar massa temps fins que Koblitz i Miller van proposar utilitzar les corbes el·líptiques per dissenyar sistemes criptogràfics, era l'any 1985, i els seus treballs van atraure molta atenció. El punt principal és el fet que els criptosistemes de clau pública tradicionals es basen en el problema del logaritme discret, que també es pot considerar en el cas de les corbes el·líptiques. Per tant, els criptosistemes tradicionals com l'intercanvi de claus Diffie-Hellman, l'encriptació de Massey-Omura, el xifrat de clau pública i la signatura digital ElGamal, etc., tenen una versió anàloga amb corbes el·líptiques.

El principal avantatge d'aquests nous criptosistemes és que, oferint la mateixa seguretat que els sistemes tradicionals com l'RSA, funcionen amb claus més petites i, per tant, requereixen menys recursos de memòria i processador. Aquest fet és especialment rellevant a l'hora d'implementar-los en targetes de crèdit, documents d'identitat i similars.

Per poder estudiar i implementar els sistemes criptogràfics amb corbes el·líptiques, es necessita una base matemàtica més forta que pels altres, per aquest motiu la primera part del treball està dedicada a estudiar els conceptes bàsics per tenir una petita base de coneixement sobre les corbes el·líptiques i poder obtenir alguns resultats necessaris.

La segona part està dedicada a l'estudi de les corbes el·líptiques sobre cossos finits, ja que aquestes corbes són les que utilitzen els criptosistemes, i al problema del logaritme discret. De les corbes el·líptiques sobre cossos finits es veurà un mètode eficient per determinar l'ordre del grup.

A continuació, a la tercera part, s'exposaran els criptosistemes amb corbes el·líptiques abans mencionats més alguns altres. També es veurà un algoritme de Lenstra per factoritzar nombres mitjançant corbes el·líptiques.

El treball inclou un paquet programat en Wolfram Language amb algunes funcions necessàries per implementar els sistemes criptogràfics amb el Mathematica. A l'última part del treball s'explicarà quines són i com funcionen les funcions del

paquet i es posaran exemples dels criptosistemes.

2 Conceptes bàsics

Definició 1. Sigui K un cos, una corba el·líptica E sobre K és el conjunt de solucions sobre el pla projectiu $P^2(K)$ d'una equació homogènia de Weierstrass de la forma:

$$E : y^2z + a_1xyz + a_2yz^2 = x^3 + a_3x^2z + a_4xz^2 + a_5z^3,$$

amb $a_1, a_2, a_3, a_4, a_5 \in K$. Aquesta corba ha de ser no singular, és a dir, si reescrivim l'equació en la forma $F(x, y, z) = 0$, aleshores les derivades parcials $\partial F/\partial x$, $\partial F/\partial y$ i $\partial F/\partial z$ no es poden anul·lar alhora a cap punt de la corba. També podem considerar el conjunt de solucions de l'equació a $P^2(L)$ on L és una extensió de K. Aleshores denotarem per $E|K$ l'equació E amb coeficients a K i per $E(L)$ (o $E(K)$) el conjunt de solucions sobre L (o K). La corba té exactament un punt amb coordenada z igual a 0, el punt (0:1:0). Aquest és el punt de l'infinit que denotarem per ∞.

Podem considerar la versió afí de l'equació :

$$E : y^2 + a_1xy + a_2y = x^3 + a_3x^2 + a_4x + a_5.$$

En aquest cas $E(K)$ és la reunió del conjunt de solucions de l'equació al pla afí i ∞.

Suposem que la característica de K és diferent de 2 i 3. Podem realitzar el canvi de variables

$$x = x' - \frac{a_1^2 + 4a_3}{12}, \;\; y = y' - \frac{a_1}{2}(x' - \frac{a_1^2 + 4a_3}{12}) - \frac{a_2}{2}.$$

Aquest canvi de variables transforma l'equació en la seva forma llarga de Weierstrass en una equació en la forma curta de Weierstrass:

$$E : y^2 = x^3 + ax + b,$$

amb $a, b \in K$.

Per simplificar, en aquest treball utilitzarem, en general, la forma curta de Weirstrass, així que suposarem que la característica de K no és ni 2 ni 3. A les primeres seccions del treball només es treballarà amb cúbiques sense arrels múltiples, és a dir, corbes no singulars; com el discriminant de la cúbica és $-(4a^3 + 27b^2)$, ha de ser $4a^3 + 27b^2 \neq 0$. Més endavant veurem què passa amb les corbes amb arrels múltiples.

Observem que si tenim una equació del tipus $cy^2 = dx^3 + ax + b$ amb $c, d \neq 0$, podem multiplicar els dos costats per c^3d^2 i obtenim

$$(c^2dy)^2 = (cdx)^3 + (c^2da)(xcd) + (bc^3d^2).$$

Aleshores, fent el canvi de variables $y_1 = c^2dy$, $x_1 = cdx$ obtenim

$$y_1^2 = x_1^2 + (ac^2d)x_1 + (bc^3d^2),$$

que és una equació de Weierstrass en la forma que volíem.

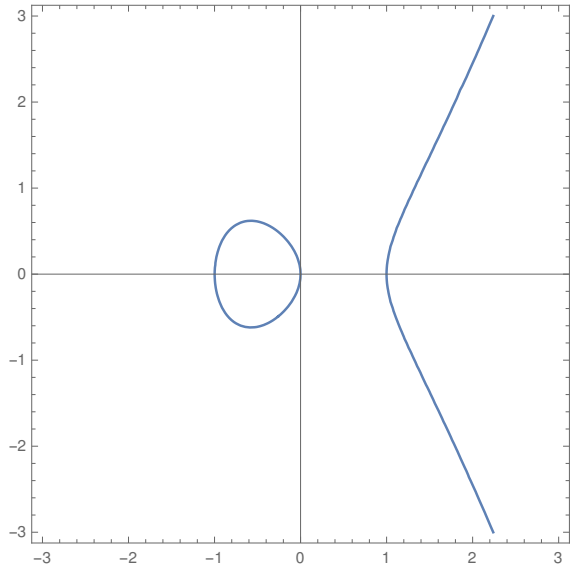

Figura 1: Gràfica de $y^2 = x^3 - x$

2.1 Llei de grup

En aquesta secció definirem una operació suma per secant i tangent dins de $E(K)$ i veurem que $E(K)$ amb aquesta operació forma un grup abelià. Començarem definint l'operació suma a partir de la seva interpretació geomètrica sobre el pla afí.

Donats dos punts $P_1 = (x_1, y_1)$ i $P_2 = (x_2, y_2)$ de $E(K)$, com veurem més endavant, la recta $\overline{P_1P_2}$ talla la corba en un tercer punt $P_3' = (x_3, y_3)$. Si aquesta recta és paral·lela a l'eix de les ordenades, és a dir $x_1 = x_2$, aleshores aquest tercer punt és el de l'infinit, i en aquest cas $P_1 + P_2 = \infty$. En cas contrari, definim la suma $P_1 + P_2 = P_3$ amb $P_3 = (x_3, -y_3)$. Per sumar un punt P amb ell mateix utilitzem el mateix procés amb la recta tangent a la corba E en P. Finalment, per tot punt P definim $P + \infty = P$. Es pot veure un exemple a la figura 2.

Volem obtenir les equacions que determinen $P_3 = (x_3, y_3)$ per obtenir una expressió algebraica de la suma a $E(K)$ per corbes $E|K$ de la forma

$$E : y^2 = x^3 + ax + b.$$

Siguin $P_1 = (x_1, y_1)$ i $P_2 = (x_2, y_2)$ punts de $E(K)$. Si $x_1 \neq x_2$, la recta $\overline{P_1P_2}$ té pendent $m = (y_2 - y_1)/(x_2 - x_1)$, per tant la seva equació és

$$y = m(x - x_1) + y_1.$$

Volem veure que hi ha un altre punt $P_3' = (x_3', y_3')$ de $E(K)$ que està a la recta $\overline{P_1P_2}$, així que substituim $\overline{P_1P_2}$ a l'equació de E i obtenim:

$$(m(x - x_1) + y_1)^2 = x^3 + ax + b,$$

que ho podem expressar com

$$x^3 - m^2x^2 + (a + 2m^2x_1 - 2my_1)x + (b - m^2x_1^2 + 2mx_1y_1 - y_1^2) = 0.$$

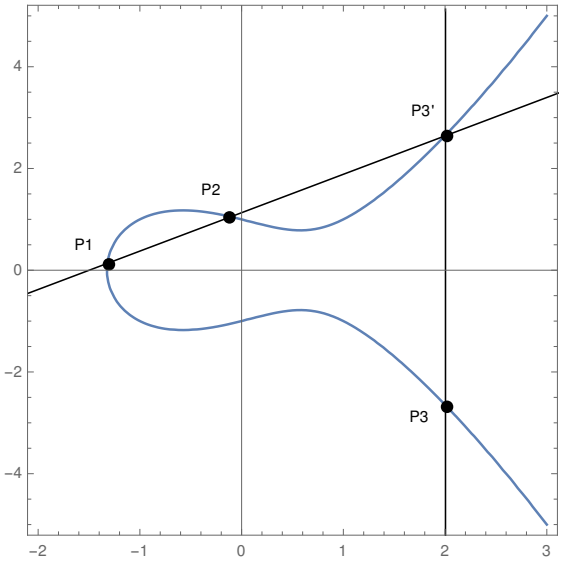

Figura 2: Representació gràfica de la suma de punts

Aquest polinomi que hem obtingut és un polinomi amb coeficients en K, per tant té tres arrels a $\overline{K}$. Com veurem a continuació, les tres arrels es troben a K. Per trobar la tercera arrel, observem que si tenim una cúbica amb arrels x_1, x_2 i x_3' aleshores:

$$(x-x_1)(x-x_2)(x-x_3') = x^3 + (-x_1-x_2-x_3')x^2 + (x_1x_2+x_1x_3'+x_2x_3')x - x_1x_2x_3'.$$

Per tant podem trobar la tercera arrel x_3' igualant els coeficients del terme x^2 per obtenir:

$$x_3' = m^2 - x_1 - x_2, \;\; y_3' = m(x_3 - x_1) + y_1.$$

Efectivament, x_3' i y_3' s'obtenen amb sumes, restes, multiplicacions i divisions d'elements de K, per tant $P_3' \in E(K)$. Finalment, fent la reflexió de P_3' respecte l'eix x obtenim les coordenades de P_3:

$$x_3 = m^2 - x_1 - x_2, \;\; y_3 = m(x_1 - x_3) - y_1.$$

Si $x_1 = x_2$ distingim dos casos: $y_1 \neq y_2$ i $y_1 = y_2$. Al primer cas la recta $\overline{P_1P_2}$ és vertical, per tant talla E a ∞. La reflexió de ∞ respecte a l'eix x dona ∞, ja que ∞ es troba tant a la part superior de l'eix y com a la inferior. La justificació d'aquest fet s'ha de buscar al pla projectiu. Podem considerar la inclusió del pla afí $A_K^2 = \{(x,y) : (x,y) \in K \times K\}$ al pla projectiu P_K^2:

$$A_K^2 \hookrightarrow P_K^2$$

$$(x,y) \mapsto (x:y:1).$$

Al pla projectiu, $E : y^2z = x^3 + axz^2 + bz^3$ té un únic punt a l'hiperplà de l'infinit, el punt $\infty = (0:1:0) = (0:-1:0)$. Per això diem que ∞ es troba tant a la part superior com inferior de l'eix y.

A l'altre cas tenim $x_1 = x_2$ i $y_1 = y_2$, és a dir, $P_1 = P_2 = (x_1, y_1)$. En aquest cas utilitzem la recta tangent a E a P_1. Podem obtenir el pendent $m = \partial y/\partial x$ d'aquesta fent la derivada parcial respecte x,

$$2y\frac{\partial y}{\partial x} = 3x^2 + a,$$

per tant

$$m = \frac{\partial y}{\partial x} = \frac{3x_1^2 + a}{2y}.$$

Aprofitem els càlculs anteriors per determinar les coordenades de P_3:

$$x_3 = m^2 - x_1 - x_2, \;\; y_3 = m(x_1 - x_3) - y_1.$$

Igual que abans, x_3 i y_3 s'obtenen amb sumes, restes, multiplicacions i divisions d'elements de K, per tant $P_3' \in E(K)$.

Finalment, la recta que talla P_1 i ∞ és vertical, per tant l'altre punt de la recta és la reflexió de P_1 respecte a l'eix x, i la reflexió de la reflexió és P_1. Notem que ∞ actua com a element neutre. Com $x_1 = x_2$ i $y_1 \neq y_2$ implica $P_1 + P_2 = \infty$, obtenim que la reflexió d'un punt P respecte de l'eix x és el seu oposat.

Com acabem de veure, $E(K)$ és tancada respecte l'operació suma que hem determinat, així que definim la llei de grup a partir dels resultats obtinguts.

Definició 2. Siguin K un cos i $E|K$ la corba $y^2 = x^3 + ax + b$. Siguin $P_1 = (x_1, y_1)$ i $P_2 = (x_2, y_2)$ punts de $E(K)$ amb $P_1, P_2 \neq \infty$. Definim $P_1 + P_2 = P_3 = (x_3, y_3)$ de la manera següent:

1. Si $x_1 \neq x_2$, aleshores

$$x_3 = m^2 - x_1 - x_2, \qquad y_3 = m(x_1 - x_3) - y_1,$$

$$\text{on } m = \frac{y_2 - y_1}{x_2 - x_1}.$$

2. Si $x_1 = x_2$ però $y_1 \neq y_2$, aleshores $P_1 + P_2 = \infty$.

3. Si $P_1 = P_2$ i $y_1 \neq 0$, aleshores

$$x_3 = m^2 - 2x_1, \qquad y_3 = m(x_1 - x_3) - y_1,$$

$$\text{on } m = \frac{3x_1^2 + a}{2y_1}.$$

4. Si $P_1 = P_2$ i $y_1 = 0$, aleshores $P_1 + P_2 = \infty$.

A més, per tot punt P de $E(K)$ definim:

$$P + \infty = P.$$

Notem que aquesta operació també està ben definida a $E(L)$ per qualsevol extensió L de K.

Proposició 1. Els punts de $E(K)$ formen un grup abelià amb ∞ com a identitat. És a dir:

1. $P_1 + P_2 = P_2 + P_1,$ per tot $P_1, P_2 \in E(K)$.
2. $P + \infty = P,$ per tot $P \in E(K)$.
3. Donat $P \in E(K),$ existeix $P' \in E(K)$ amb $P + P' = \infty$. (Denotarem l'invers per $-P$).
4. $(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3),$ per tot $P_1, P_2, P_3 \in E$.

Demostració. L'existència de neutre és certa per com hem definit les operacions, així que comencem per veure la commutativitat. Tal com hem definit la suma de punts, els casos 2, 3 i 4 són molt senzills: al cas 3 i 4 tenim $P_1 = P_2$, per tant la suma commuta. Al cas 2 tenim $P_1 + P_2 = \infty$ si $x_1 = x_2$ i $y_1 \neq y_2$, per tant $P_2 + P_1$ també és el cas 2 i tenim que $P_2 + P_1 = \infty = P_1 + P_2$. El cas 1 té una mica més de feina.
Suposem $x_1 \neq x_2$ i veiem que $P_1 + P_2 = P_2 + P_1$. Definim $P_1 + P_2 = P_3 = (x_3, y_3)$ i $P_2 + P_1 = P_4 = (x_4, y_4)$. Tenim

$$\frac{y_1 - y_2}{x_1 - x_2} = \frac{y_2 - y_1}{x_2 - x_1}$$

per tant m és la mateixa als dos casos. També tenim que $x_3 = x_4$, ja que

$$m^2 - x_1 - x_2 = m^2 - x_2 - x_1$$

Ara veiem que $y_3 - y_4 = 0$:

$$y_3 = m(x_1 - x_3) - y_1$$

$$y_4 = m(x_2 - x_3) - y_2$$

$$y_3 - y_4 = \frac{(y_1 - y_2)\left(-\frac{(y_1-y_2)^2}{(x_1-x_2)^2} + 2x_1 + x_2\right)}{x_1 - x_2} - \frac{(y_1 - y_2)\left(-\frac{(y_1-y_2)^2}{(x_1-x_2)^2} + x_1 + 2x_2\right)}{x_1 - x_2} - y_1 + y_2 =$$

$$= \frac{x_1 y_1}{x_1 - x_2} - \frac{x_2 y_1}{x_1 - x_2} + \frac{x_2 y_2}{x_1 - x_2} - \frac{x_1 y_2}{x_1 - x_2} - y_1 + y_2 =$$

$$= \frac{x_1 y_1}{x_1 - x_2} - \frac{x_2 y_1}{x_1 - x_2} + \frac{x_2 y_2}{x_1 - x_2} - \frac{x_1 y_2}{x_1 - x_2} - \frac{y_1(x_1 - x_2)}{x_1 - x_2} + \frac{y_2(x_1 - x_2)}{x_1 - x_2} = 0$$

Per veure l'existència d'invers hem de notar que si $P_1 = (x, y) \in E(K)$ aleshores $P_2 = (x, -y) \in E(K)$ i a més tenim que $P_1 + P_2 = \infty$, que és el tercer cas de la definició. Una prova elemental de l'associativitat es pot trobar al segon capítol de [2]. □

Exemple 1. Siguin $E : y^2 = x^3 - x$, la corba de la figura 1, i el grup abelià $E(\mathbb{Q}(\sqrt{6}))$. Siguin $P_1 = (-1, 0)$ i $P_2 = (2, \sqrt{6})$ punts de $E(\mathbb{Q}(\sqrt{6}))$. Com $-1 \neq 2$ som al primer cas. El pendent de la recta $\overline{P_1 P_2}$ és $\sqrt{6}/3 = \sqrt{2/3}$. Per tant,

$$(-1, 0) + (2, \sqrt{6}) = (-1/3, -2/3\sqrt{2/3}).$$

Observem que la suma dels punts no és la suma de les seves coordenades: $(1, \sqrt{6})$.

Exemple 2. Siguin $E : y^2 = x^3 - x$, la mateixa corba de l'exemple anterior, i el grup abelià $E(\mathbb{Q}(\sqrt{6}))$. Sigui $P = (2, \sqrt{6})$ un punt de $E(\mathbb{Q}(\sqrt{6}))$. Volem calcular $2P$, així que ens trobem al tercer cas. El pendent de la recta tangent a P és $11/2\sqrt{6}$. Per tant,

$$x_3 = \left(\frac{11}{2\sqrt{6}}\right)^2 - 4 = 25/24,$$

$$y_3 = \frac{11\,(2 - 25/24)}{2\sqrt{6}} - \sqrt{6} = \frac{253}{48\sqrt{6}} - \sqrt{6}.$$

Com hem vist, sumar dos punts a $E(K)$ requereix unes quantes operacions dins del cos K. Entre sumes, multiplicacions, restes i divisions al primer cas s'han de fer 9 operacions i al tercer 11, per tant la suma de punts és un procés feixuc, que es pot tornar complicat quan treballem amb determinats cossos finits.

2.2 Multiplicació de punts per un enter

Quan parlem de multiplicació de punts ens referim a l'operació de multiplicar un punt de la corba per un enter. Si K és un cos finit, aquesta operació està definida mòdul l'ordre de $E(K)$. Per fer aquesta operació, un mètode simple i eficient és el mètode binari. Siguin K un cos, $E|K$ una corba el·líptica, P un punt de $E(K)$ (o de $E(L)$ per una extensió qualsevol L de K) i m un enter; suposem que volem calcular $mP \in E(K)$.

Si m és un enter de ℓ bits amb $m = \sum_{j=0}^{\ell-1} n_j 2^j$, on $n_j \in \{0, 1\}$, podem calcular mP de la manera següent.

Per $j = \ell - 1$ fins 0:

$$Q_j = \begin{cases} 2Q_{j+1} & \text{si } n_j = 0 \\ 2Q_{j+1} + P & \text{si } n_j = 1 \end{cases}$$

on $Q_\ell = \infty$ i $Q_0 = mP$.

Vegem-ho més clar amb un exemple.

Exemple 3. Si volem calcular $20P$, podem fer $2(2(2(P + P) + P))$, ja que $20 = (10100)_2$.

Tot i que el mètode binari és un mètode molt eficient, existeixen altres que o són més aprofitant el fet que $-(x, y) = (x, -y)$ i considerant altres bases numèriques. El lector interessat pot trobar aquests mètodes més elaborats a la secció IV.2. de [3].

En algunes de les aplicacions criptogràfiques, com el mètode de Diffie-Hellman, s'han de calcular múltiples d'un punt fixat P. En aquests casos pot ser útil guardar una llista dels múltiples de P, a mesura que es vagin calculant, per aprofitar-los en càlculs següents.

2.3 El j-invariant

Definició 3. Siguin K un cos i $E|K$ la corba el·líptica donada per $y^2 = x^3 + ax + b$, definim el j-invariant de E com:

$$j(E) = 1728\frac{4a^3}{4a^3 + 27b^2}.$$

Com el denominador és el discriminant de la cúbica, i aquest és diferent de 0 (perquè la cúbica no té arrels múltiples), el j-invariant està ben definit.

Ara veurem una proposició que serà útil per entendre millor la demostració del següent teorema.

Proposició 2. Donades $E|K$ amb equació $y^2 = x^3 + ax + b$ i $\mu \in \overline{K}^*$, el canvi de variables

$$x_1 = \mu^2 x, \qquad y_1 = \mu^3 y,$$

porta a l'equació

$$y_1^2 = x_1^3 + \mu^4 a x_1 + \mu^6 b.$$

Demostració. $y^2 = x^3 + ax + b \implies \mu^6 y^2 = \mu^6 x^3 + \mu^6 ax + \mu^6 b \implies (\mu^3 y)^2 = (\mu^2 x)^3 + \mu^4 a(\mu^2 x) + \mu^6 b \implies y_1^2 = x_1^3 + \mu^4 a x_1 + \mu^6 b.$ □

Teorema 1. Siguin

$$y_1^2 = x_1^3 + a_1 x_1 + b_1,$$
$$y_2^2 = x_2^3 + a_2 x_2 + b_2$$

dos corbes el·líptiques sobre K amb j-invariants j_1 i j_2 respectivament. Si $j_1 = j_2$, aleshores existeix $\mu \in \overline{K}$ diferent de 0 tal que

$$a_2 = \mu^4 a_1, \qquad b_2 = \mu^6 b_1.$$

Demostració. Suposem $a_1 \neq 0$, això implica $j_1 \neq 0$ i $a_2 \neq 0$. Sigui $\mu \in \overline{K}$ tal que $a_2 = \mu^4 a_1$. Aleshores

$$\frac{4a_2^3}{4a_2^3 + 27b_2^2} = \frac{4a_1^3}{4a_1^3 + 27b_1^2} = \frac{4\mu^{-12}a_2^3}{4\mu^{-12}a_2^3 + 27b_1^2} = \frac{4a_2^3}{4a_2^3 + 27\mu^{12}b_1^2}.$$

Tenim $27b_2^2 = 27\mu^{12}b_1^2 \implies b_2^2 = (\mu^6 b_1)^2 \implies b_2 = \pm\mu^6 b_1$. Si $b_2 = \mu^6 b_1$, el canvi $i\mu \in \overline{K}$ manté $a_2 = \mu^4 a_1$ i aleshores $b_2 = \mu^6 b_1$.

Si $a_1 = 0$, aleshores $a_2 = 0$. Com $4a_i^3 + 27b_i^2 \neq 0$, tenim $b_1, b_2 \neq 0$. En aquest cas escollim μ tal que $b_2 = \mu^6 b_1$. □

Hi ha dos casos especials: les corbes amb j=0 i j=1728. Si j=0 aleshores la corba E és de la forma $y^2 = x^3 + b$ i si j=1728 la corba és de la forma $y^2 = x^3 + ax$. A més, per $j \neq 0, 1728$, es té que j és el j-invariant de la corba

$$y^2 = x^3 + \frac{3j}{1728 - j}x + \frac{2j}{1728 - j}.$$

Per tant, per tot $0 \leq j \leq 1728$ existeix una corba amb j-invariant j.

2.4 Corbes singulars

Fins ara hem assumit que x^3+ax+b no tenia arrels múltiples, és a dir, $y^2 = x^3+ax+b$ defineix una corba el·líptica, però també és interessant el cas en què sí que té arrels múltiples ja que, si traiem els punts singulars, la resta de punts juntament amb una operació essencialment igual a l'operació suma que hem definit al principi del capítol formen un grup. Donada una corba $E|K$ denotarem per $E_{ns}(K)$ el conjunt dels punts no singulars de $E(K)$, és a dir, els punts (x, y) tals que x no és arrel múltiple de $x^3 + ax + b$.

Comencem pel cas en què $x^3 + ax + b$ té una arrel triple $x = 0$. Tenim l'equació $y^2 = x^3$.

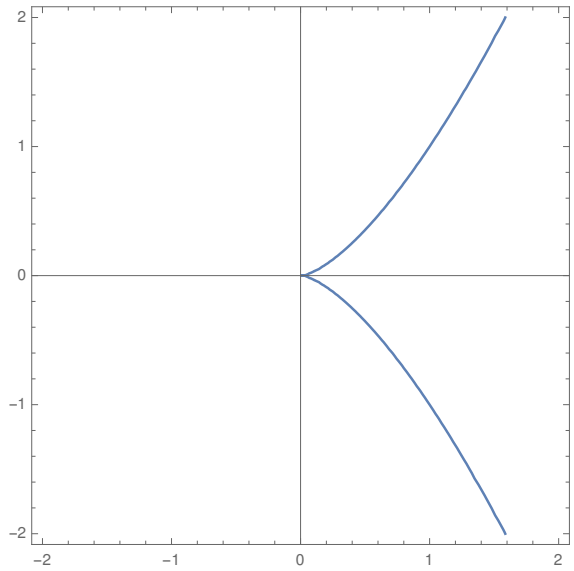

Figura 3: Gràfica de $y^2 = x^3$

Teorema 2. Sigui E la corba $y^2 = x^3$ i sigui $E_{ns}(K)$ el conjunt dels punts no singulars de $E(K)$ (∞ inclòs), la funció

$$f : E_{ns}(K) \to K$$

$$(x, y) \mapsto x/y,$$

$$\infty \mapsto 0,$$

és un isomorfisme de grups entre $E_{ns}(K)$, amb la llei de grup definida per secant i tangent, i K, considerat com a grup additiu.

Demostració. Considerem el canvi de variables $t = x/y$ (com hem tret el punt $(0, 0)$ està ben definit). Aleshores

$$y^2 = x^3 \implies x = \frac{y^2}{x^2} \implies x = \frac{1}{t^2},$$

$$y^2 = x^3 \implies y = x^2\frac{x}{y} \implies y = \frac{1}{t^3}.$$

Per tant podem expressar els punts de $E_{ns}(K)$ en termes del paràmetre t, amb $t = 0$ el punt de l'infinit, així que f és una bijecció. Falta veure que és morfisme de grups.

Suposem que $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$. Hem de veure que $t_1 + t_2 = t_3$, on $t_i = x_i/y_i$. Si $(x_1, y_1) \neq (x_2, y_2)$ (cas 1) aleshores

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2.$$

Substituint x_i per $1/t_i^2$ i y_i per $1/t_i^3$ tenim

$$x_3 = \frac{(1/t_2^3 - 1/t_1^3)^2}{(1/t_2^2 - 1/t_1^2)^2} - \frac{1}{t_1^2} - \frac{1}{t_2^2},$$

que es pot simplificar i obtenir

$$x_3 = \frac{1}{(t_1 + t_2)^2}.$$

Anàlogament

$$y_3 = \frac{y_2 - y_1}{x_2 - x_1}(x_1 - x_3) + y_1,$$

tornem a substituir x_i per $1/t_i^2$ i y_i per $1/t_i^3$ i tenim

$$y_3 = \left(1/t_2^3 - 1/t_1^3\right)\left(-\frac{(1/t_2^3 - 1/t_1^3)^2}{(1/t_2^2 - 1/t_1^2)^2} + \frac{2}{t_1^2} + \frac{1}{t_2^2}\right)\left(\frac{1}{t_2^2} - \frac{1}{t_1^2}\right)^{-1} - \frac{1}{t_1^3},$$

que es pot simplificar per obtenir

$$y_3 = \frac{1}{(t_1 + t_2)^3}.$$

Per tant, $t_3 = x_3/y_3 = t_1 + t_2$.

Veiem ara el cas $(x_1, y_1) = (x_2, y_2)$ i $y_1 \neq 0$ (cas 3). En aquest cas els càlculs són més senzills:

$$m = \frac{3x_1^2}{2y_1} = \frac{3}{2t_1},$$

$$x_3 = m^2 - 2x_1 = \frac{1}{4t_1^2},$$

$$y_3 = m\,(x_1 - x_3) - y_1 = \frac{1}{8t_1^3}.$$

Així que $t_3 = x_3/y_3 = 2t_1$.

Els dos casos que falten són $x_1 = x_2$ amb $y_1 \neq y_2$ (cas 2) i $x_1 = x_2$ amb $y_1 = y_2 = 0$ (cas 4). Al primer tenim $(x_1, y_1) + (x_2, y_2) = \infty$ i $y_2 = -y_1$, per tant $t_1 + t_2 = x_1/y_1 - x_1/y_1 = 0$, com 0 és la imatge de ∞ ja hem acabat. L'altre és trivialment cert, ja que no hi ha cap punt a $E_{ns}(K)$ amb $y = 0$. □

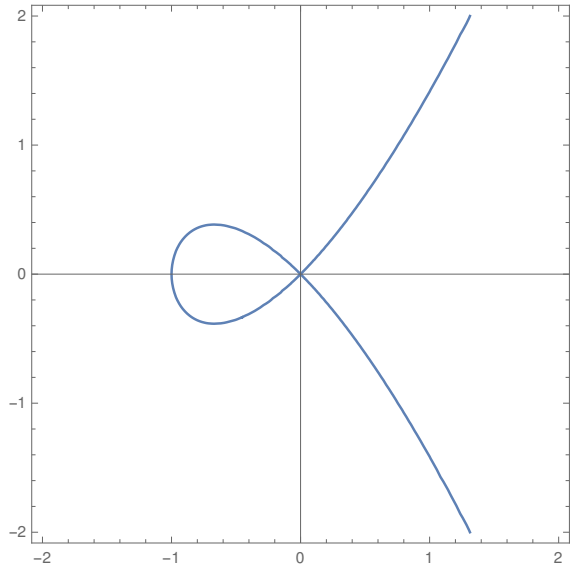

Figura 4: Gràfica de $y^2 = x^2(x+1)$

Considerem ara el cas en què $x^3 + ax + b$ té una arrel doble a $x = 0$, és a dir, considerem la corba E amb equació $y^2 = x^2(x+a)$, $a \neq 0$. Considerem $\alpha^2 = a$, aquest α pertany a alguna extensió de K. L'equació de E pot ser reescrita com

$$\left(\frac{y}{x}\right)^2 = a + x.$$

Per tant, quan $x \to 0$ podem aproximar E per

$$\left(\frac{y}{x}\right)^2 = a \qquad o \qquad \frac{y}{x} = \alpha,$$

així que les dues tangents de E al $(0,0)$ són

$$y = \alpha x \qquad i \qquad y = -\alpha x.$$

Teorema 3. Sigui $E|K$ la corba $y^2 = x^2(x+a)$ amb $a \neq 0$. Sigui α tal que $\alpha^2 = a$ i $L = K(\alpha)$. Considerem la funció

$$f : E_{ns}(L) \to L^*$$
$$(x, y) \mapsto \frac{y + \alpha x}{y - \alpha x}$$
$$\infty \mapsto 1$$

Aleshores f és un isomorfisme de grups entre $E_{ns}(L)$, amb la llei de grup definida per secant i tangent, i L, considerat com a grup multiplicatiu.

Demostració. La demostració segueix el mateix esquema que l'anterior. Considerem el canvi de variables $t = (y + \alpha x)/(y - \alpha x)$. Tenim que:

$$t + 1 = \frac{2y}{y - \alpha x}, \quad t - 1 = \frac{2\alpha x}{y - \alpha x},$$

per tant

$$\frac{t+1}{t-1} = \frac{2y}{2\alpha x},$$

és a dir

$$\frac{y}{x} = \frac{t+1}{t-1}\alpha.$$

Substituint a l'equació de la corba tenim que

$$x = \left(\frac{t+1}{t-1}\alpha\right)^2 - a = \frac{4\alpha^2 t}{(t-1)^2}.$$

Per aïllar y podem utilitzar que $y = x(y/x)$:

$$y = \frac{4\alpha^2 t}{(t-1)^2}\frac{t+1}{t-1}\alpha = \frac{4\alpha^3 t(t+1)}{(t-1)^3}.$$

Per tant f és bijectiva. Falta veure que és morfisme de grups.

Suposem que $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$. Hem de veure que $t_1 t_2 = t_3$, on

$$t_i = \frac{y_i + \alpha x_i}{y_i - \alpha x_i}.$$

Si $(x_1, y_1) \neq (x_2, y_2)$ (cas 1) aleshores

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - a - x_1 - x_2.$$

Observem que en aquest cas la fórmula és diferent, ja que el pendent és $\left(\frac{y_2-y_1}{x_2-x_1}\right)^2 - a$. Anàlogament

$$y_3 = \frac{(x_1 - x_3)(y_2 - y_1)}{x_2 - x_1} - y_1.$$

Substituint x_3 i y_3 a

$$t_3 = \frac{y_3 + \alpha x_3}{y_3 - \alpha x_3}$$

i simplificant obtenim $t_3 = t_1 t_2$.

Veiem ara el cas $(x_1, y_1) = (x_2, y_2)$ i $y_1 \neq 0$ (cas 3). Tenim

$$m = \frac{2ax_1 + 3x_1^2}{2y_1},$$

$$x_3 = m^2 - a - 2x_1 \text{ i}$$

$$y_3 = m(x_1 - x_3) - y_1.$$

Substituint x_3 i y_3 a

$$t_3 = \frac{y_3 + \alpha x_3}{y_3 - \alpha x_3}$$

i simplificant obtenim $t_3 = t_1^2$.

Els dos casos que queden són $x_1 = x_2$ però $y_1 \neq y_2$ (cas 2) i $x_1 = x_2$ amb $y_1 = y_2 = 0$ (cas 4). Al primer tenim $(x_1, y_1) + (x_2, y_2) = \infty$ i $y_2 = -y_1$, per tant

$$t_1 t_2 = \frac{y_1 + \alpha x_1}{y_1 - \alpha x_1} \frac{-y_1 + \alpha x_1}{-y_1 - \alpha x_1} = \frac{y_1 + \alpha x_1}{y_1 - \alpha x_1} \frac{y_1 - \alpha x_1}{y_1 + \alpha x_1} = 1,$$

com 1 és la imatge de ∞ ja hem acabat. L'altre és trivialment cert ja que si $y_1 = y_2 = 0$, aleshores $t_1 = t_2 = -1$ i, per tant, $t_3 = 1$. □

2.5 Punts de torsió

S'anomena punts de torsió als punts d'ordre finit. Com tots els punts de corbes el·líptiques definides a cossos finits són de torsió, serà útil per a més endavant veure algunes propietats.

Definició 4. Sigui E una corba el·líptica sobre K i n un enter positiu, definim

$$E[n] := \{P \in E(\overline{K}) : nP = \infty\},$$

és a dir, $E[n]$ són els punts de n-torsió a la clausura algebraica de K.

Mirant la llei de grup podem adonar-nos que $(x, y) \in E(\overline{K})$ és de 2-torsió només si $y = 0$, i els punts que compleixen això són les solucions de l'equació $x^3+ax+b = 0$, que al nostre cas són totes tres diferents. Si x_1, x_2 i x_3 són les arrels del polinomi $x^3 + ax + b$, aleshores

$$E[2] = \{(x_1, 0), (x_2, 0), (x_3, 0), \infty\}.$$

Observem que tenim l'isomorfisme de grups següent

$$E[2] \cong \mathbb{Z}/2\mathbb{Z} \oplus \mathbb{Z}/2\mathbb{Z}.$$

En el cas general es té el resultant següent, del qual es pot trobar una demostració a [2].

Teorema 4. Sigui E una corba el·líptica sobre un cos K i n un enter positiu. Si la característica de K no divideix n o és 0 aleshores

$$E[n] \cong \mathbb{Z}/n\mathbb{Z} \oplus \mathbb{Z}/n\mathbb{Z}.$$

Si la característica de K és $p > 0$ i p divideix n, tenim que n és de la forma $p^r n'$ on n' no és un múltiple de p. En aquest cas es té

$$E[n] \cong \mathbb{Z}/n'\mathbb{Z} \oplus \mathbb{Z}/n'\mathbb{Z}$$

o

$$E[n] \cong \mathbb{Z}/n\mathbb{Z} \oplus \mathbb{Z}/n'\mathbb{Z}.$$

Definició 5. Diem que una corba E sobre un cos de característica p és ordinària si $E[p] \cong \mathbb{Z}/p\mathbb{Z}$ i diem que és supersingular si $E[p] \cong 0$.

3 Corbes el·líptiques sobre cossos finits

A les aplicacions criptogràfiques necessitarem treballar amb grups finits, i les corbes el·líptiques sobre cossos finits ens aporten un ventall inexhaurible d'ells. En general, treballar amb un cos finit d'un nombre arbitrari q d'elements no és senzill, ja que la majoria tenen una estructura de grup complexa, però el cas en què $q = p$ és primer és molt més senzill, gràcies a que $\mathbb{F}_p^* \cong \mathbb{Z}/(p-1)\mathbb{Z}$.

Exemple 4. Sigui E la corba $y^2 = x^3 + x + 1$ sobre $\mathbb{F}_7$, observem l'esquema

x	x^3+x+1	y	punts
0	1	± 1	(0,1) (0,6)
1	3	-	-
2	4	± 2	(2,2) (2,5)
3	3	-	-
4	6	-	-
5	5	-	-
6	6	-	-
∞	∞	∞	∞

Conèixer l'estructura i la quantitat d'elements (o ordre) dels grups finits amb què treballem resulta molt útil, i en aquest capítol estudiarem aquestes dues característiques dels grups $E(\mathbb{F}_q)$. Començarem per veure un teorema que determina l'estructura de grup de $E(\mathbb{F}_q)$ i que resulta ser un corol·lari del teorema 4.

Teorema 5. Sigui E una corba el·líptica sobre $\mathbb{F}_q$, aleshores:

$$E(\mathbb{F}_q) \cong \mathbb{Z}/n\mathbb{Z} \text{ o}$$

$$E(\mathbb{F}_q) \cong \mathbb{Z}/n_1\mathbb{Z} \oplus \mathbb{Z}/n_2\mathbb{Z},$$

per algun enter $n \geq 1$ o per alguns enters $n_1, n_2 \geq 1$ amb $n_1 | n_2$.

Demostració. Com $E(\mathbb{F}_q)$ és un grup abelià finit és isomorf a

$$\mathbb{Z}/n_1\mathbb{Z} \oplus \mathbb{Z}/n_2\mathbb{Z} \oplus ... \oplus \mathbb{Z}/n_s\mathbb{Z}$$

amb $n_i | n_{i+1}$ per $i = 1, 2, ..., s-1$. Com $\mathbb{Z}/n_i\mathbb{Z}$ té n_1 elements d'ordre un divisor de n_1, tenim que $E(\mathbb{F}_q)$ té n_1^s elements d'ordre un divisor de n_1.

Pel teorema 4, $E(\mathbb{F}_q)$ té com a molt n_1^2 elements d'ordre n_1 (aquest seria el cas $E[n_1] \cong \mathbb{Z}/n_1\mathbb{Z} \oplus \mathbb{Z}/n_1\mathbb{Z}$), per tant $s \leq 2$. □

3.1 Determinar l'ordre del grup

Siguin $\mathbb{F}_q$ el cos finit de q elements i E una corba el·líptica sobre $\mathbb{F}_q$, si denotem per $\#E(\mathbb{F}_q)$ el nombre de punts de $E(\mathbb{F}_q)$, tenim que $\#E(\mathbb{F}_q) \leq 2q + 1$.

Però la cota anterior es pot afinar més. Si χ és el caràcter quadràtic de $\mathbb{F}_q$, és a dir, l'aplicació que envía els elements de $\mathbb{F}_q^*$ a +1 si són quadrats a $\mathbb{F}_q$, i a -1 si no ho són, tenim que

$$\#E(\mathbb{F}_q) = 1 + \sum_{x \in \mathbb{F}_q} (1 + \chi(x^3 + ax + b)) = 1 + q + \sum_{x \in \mathbb{F}_q} \chi(x^3 + ax + b).$$

El problema es troba a l'hora de determinar com es comporta $\sum_{x \in \mathbb{F}_q} \chi(x^3 + ax + b)$. Un resultat que ens aporta una cota de $\sum_{x \in \mathbb{F}_q} \chi(x^3 + ax + b)$ és el teorema demostrat per Helmut Hasse a la dècada de 1930, del qual es pot trobar una prova a [2].

Teorema 6. Teorema de Hasse. Sigui E una corba el·líptica sobre un cos finit $\mathbb{F}_q$, aleshores

$$q + 1 - 2\sqrt{q} \leq \#E(\mathbb{F}_q) \leq q + 1 + 2\sqrt{q}.$$

Aquest teorema dóna una cota que es pot aprofitar per desenvolupar algoritmes per calcular $\#E(\mathbb{F}_q)$. Suposem que coneixem l'ordre de diversos punts de la corba. Pel teorema de Lagrange (de teoria de grups) sabem que aquests ordres divideixen l'ordre de la corba, així que el mínim comú múltiple dels ordres d'aquests punts també ha de dividir l'ordre de la corba. Si aquest mínim comú múltiple és més gran que $4\sqrt{q}$, només tindrà un múltiple dins de l'interval que determina el teorema de Hasse, i aquest múltiple ha de ser $\#E(\mathbb{F}_q)$.

Per tant, un algoritme per determinar l'ordre de la corba pot ser generar aleatòriament punts de $E(\mathbb{F}_q)$ i determinar el seu ordre fins que el mínim comú múltiple dels ordres sigui més gran que $4\sqrt{q}$. Per posar en pràctica aquest algoritme necessitem poder determinar l'ordre dels punts de $E(\mathbb{F}_q)$ d'una manera eficient. En aquest procés també podem aprofitar el teorema de Hasse, ja que si P és un punt de $E(\mathbb{F}_q)$, sabem que existeix com a mínim un enter k dins de l'interval $[q+1-2\sqrt{q}, q+1+2\sqrt{q}]$ tal que $kP = \infty$. Podem anar calculant kP per $k = q+1-2\sqrt{q}, q+1-2\sqrt{q}+1, ...$ fins a trobar un que compleixi $kP = \infty$ i, un cop trobat, determinar quin és el mínim divisor d de k tal que $dP = \infty$. Aquest enter d serà l'ordre de P. Per obtenir k necessitarem, en el pitjor dels casos, $4\sqrt{q}$ passos.

Finalment, falta veure que aquest és un procés eficient per calcular $\#E(\mathbb{F}_q)$, és a dir, donat que calcular l'ordre dels punts de $E(\mathbb{F}_q)$ és un procés relativament costós per q elevat, falta justificar que amb uns pocs punts podrem aconseguir un mínim comú múltiple més gran que $4\sqrt{q}$.

Teorema 7. Siguin $\mathbb{F}_q$ un cos finit, $E|\mathbb{F}_q$ una corba el·líptica sobre $\mathbb{F}_q$, $P \in E(\mathbb{F}_q)$ i k l'ordre de P. La probabilitat que donat un factor primer p de $\#E(\mathbb{F}_q)$ k contingui la potència més gran de p que divideix $\#E(\mathbb{F}_q)$ és 1-1/p.

Demostració. Suposem $E(\mathbb{F}_q) \cong \mathbb{Z}/n_1\mathbb{Z} \oplus \mathbb{Z}/n_2\mathbb{Z}$ amb $n_1|n_2$. Aleshores l'ordre de tot element divideix n_2. Siguin P_1 i P_2 punts d'ordres n_1 i n_2 respectivament tals que tot punt P de $E(\mathbb{F}_q)$ té una expressió única de la forma $P = a_1P_1 + a_2P_2$. Sigui p un primer dividint n_2, agafem aleatòriament un punt P de $E(\mathbb{F}_q)$. Entre 0 i n_2 hi ha n_2/p múltiples de p, per tant, la probabilitat que $p \nmid a_2$ és $(n_2 - n_2/p)/n_2 = 1 - 1/p$. Sigui k l'ordre de P, com $kP = ka_1P_1 + ka_2P_2 = \infty$ i P_1 i P_2 són una base de $E(\mathbb{F}_q)$,

ha de ser $ka_2P_2 = \infty$, és a dir, ka_2 és un múltiple de n_2, però com $p|n_2$ i $p \nmid a_2$, es té que $p|k$. En particular, k conté la màxima potència de p que divideix n_2.

El mateix raonament serveix en el cas $E(\mathbb{F}_q) \cong \mathbb{Z}/n_1\mathbb{Z}$. □

Si p és petit, posem $p = 2$, la probabilitat que l'ordre d'un punt aleatori contingui la màxima potència de 2 que divideix $\#E(\mathbb{F}_q)$ és $1/2$, per tant, amb uns pocs punts hauríem de tenir suficient per trobar $\#E(\mathbb{F}_q)$. En conseqüència, si tenim un mètode eficient per determinar l'ordre de punts de la corba, tindrem un mètode eficient per determinar l'ordre del grup. A continuació exposarem un mètode que determina l'ordre d'un punt en, com a molt, $4q^{1/4}$ passos.

3.2 Baby step, Giant step

Sigui $E(\mathbb{F}_q)$ una corba el·líptica i $P \in E(\mathbb{F}_q)$, volem trobar un enter k a l'interval $[q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}]$ tal que $kP = \infty$. Per fer-ho seguim el següent procés:

1. Calculem $Q = (q + 1)P$.
2. Escollim un enter m amb $m > q^{1/4}$, aleshores calculem els punts jP per $j = 0, 1, 2, ..., m$ i guardem a una llista la seva coordenada en x.
3. Calculem els punts $Q + k(2mP)$ per $k = -m, -m + 1, ..., m$ fins que la coordenada en x d'algun d'ells coincideixi amb alguna de les coordenades en x de la llista anterior. Tenim que $Q + k(2mP) = \pm jP$. (Observem que ja hem calculat mP al pas anterior).
4. Aleshores $(q + 1 + 2mk \mp j)P = \infty$.

El mètode funciona perquè tenim $jP = Q + k(2mP)$, que implica

$$jP - jP = (q + 1)P + k(2mP) - jP,$$

és a dir

$$\infty = (q + 1 + 2mk - j)P.$$

Un cop hem trobat un enter k tal que $kP = \infty$, si volem trobar l'ordre de P, només queda determinar quin és el mínim divisor d de k tal que $dP = \infty$. Aquest d serà l'ordre de P.

4 El problema del logaritme discret

Siguin p primer i $a, b \in (\mathbb{Z}/p\mathbb{Z})^*$. Suposem que existeix un altre enter k tal que:

$$a^k \equiv b \pmod{p}$$

Trobar k és resoldre el problema del logaritme discret al grup multiplicatiu de $\mathbb{Z}/p\mathbb{Z}$. Com $a^{p-1} \equiv 1 \pmod{p}$, la solució del problema està definida mòdul $p-1$. En comptes de a $(\mathbb{Z}/p\mathbb{Z})^*$, podem considerar el problema del logaritme discret a una corba el·líptica $E|K$ on la llei és additiva, és a dir, donats $P, Q \in E(K)$ trobar k tal que

$$kP = Q$$

Resoldre el problema del logaritme discret a $(\mathbb{Z}/p\mathbb{Z})^*$ a força bruta (calculant $a^k \pmod{p}$ per $k = 0, 1, 2...$ fins que $a^k \equiv b \pmod{p}$) és inviable quan prenem un enter k i un primer p d'una magnitud elevada. De manera semblant, si prenem un enter k i un grup $E(K)$ prou grans, resoldre el problema del logaritme discret a $E(K)$ és inviable, ja que l'exponenciació a $(\mathbb{Z}/p\mathbb{Z})^*$ i la multiplicació per un enter a $E(K)$ tenen un cost computacional semblant. Aquest fet es deu a què per calcular ràpidament potències d'un enter arbitrari, el mètode més efectiu és l'exponenciació binaria, que és equivalent al mètode exposat per calcular múltiples d'un punt.

Tot i això, en alguns casos hi ha propietats de les corbes que poden ser aprofitades per simplificar el problema del logaritme discret. En aquest capítol estudiarem alguns atacs al problema del logaritme discret a corbes el·líptiques per tal d'obtenir condicions que ens permetin escollir corbes on resoldre el problema del logaritme discret sigui inviable.

4.1 Baby step, Giant step (D. Shanks)

Aquest és un mètode per resoldre el problema del logaritme discret a grups finits arbitraris. A continuació veurem un anàleg més eficient per corbes el·líptiques, tal com es proposa a l'exercici 5.1 de [2].

Siguin $\mathbb{F}_q$ un cos finit, $E|\mathbb{F}_q$ una corba el·líptica sobre $\mathbb{F}_q$, $N = \#E(\mathbb{F}_q)$, $P, Q \in E(\mathbb{F}_q)$ i k un enter tal que $kP = Q$; volem trobar k.

Sigui $m \geq \sqrt{N}$, aleshores podem escriure k com $k = k_0 + k_1 m$ amb $0 \leq k_0 \leq m$ i $0 \leq k_1 < m$, ja que m^2 és més gran que N. Partint d'aquesta base, que és la que s'utilitza en el cas de grups finits arbitraris, podem fer una petita modificació per aprofitar que un punt de la corba P i el seu oposat $-P$ tenen la mateixa coordenada en x. Podem prendre

$$-\frac{m}{2} \leq k_0 \leq \frac{m}{2} \qquad \text{i} \qquad 0 \leq k_1 \leq m.$$

Aleshores $Q - mk_1P = (k - mk_1)P = k_0P$.

Podem fer una llista dels valors de $Q - mk_1P$ per $0 \leq k_1 \leq m$ i una altra dels valors de k_0P per $0 \leq k_0 \leq \frac{m}{2}$. Com iP i $-iP$ tenen la mateixa coordenada en x,

tant si $Q - mk_1P = k_0P$ com $Q - mk_1P = -k_0P$ tindrem que a la primera llista hi haurà un element amb la mateixa coordenada en x que a la segona. Basant-nos en això desenvolupem el següent algoritme:

1. Fixem un enter parell $m \geq \sqrt{N}$. Si $\sqrt{N}$ és un nombre parell $m = \sqrt{N}$, ja ens vindrà bé. Si no coneguéssim N podríem aprofitar el teorema de Hasse i escollir m tal que $m^2 \geq q + 1 + 2\sqrt{q}$.
2. Fem una llista de les coordenades en x de iP per $0 \leq i \leq \frac{m}{2}$.
3. Aprofitem per calcular $mP = \frac{m}{2}P + \frac{m}{2}P$.
4. Calculem punts $Q - jmP$ per $j = 0, 1, 2, ...$, fins que la coordenada en x d'algun d'ells coincideixi amb la coordenada en x d'algun de la llista.
5. Determinem si $Q - jmP = iP$ o $Q - jmP = -iP$.
6. Si $Q - jmP = \pm iP$ aleshores $k \equiv \pm i + jm \quad (mod N)$.

Aquest mètode s'anomena baby step, giant step perquè combina passos petits, sumar P a $(i-1)P$, amb passos grans, restar mP a $Q - (j-1)mP$.
Per implementar aquest mètode es necessita guardar aproximadament $\frac{\sqrt{N}}{2}$ punts i fer $\sqrt{N}$ sumes, per tant només funciona bé si N no és massa gran.

4.2 Pohlig-Hellman

Aquest mètode serveix per resoldre el problema del logaritme discret per grups finits arbitraris. En el nostre cas el plantejarem per corbes el·líptiques. Siguin $\mathbb{F}_q$ un cos finit, $E|\mathbb{F}_q$ una corba el·líptica, P i Q punts de $E(\mathbb{F}_q)$ i k un enter tal que $Q = kP$. Sigui N l'ordre de P, suposem que coneixem la factorització

$$N = \prod p_i^{e_i},$$

on p_i són primers. El mètode consisteix en trobar k mòdul $p_i^{e_i}$ per tot i i, aleshores, mitjançant el teorema xinès del residu determinar k, ja que com hem mencionat a la introducció del capítol, el problema del logaritme discret $Q = kP$ està definit mòdul l'ordre de P.

Per determinar k mòdul p^e aprofitem que $k = k_0 + k_1p + k_2p^2 + ... + k_{e-1}p^{e-1}$ amb $0 \leq k_i < p$, i determinem successivament els k_i. Això ho fem aprofitant que

$$\frac{N}{p}Q = \frac{N}{p}kP = \frac{N}{p}(k_0 + k_1p + k_2p^2... + k_{e-1}p^{e-1})P =$$

$$= \frac{N}{p}k_0P + (k_1 + k_2p + ... + k_{e-1}p^{e-1})NP = \frac{N}{p}k_0P + \infty = \frac{N}{p}k_0P.$$

Com $0 \leq k_0 < p$, podem calcular $j\frac{N}{p}P$ per $j = 0, 1, ...$, fins trobar el que compleix

$$\frac{N}{p}Q = j\frac{N}{p}P,$$

aquest serà k_0.

A continuació definim $Q_1 = Q - k_0 P$, aleshores

$$Q_1 = Q - k_0 P = (k_1 p + k_2 p^2 + ... + k_{e-1} p^{e-1}) P,$$

així que mitjançant el mateix procés d'abans amb p^2 en comptes de p obtindrem k_1. Podem repetir el procés fins obtenir k_{e-1}.

Observem que $k_0 + k_1 p + k_2 p^2 + ... + k_{e-1} p^{e-1} \equiv k \mod p^e$.

Podem aplicar l'algoritme següent:

1. Factoritzem l'ordre N de P:
$$N = \prod p_i^{e_i}$$
amb p_i primers. Aleshores, per cada i:

2. Fem una llista dels $j(\frac{N}{p_i} P)$ per $0 \leq j < p_i$.

3. Calculem $\frac{N}{p_i} Q$ i busquem a la llista anterior l'element $j(\frac{N}{p_i} P) = \frac{N}{p_i} Q$. Definim $k_0 = j$.

4. Si $e_i = 1$ anem al pas 8.

5. Suposem que coneixem $k_0, ..., k_{r-1}$ i $Q = Q_0, ..., Q_{r-1}$. Calculem $Q_r = Q_{r-1} - k_{r-1} p_i^{r-1} P$.

6. Calculem $\frac{N}{p_i^{r+1}} Q_r$ i busquem a la llista anterior l'element $j(\frac{N}{p_i} P) = \frac{N}{p_i^{r+1}} Q_r$. Definim $k_r = j$.

7. Si $r \neq e - 1$ tornem al pas 5.

8. Apliquem el teorema xinès del residu per trobar k mòdul N.

Aquest mètode resulta eficaç sempre que els factors p_i de N no siguin massa grans, ja que per cada p_i hem de resoldre e_i cops el problema del logaritme discret mòdul p_i per trobar $j((N/p_i)P) = (N/p_i)Q$. Per tant, quan escollim una corba el·líptica per utilitzar-la en algun dels mètodes d'encriptació, haurem d'escollir-la tal que el seu ordre N contingui algun primer d'una magnitud suficient perquè sigui inviable resoldre el problema del logaritme discret mòdul aquest primer. Això ho podem fer escollint un punt de la corba el·líptica a l'atzar i factoritzant el seu ordre. Com hem vist abans, si p és un primer dividint l'ordre de la corba el·líptica, la probabilitat que l'ordre d'un punt de la corba contingui la màxima potència de p que divideix N és $1 - 1/p$, així que si els primers que divideixen l'ordre d'aquest punt no són de la magnitud suficient, és molt improbable que existeixi un primer de magnitud suficient dividint N.

4.3 Altres atacs

S'han dissenyat molts més algoritmes per atacar el problema del logaritme discret a corbes el·líptiques, alguns d'ells són els mètodes ρ i λ de Pollard, que tenen una eficàcia semblant al baby step, giant step, però que a diferència d'aquest, són algoritmes probabilístics. Un altre és l'atac MOV (Menezes, Okamoto i Vanstone), que funciona amb les corbes supersingulars sobre un cos finit $\mathbb{F}_q$ tals que $q + 1 = \#E(\mathbb{F}_q)$. Per evitar l'atac MOV, es va suggerir que s'utilitzessin corbes tals que $q = \#E(\mathbb{F}_q)$. Aquestes s'anomenen corbes anòmales, però, malauradament, en aquest cas també existeix un mètode per resoldre el problema del logaritme discret.

En conseqüència, quan escollim corbes per utilitzar-les en alguna de les aplicacions criptogràfiques les agafarem tenint en compte que $\#E(\mathbb{F}_q)$ ha de ser prou gran, que ha de tenir un factor primer p de magnitud semblant i que $\#E(\mathbb{F}_q)$ ha de ser diferent de q i $q + 1$.

5 Criptografia amb corbes el·líptiques

En aquest capítol s'exposaran diferents sistemes criptogràfics de clau pública que utilitzen corbes el·líptiques. Alguns d'ells, com el criptosistema de ElGamal, els vam veure en la seva versió per grups $(\mathbb{Z}/p\mathbb{Z})^*$ a l'assignatura d'Aritmètica de primer curs.

En alguns dels mètodes exposats en aquest capítol és necessari representar el missatge m com a un punt de $E(\mathbb{F}_p)$. Una manera senzilla de fer-ho és la següent.

Primerament convertim m en un enter més petit que $p/100-100$. Si el missatge és massa llarg, el podem dividir en blocs de k lletres i assignar un valor numèric $m < (p/100-100)$ a cada bloc. Per exemple, si m és un missatge format per lletres de l'alfabet anglès i nombres sense espais, aleshores m és un enter en base 36 (les xifres en base 36 són $0, 1, 2, ..., X, Y, Z$) i es pot fer un canvi a base decimal. Com que necessitem $m < (p/100 - 100)$, tenim que $(zzzzz)_{36} = 60466175$ i és raonable treballar amb primers $p > 6046627500$, es podria dividir el missatge en blocs de 5 caràcters.

Un cop tenim l'enter $m < (p/100 - 100)$ que volem expressar com a punt de $E(\mathbb{F}_p)$, on $E : y^2 = x^3 + ax + b$, considerem els enters $x_j = 100m + j \in \mathbb{F}_p$ per $0 \leq j < 100$. Per cada j calculem $s_j \equiv x_j^3 + ax_j + b \pmod{p}$, fins que algun d'ells sigui un quadrat a $\mathbb{F}_p$. Aleshores el punt $P \in E(\mathbb{F}_q)$ corresponent a m seria $(x_j, s_j^{1/2})$. Com la meitat dels elements de $\mathbb{F}_p^*$ són quadrats, la probabilitat que s_j no sigui un quadrat és $1/2$, així que la probabilitat que cap dels s_j sigui un quadrat és $1/2^{100}$, que és la probabilitat que no aconseguim expressar m com a un punt de $E(\mathbb{F}_p)$ (en aquest improbable cas només hauríem de canviar de corba).

Per recuperar m a partir del punt $P = (x_j, s_j^{1/2})$ només hem de calcular $[x_j/100]$, ja que $[x_j/100] = [m + j/100] = m$.

Donats un cos finit $\mathbb{F}_p$ i una corba $E : y^2 = x^3 + ax + b$ amb $a, b \in \mathbb{F}_p$, també serà necessari saber escollir aleatòriament punts de $E(\mathbb{F}_p)$, i per fer-ho podem utilitzar el següent algoritme:

1. Escollim aleatòriament $x \in \mathbb{F}_p$.
2. Calculem $z = x^3 + ax + b$.
3. Determinem el símbol de Legendre $(\frac{z}{p})$.
4. Si $(\frac{z}{p}) = 1$, calculem $y = z^{1/2} \in \mathbb{F}_p$. Si $(\frac{z}{p}) \neq 1$ tornem a començar.
5. Escollim aleatòriament entre $P = (x, y)$ i $P = (x, -y)$.

5.1 Intercanvi de claus Diffie-Hellman

Dues persones (o màquines) A i B es volen posar d'acord per escollir una clau que els hi serveixi per intercanviar dades amb un sistema d'encriptació simètric. A més,

suposem que l'únic mètode de contacte que tenen és públic. Diffie-Hellman és un mètode per establir aquesta clau privada mitjançant el següent procés:

1. A i B escullen un cos finit $\mathbb{F}_q$ i una corba el·líptica $E|\mathbb{F}_q$ tal que el problema del logaritme discret sigui complicat de resoldre a $E(\mathbb{F}_q)$. Aleshores escullen públicament un punt $P \in E(\mathbb{F}_q)$ tal que el subgrup generat per P tingui un ordre elevat, usualment primer.

2. A escull un enter a, que serà secret, calcula $aP \in E(\mathbb{F}_q)$ i se l'envia a B.

3. B escull un enter b, que serà secret, calcula $bP \in E(\mathbb{F}_q)$ i se l'envia a A.

4. A i B calculen $abP \in E(\mathbb{F}_q)$.

5. A i B utilitzen algun mètode, que s'haurà escollit públicament, per extreure de abP la clau privada. Per exemple, si després volen utilitzar el sistema Rijndael, poden utilitzar els últims 128 bits (o 192 o 256) de la coordenada x de abP com a clau. Un altre mètode podria ser escollir públicament una altra funció f i avaluar-la a abP.

Durant aquest procés la informació pública és E, $\mathbb{F}_q$, P, aP i bP, a part dels mètodes per extreure la clau, per escollir el cos i la corba i el mètode d'encriptació simètrica. No és coneix cap mètode per trobar abP a partir d'aquesta informació sense resoldre el problema del logaritme discret.

5.2 Encriptació de Massey-Omura

Suposem una situació semblant a la de l'apartat anterior, on A i B es volen comunicar per canals públics. Suposem que A vol enviar un missatge m a B; el mètode de Massey-Omura és el següent:

1. A i B escullen un cos finit $\mathbb{F}_q$ i una corba $E|\mathbb{F}_q$ tal que el problema de logaritme discret sigui complicat de resoldre a $E(\mathbb{F}_q)$, aleshores es computa $N = \#E(\mathbb{F}_q)$ que serà de domini públic.

2. A representa el missatge m com un punt $P_m \in E(\mathbb{F}_q)$; el mètode per representar el missatge com a punt de la corba ha de ser conegut per B i, per tant, públic.

3. A escull un enter e_A secret entre 1 i N tal que $mcd(e_A, N) = 1$ i calcula el seu invers $d_A \equiv e_A^{-1} \pmod{N}$ utilitzant l'algoritme d'Euclides. Aleshores calcula $e_A P_m$ i l'envia a B.

En aquest punt, si algú pogués resoldre el problema del logaritme discret podria trobar e_A, d_A i P_m, i per tant obtindria el missatge.

4. B escull un enter e_B secret entre 1 i N tal que $mcd(e_B, N) = 1$ i calcula el seu invers $d_B \equiv e_B^{-1} \pmod{N}$. Calcula $e_B e_A P_m$ i ho envia a A.

Pel teorema de Lagrange sabem que $NP_m = \infty$. $d_A e_A \equiv 1 \pmod{N}$ implica que existeix $k \in \mathbb{Z}$ tal que $d_A e_A = 1 + kN$. Per tant

$$d_A e_A P_m = (1 + kN)P_m = P_m + k\infty = P_m$$

5. A calcula $d_A e_B e_A P_m = e_B P_m$ i l'envia a B.

6. B calcula $d_B e_B P_m = P_m$. Ara a B ja només li queda convertir P_m en m.

5.3 Xifrat de clau pública ElGamal

Suposem que A i B es volen comunicar per canals públics. Per fer-ho es pot utilitzar el següent algoritme:

1. B escull un cos finit $\mathbb{F}_q$ i una corba $E|\mathbb{F}_q$ (tals que el problema del logaritme discret sigui difícil de resoldre a $E(\mathbb{F}_q)$).

2. B escull $P \in E(\mathbb{F}_q)$ i $s \in \mathbb{Z}$ (s serà la seva clau privada), i calcula sP. La clau pública de B seran el parell (P,sP), E i $\mathbb{F}_q$.

3. B envia la seva clau pública a A (o B la penja i A la descarrega). Tot seguit expressa el seu missatge m com a punt de $P_m \in E(\mathbb{F}_q)$.

4. A escull $k \in \mathbb{Z}$, calcula el parell (kP, $P_m + ksP$) i se'ls envia a B.

5. B calcula $(P_m + ksP) - s(kP) = P_m + ksP - ksP = P_m$.

Notem que ksP és necessari per obtenir P_m i que només pot ser obtingut per A i B. Com als casos anteriors, es creu que no hi ha manera de trobar P_m (sense conèixer s) sense resoldre el problema del logaritme discret. Una "vulnerabilitat" d'aquest mètode és el fet que si A utilitza el mateix enter k per enviar dos missatges P_{m_1} i P_{m_2} a B amb els seus corresponents parells $(kP, P_{m_1} + ksP)$ i $(kP, P_{m_2} + ksP)$, aleshores si algun dels missatges es fes públic, es podria trobar l'altre, ja que

$$(P_{m_1} + ksP) - (P_{m_2} + ksP) = P_{m_1} - P_{m_2}.$$

A més es podria detectar fàcilment aquest error perquè kP seria el mateix en els dos casos.

5.4 Signatura digital ElGamal

Suposem que A vol signar un document digital. Signar un document digitalment consisteix a donar una informació que només pugui conèixer la persona que signa, i que aquest fet es pugui verificar. El mètode de ElGamal per fer-ho és el següent.

1. A escull $\mathbb{F}_q$ i $E|\mathbb{F}_q$ (tals que el problema del logaritme discret sigui difícil de resoldre a $E(\mathbb{F}_q)$). També escull $P \in E(\mathbb{F}_q)$ d'orde N (normalment s'agafa un punt P amb ordre N primer de magnitud elevada), $a \in \mathbb{Z}$ (que serà secret) i una funció
$$f : E(\mathbb{F}_q) \to \mathbb{Z}$$
(per exemple, si $\mathbb{F}_q = \mathbb{F}_p$, es podria utilitzar la funció $f(x, y) = x$). Aquesta funció no té per què ser injectiva però hauria de complir que pocs (2 o 3) punts tinguessin la mateixa imatge. Finalment calcula $aP \in E(\mathbb{F}_q)$. Aleshores la clau pública és: E, $\mathbb{F}_q$, P, aP i f. Aquesta clau pot ser generada un cop i utilitzada per signar diversos missatges.

2. A representa el missatge que vol signar com a $m \in \mathbb{Z}$ amb $m < N$ (si $m \geq N$ veurem després com solucionar-ho). A continuació escull $k \in \mathbb{Z}$ (juntament amb a, seran la seva clau secreta) tal que $mcd(k, N) = 1$ i calcula kP i $s = k^{-1}(m - af(kP))$ mòdul N. El document de la signatura i la clau seran la terna (m, kP, s).

3. Per verificar la signatura a partir de (m, kP, s) i la clau pública s'han de calcular
$$V_1 = f(kP)aP + skP,$$
$$V_2 = mP.$$
Si $V_1 = V_2$ la signatura és vàlida.

Veiem per què funciona això:

$$s \equiv k^{-1}(m - af(kP)) \pmod{N} \implies sk \equiv (m - af(kP)) \pmod{N} \implies$$
$$\implies sk = m - af(kP) + xN \text{ per algún } x \in \mathbb{Z} \implies$$
$$\implies skP = mP - af(kP)P + xNP = mP - af(kP)P + \infty = mP - af(kP)P.$$

Per tant,

$$V_1 = f(kP)aP + skP = f(kP)aP + k^{-1}(m - af(kP))kP = f(kP)aP + mP - af(kP)P =$$
$$= mP = V_2.$$

En cas que $m \geq N$ es pot utilitzar una funció hash H que donat un missatge m de longitud arbitrària, $H(m)$ sigui un missatge d'una longitud fixada, per exemple 160 o 128 bits. En aquest cas, el document amb la signatura serà
$$(m, kP, s_H),$$
$$\text{on } s_H \equiv k^{-1}(H(m) - af(kP)) \pmod{N}.$$

En general, hi ha moltes propietats que s'estudien de les funcions hash, però per aquest mètode per signar les següents són suficients:

1. Baix cost: calcular el valor $H(m)$ ha de necessitar pocs recursos (computacionals, de memòria, ...).

2. Resistència a la preimatge: donat y ha de ser computacionalment intractable trobar x tal que $H(x) = y$.

3. Resistència a col·lisions: ha de ser computacionalment intractable trobar una parella x, y, $x \neq y$ tal que $H(x) = H(y)$.

Les propietats 2 i 3 són necessàries per evitar la falsificació de documents. Suposem que A signa un document (m, kP, s_H). Si algú pogués generar m' amb $H(m) = H(m')$, aleshores (m', kP, s_H) també seria una signatura vàlida, així que es podria falsificar la signatura. Tot i això és molt improbable que m' tingués sentit.

Si algú pogués calcular logaritmes discrets, podria utilitzar P i aP per trobar a i podria falsificar la signatura de A. També és important utilitzar diferents enters k per cada signatura. Suposem que A signa dos missatges amb la mateixa k, tenim els parells (m, kP, s) (m', kP, s'). Primerament, és fàcil adonar-se que s'ha utilitzat la mateixa k per signar els dos missatges perquè kP forma part de les dues signatures. En aquest cas tenim

$$ks \equiv m - af(kP) \pmod{N},$$
$$ks' \equiv m' - af(kP) \pmod{N}.$$

Per tant $k(s - s') \equiv m - m' \pmod{N}$. Sigui $d = mcd(s - s', N)$, hi ha d valors diferents per k, així que només s'han de provar els d valors diferents fins a trobar el que compleix que kP és el de la signatura. Un cop s'ha trobat k, es pot utilitzar la congruència

$$ks \equiv m - af(kP) \pmod{N},$$

per trobar a. Igual que abans, es pot anar provant valors fins a trobar la a correcta.

Si algú volgués verificar la signatura de A, hauria de coneixer la seva clau pública, però podria tenir el mateix dubte que amb la signatura: com sé que la clau pública de A realment és seva? La solució general a aquest problema és que qui vol verificar una signatura tingui un fitxer que relacioni totes les persones amb la seva clau pública. Per exemple, en el cas d'una empresa, la solució més senzilla seria que tothom tingués accés a una base de dades on estiguessin totes les claus públiques, així tothom podria verificar la identitat de la persona quan rebés un missatge verificant la seva signatura.

5.5 Algoritme de signatura digital (ECDSA)

Aquest mètode és una variant de l'anterior. Suposem que A vol signar un document $m \in \mathbb{Z}$, per fer-ho pot utilitzar el següent mètode:

1. A escull un cos finit $\mathbb{F}_q$ i una corba $E|\mathbb{F}_q$ tal que $\#E(\mathbb{F}_q) = fr$ on f és un enter petit (normalment 1,2 o 4) i r és un primer gran. També escull $P \in E(\mathbb{F}_q)$ d'ordre r. Finalment, A escull un altre enter a secret i calcula aP. La seva clau pública serà $\mathbb{F}_q$, E, r, P i aP.

2. A escull un altre enter k, $1 \leq k < r$, calcula $kP = (x, y)$ i $s \equiv k^{-1}(m + ax) \pmod{r}$. El document signat serà (m, kP, s).

3. Per verificar la signatura s'han de calcular $u_1 \equiv s^{-1}m \pmod{r}$ i $u_2 \equiv s^{-1}x \pmod{r}$, aleshores la signatura serà vàlida si $V = kP$, ja que:

$$V = u_1P + u_2aP = s^{-1}mP + s^{-1}xaP = s^{-1}(m + ax)P = kP.$$

Amb el mètode de ElGamal, per verificar una signatura s'ha de multiplicar un punt de la corba per un enter 3 vegades i en aquest només 2. Com que aquesta és la part més costosa a escala computacional dels mètodes, si s'han de fer moltes verificacions, ECDSA és una mica millor.

5.6 ECIES

Suposem que A vol enviar un missatge a B, per fer-ho pot utilitzar el sistema ECIES (Elliptic Curve Integrated Encryption Scheme):

1. B estableix la seva clau pública: escull $\mathbb{F}_q$ i $E|\mathbb{F}_q$ tals que el problema del logaritme discret sigui difícil de resoldre a $E(\mathbb{F}_q)$. També escull $P \in E(\mathbb{F}_q)$ d'ordre N (normalment un primer gran) i un enter secret s. Finalment calcula sP. La clau pública de B serà (q, E, N, P, sP) i la privada s.

2. A i B es posen d'acord per escollir dues funcions hash, H_1 i H_2, i una funció d'encriptació simètrica E_k que depengui d'un enter k amb la seva corresponent funció de desencriptació D_k.

3. A encripta el seu missatge m utilitzant la clau pública de B de la manera següent: escull un enter k, $1 \leq k < N$. Calcula $R = kP$ i $Z = k(sP)$ i els utilitza per calcular $H_1(R, Z) = k_1 \parallel k_2$, on k_1 i k_2 tenen una mesura específica ($k = k_1 \parallel k_2$ vol dir que k és el resultat de concatenar k_1 i k_2. Per exemple, $123456 = 123 \parallel 456$). Finalment calcula $C = E_{k_1}(m)$ i $t = H_2(C, k_2)$ i envia (R, C, t) a B.

4. Per desencriptar el missatge B fa el següent: calcula $Z = sR$ (aquest és el pas on utilitza la seva clau privada) i computa $H_1(C, k_2)$.

Si $H_2(C, k_2) \neq t$ aleshores no es segueix amb el procés. Això és una mesura de seguretat perquè hi ha atacs que es basen en forçar a B a desencriptar diversos missatges.

5. Si $H_2(C, k_2) = t$ aleshores B calcula $m = D_{k_1}(c)$.

Un avantatge d'aquest mètode respecte Massey-Omura i ElGamal és el fet que el missatge no s'ha de representar com a un punt de la corba.

6 Factorització amb corbes el·líptiques

L'objectiu d'aquesta part és estudiar el mètode de factorització amb corbes el·líptiques de Lenstra. Aquest mètode, desenvolupat a la dècada de 1980, és millor en alguns aspectes que els coneguts fins aleshores. No obstant això, aquesta millora no suposa una amenaça a la seguretat dels criptosistemes que es basen en la intractabilitat computacional de la factorització de nombres. Comencem recordant l'algoritme p-1 de Pollard.

6.1 Algoritme p-1 de Pollard (1974)

Suposem que volem factoritzar el nombre compost n i que p és un factor primer de n. Si $p-1$ té la propietat de no tenir un divisor primer gran, aleshores és molt probable que trobem p mitjançant l'algoritme següent:

1. Escollim aleatòriament un enter a, $2 \leq a \leq n-2$ i un altre enter B.
2. Calculem $a_1 \equiv a^{B!} \pmod{n}$. Ho podem fer recursivament:

$$a^{2!} \equiv a^2 \pmod{n},$$

$$a^{3!} \equiv (a^{2!})^3 \pmod{n},$$

$$a^{4!} \equiv (a^{3!})^4 \pmod{n},$$

$$\vdots$$

$$a^{B!} \equiv (a^{B-1!})^B \pmod{n}.$$

3. Calculem $d = mcd(a_1 - 1, n)$ mitjançant l'algoritme d'Euclides.
4. Si d no és un divisor no trivial de n, comencem amb un nou a i/o un nou B.

Veiem per què funciona aquest mètode. Si tots els factors primers de $p-1$ són menors o iguals que B, aleshores és probable que $B!$ sigui un múltiple de $p-1$ (la principal excepció és quan $p-1$ és divisible pel quadrat d'un primer entre $B/2$ i B). Si hi ha hagut sort i $B!$ és un múltiple de $p-1$, aleshores pel Petit teorema de Fermat tenim que:

$$a_1 \equiv a^{B!} \equiv 1 \pmod{p} \implies a_1 - 1 \equiv 0 \pmod{p} \implies$$

$$\implies p | mcd(a_1 - 1, n).$$

En aquest cas l'única manera en què no aconseguiríem un factor no trivial de n és el cas $a^{B!} \equiv 1 \pmod{n}$, però això és molt improbable.

Exemple 5. Intentem factoritzar $n = 4331$. Prenem $B = 4$, $B! = 24$ i $a = 2$.

$$2^{24} \equiv 3253 \pmod{4331}.$$

Tenim $mcd(4331, 3252) = 1$, per tant l'intent no ha funcionat. Provem ara amb $B = 5$, $B! = 120$ i $a = 2$.

$$2^{120} \equiv 2380 \pmod{4331}.$$

Tenim $mcd(4331, 2379) = 61$. Obtenim $4331/61 = 71$; hem aconseguit factoritzar 4331.

El primer intent no ha sortit perquè 24 no és múltiple ni de 60 ni de 70. En canvi, al segon intent tenim $B! = 120 = 2{\cdot}60$, per tant ha sortit bé.

El problema principal de l'algoritme és escollir B tal que un dels factors primers p de n compleixi que els factors primers de $p - 1$ siguin més petits que B. Si treballem amb nombres grans i resulta que $p - 1$ té algun factor primer elevat es torna bastant complicat que aquest mètode funcioni. En aquest cas les corbes el·líptiques ens aporten un nou mètode més eficient, ja que es poden canviar els grups.

Abans de començar hem d'introduir dos conceptes. Siguin E la corba el·líptica $y^2 = x^3 + ax + b$ amb $a, b \in \mathbb{Z}$ i p un primer tal que no divideix $4a^3 + 27b^2$, denotem per $E \pmod p$ la corba E sobre $\mathbb{F}_p$ obtinguda al reduir mòdul p els coeficients de l'equació $y^2 = x^3 + ax + b$. Sigui $P = (x, y)$ un punt de la corba el·líptica anterior i n un enter, denotem per $P \pmod n$ la reducció de les coordenades de P mòdul n, és a dir, $P \pmod n$=($x \pmod n$, $y \pmod n$). Per poder treballar amb $P \pmod n$, per exemple per calcular el seu doble $2P$, hi ha una condició necessària que s'ha de complir: que tots els denominadors siguin coprimers amb n.

6.2 Algoritme de Lenstra

El mètode que veurem a continuació és un anàleg de l'algoritme p-1 de Pollard que utilitza corbes el·líptiques.

1. Escollim una corba el·líptica $E : y^2 = x^3 + ax + b$ amb $a, b \in \mathbb{Z}$, un enter n tal que $mcd(4a^3 + 27b^2, n) = 1$ i un punt $P \pmod n$ que verifiqui l'equació.
2. Escollim un enter k i computem $(k!)P$ a E.
3. Si el pas 2 falla perquè el pendent no existeix mod n, haurem trobat un factor de n.
4. Si el pas 2 es pot completar, augmentem k o escollim una nova corba E i un punt P.

Veiem un exemple.

Exemple 6. Volem factoritzar 4453. Sigui E la corba $y^2 = x^3 + 10x - 2 \pmod{4453}$ i $P = (1, 3)$. Calculem $2P$. El pendent de la recta tangent a P és

$$\frac{3x^2 + 10}{2y} = \frac{13}{6} \equiv 3713 \pmod{4453}.$$

$mcd(6, 4453) = 1 \implies 6 \cdot 6^{-1} \equiv 1 \pmod{4453}$. Tenim que $6^{-1} = (4453 \cdot 5 + 1)/6 = 3711$. Per tant tenim que si $2P = (x, y)$, aleshores:

$$x \equiv 3713^2 - 2 \equiv 4332 \pmod{4453}, \quad y \equiv -3713(x - 1) - 3 \equiv 3230 \pmod{4453}.$$

Calculem ara $3P = P + 2P$, el pendent és

$$\frac{3230 - 3}{4332 - 1} = \frac{3227}{4331}.$$

Però $mcd(4331, 4453) = 61 \neq 1$. Així que $61|4453$. Concretament $4453 = 61 \cdot 73$.

7 Programes del paquet pel Mathematica

Per aquest treball s'ha creat un paquet per treballar amb la versió 11 del Mathematica amb l'objectiu de poder posar exemples dels diferents mètodes d'encriptació. S'ha escollit aquest programa perquè té predefinides moltes funcions necessàries per poder treballar amb cossos finits; a més la Universitat de Barcelona atorga llicències anuals als estudiants del grau de matemàtiques.

En aquesta secció explicarem el funcionament dels programes que venen al paquet adjunt al treball. Es va decidir només treballar amb corbes sobre cossos finits $\mathbb{F}_p$ amb p primer, ja que l'isomorfisme $\mathbb{F}_p \cong \mathbb{Z}/p\mathbb{Z}$ ens permet treballar més fàcilment sense que els mètodes d'encriptació siguin menys segurs.

La corba $E : y^2 = x^3 + ax + b$ sobre el cos finit $\mathbb{F}_p$ queda determinada per tres factors: els enters a i b i el primer p, per això al programa la informació de la corba es transmet com el vector $\{a,b,p\}$. Els punts $P = (x, y)$ de la corba el·líptica es representen com a vectors $\{x,y\}$, i el punt de l'infinit es representa per $\{\infty,\infty\}$ (La comanda per escriure ∞ amb el Mathematica és esc+"inf"+esc). Les variables del programa normalment reben com a nom una lletra seguida de les tres primeres lletres del programa en qüestió. Per exemple, al programa "escollirpunt", en comptes d'anomenar a una variable simplement "r" se l'anomena "resc". Això s'ha fet així perquè quan al Mathematica li assignes a una variable un valor, aquest es queda guardat i si després el crides a un altre programa pot donar errors. Aquest fet resultava problemàtic quan un programa del paquet cridava a un altre, així que es va decidir donar-li noms diferents a totes les variables dels diferents programes.

El primer programa que apareix al paquet és "EscollirPunt". Aquest programa escull aleatòriament un punt de la corba $y^2 = x^3 + ax + b$ sobre $\mathbb{F}_p$, és a dir, la corba determinada per $\{a,b,p\}$. Per fer-ho genera aleatòriament un punt $r \in \mathbb{F}_p$ i calcula $s = r^3 + ar + b$. Determina si s és un quadrat mòdul p mitjançant el símbol de Legendre i en cas de que ho sigui imprimeix $\{r,\sqrt{s}\}$. Aquest programa aprofita la funció "PowerMod[a,b,m]", que donat un enter a, calcula a^b mòdul m. Poder utilitzar aquesta funció per trobar arrels quadrades i inversos mòdul p és un dels grans avantatges que aporta el Mathematica a aquest treball. Una altra funció útil que aquí s'utilitza és "RandomInteger[$\{i_{min}, i_{max}\}$]", que genera aleatòriament un enter a l'interval $[i_{min}, i_{max}]$.

```
EscollirPunt[{a_, b_, p_}]:=Flatten[{
  resc = RandomInteger[{0, p − 1}];
  sesc = Mod[resc^3 + a ∗ resc + b, p];
  If[
    JacobiSymbol[sesc, p]==1,
    {resc, PowerMod[sesc, 1/2, p]},
    EscollirPunt[{a, b, p}]
  ]
}];
```

El següent programa és "ComprovarPunt". Aquest programa simplement determina si, donats un punt $\{x,y\}$ i una corba $\{a,b,p\}$, el punt compleix l'equació de la corba.

```
ComprovarPunt[{x_, y_}, {a_, b_, p_}]:=
  If[
    x==∞,
    True,
    If[
      Mod[y^2, p]==Mod[x^3 + a * x + b, p],
      True,
      False
    ]
  ]
```

El programa "SumaPunts" suma dos punts d'una mateixa corba el·líptica. El primer que fa és mirar si algun dels dos punts és ∞. En cas que algun dels dos ho sigui retorna l'altre, i en cas que no aplica les fórmules que apareixen al primer capítol.

```
SumaPunts[{x_, y_}, {w_, z_}, {a_, b_, p_}]:=
  If[x==∞||w==∞,
    If[
      x==∞,
      {w, z},
      {x, y}
    ],
    If[
      x!=w,
        msum = (z - y) * PowerMod[w - x, -1, p];
        rsum = Mod[(msum)^2 - x - w, p];
        ssum = Mod[(msum)(x - rsum) - y, p];
      {rsum, ssum},
      If[
        y!=z,
        {∞, ∞},
        If[
          y!=0,
          msum = (3x^2 + a) * PowerMod[2y, -1, p];
          rsum = Mod[msum^2 - 2x, p];
          ssum = Mod[msum(x - rsum) - y, p];
          {rsum, ssum},
          {∞, ∞}
        ]
```

```
      ]
    ]
  ]
```

Una altra operació bàsica necessària és la multiplicació d'un punt per un enter. Donats un enter k i un punt $P = \{x, y\}$ d'una corba el·líptica $\{a, b, p\}$, aquest programa calcula kP amb l'algoritme donat a la secció 2.3. Per fer-ho es necessita l'expressió en base 2 de k. La primera part del programa, fins al tercer "If", crea un vector que es correspon amb l'expressió en base 2 de k capgirada i sense el primer 1. Per exemple, si $k = 20 = (10100)_2$, es crea el vector $\{0, 0, 1, 0\}$. Primerament es genera un vector anomenat "vmul" buit, i a continuació, al "While", s'omple el vector aprofitant la funció "Append", que donats un vector de dimensió n i un element genera un vector de dimensió n+1 amb l'element al final. Un altre punt a tenir en compte del programa és que si k és negatiu, aleshores es calcula $k(-P) = k\{x, -y\}$.

```
MultiplicaPunt[k_, {x_, y_}, {a_, b_, p_}]:=Flatten[{
  If[
    k==0,
    {∞, ∞},
    If[k > 0, nmul = k, nmul = −k];
    vmul = {};
    imul = 0;
    While[
      nmul!=1,
      lmul = QuotientRemainder[nmul, 2];
      nmul = lmul[[1]];
      vmul = Append[vmul, lmul[[2]]];
      imul+=1;
    ];
    If[k > 0, {rmul, smul} = {x, y}, {rmul, smul} = {x, −y}];
    While[
      imul > 0,
      {rmul, smul} = SumaPunts[{rmul, smul}, {rmul, smul}, {a, b, p}];
      If[vmul[[imul]]==1, {rmul, smul} = SumaPunts[{rmul, smul}, {x, y}, {a, b, p}]; ];
      imul-=1;
    ];
    {rmul, smul}
  ]
}];
```

El programa més llarg del treball és el que utilitza el mètode Baby step, Giant step per determinar l'ordre d'un punt. El programa funciona d'una manera molt

semblant a l'algoritme explicat a la secció 3.2, i com a m s'utilitza $[p^{1/4}]+1$. El primer que fa el programa és crear la llista de les coordenades en x dels punts jP, per $j = 0, 1, 2, ..., m$. Després es calcula $2mP$, ho assigna a una variable auxiliar "auxbg", i calcula $Q - 2m^2P$. Per determinar els $Q + k(2mP)$ per $k = -m, -m+1, ..., m$, el programa comença amb $Q-2m^2P$, i li va sumant "auxbg"=$2mP$. Quan coincideix amb la coordenada en x d'algun jP, el programa determina si $Q + k(2mP) = jP$ o $Q+k(2mP) = -jP$, i li asaigna a "ordbg" el valor $q+1+2mk-j$. Com havíem vist a la secció 3.1, només podem assegurar que aquest "ordbg" és un múltiple de l'ordre de P, per tant l'últim pas és trobar el divisor més petit de "ordbg" que envia P a l'infinit. Per fer-ho es factoritza "ordbg" amb la funció "FactorInteger" i es divideix "ordbg" entre cada un dels seus factors primers tantes vegades com es pugui, vigilant que el quocient continuï sent una múltiple de l'ordre de P. Les parts del programa entre "(* *)" són comentaris.

```
BabyGiant[{x_, y_}, {a_, b_, p_}]:={
  qbg = MultiplicaPunt[p + 1, {x, y}, {a, b, p}];
  mbg = IntegerPart[p^(1/4)] + 1;
  jbg = 1;
  lbg = {∞};
  {rbg, sbg} = {∞, ∞};
  While[
    jbg<=mbg,
    {rbg, sbg} = SumaPunts[{rbg, sbg}, {x, y}, {a, b, p}];
    lbg = Append[lbg, rbg];
    jbg+=1;
  ];   (*{r, s} = mP*)
  auxbg = MultiplicaPunt[2, {rbg, sbg}, {a, b, p}];   (*aux = 2mP*)
  {rbg, sbg} = MultiplicaPunt[mbg, auxbg, {a, b, p}];   (*{r, s} = m2mP*)
  {rbg, sbg} = {rbg, −sbg};   (*{r, s} = −m2mP*)
  condbg = 0;
  ibg = −mbg;
  jbg = 0;
  {rbg, sbg} = SumaPunts[qbg, {rbg, sbg}, {a, b, p}];   (*{r, s} = Q − m2mP*)
  While[
    jbg < (mbg + 1)&&condbg==0,
    jbg+=1;
    If[
      rbg==lbg[[jbg]],
      condbg = 1;
    ];
  ];
  While[
    condbg==0&&ibg < mbg,
    ibg+=1;
    {rbg, sbg} = SumaPunts[{rbg, sbg}, auxbg, {a, b, p}];
    jbg = 0;
```

```
    While[
      jbg < (mbg + 1)&&condbg==0,
      jbg+=1;
      If[
        rbg==lbg[[jbg]],
        condbg = 1;
      ];
    ];
  ];
  jbg-=1;   (*lbg[[1]] es correspon amb 0P*)
  auxbg = MultiplicaPunt[jbg, {x, y}, {a, b, p}];
  If[auxbg[[2]]!=sbg, jbg = −jbg];
  ordbg = p + 1 + 2 * mbg * ibg − jbg; (*En aquest moment tenim Q + i2mP = jP*)
  factorbg = FactorInteger[ordbg];
  ibg = Dimensions[factorbg][[1]];
  jbg = 1;
  While[
    jbg<=ibg,
    If[
      MultiplicaPunt[ordbg/factorbg[[jbg]][[1]], {x, y}, {a, b, p}][[1]]==∞,
      ordbg = ordbg/factorbg[[jbg]][[1]];
      kbg = 1;
      While[
        kbg < factorbg[[jbg]][[2]]
        &&MultiplicaPunt[ordbg/factorbg[[jbg]][[1]], {x, y}, {a, b, p}][[1]]==∞,
        ordbg = ordbg/factorbg[[jbg]][[1]];
        kbg+=1;
      ];
    ];
    jbg+=1;
  ];
  ordbg
}[[1]]
```

La següent funció ha estat creada per tenir exemples de punts i els seus ordres i així poder comprovar que la funció "BabyGiant" funcionés bé, ja que no hi ha massa exemples als llibres. El seu funcionament és molt senzill: calcula p, $2p$, $3p$..., fins que algun d'ells sigui ∞.

```
DeterminarOrdrePunt[{x_, y_}, {a_, b_, p_}]:=Flatten[{
  {rdet, sdet} = {x, y};
  idet = 1;
  While[
    rdet!=∞,
```

```
    {rdet, sdet} = SumaPunts[{rdet, sdet}, {x, y}, {a, b, p}];
    idet+=1;
  ];
  idet
}]; [[1]]
```

A continuació hi ha un exemple del funcionament de les funcions "BabyGiant" i "DeterminarOrdrePunt". A l'exemple es pot veure que totes dues coincideixen i que la funció "BabyGiant" és molt més eficient. A més, s'ha utilitzat la funció "AbsolutTiming" que retorna un vector de dimensió 2: a la primera casella hi ha el temps en segons que s'ha trigat en executar el programa i a la segona el resultat. Les línies en negreta són el codi que s'introdueix al Mathematica i les que no estan en negreta és el que el programa retorna. Per al punt p_4 el programa "DeterminarOrdrePunt" ja no convergia. La funció "Prime[n]" retorna el n-éssim primer.

c_1 = {RandomInteger[{−100, 100}], RandomInteger[{−100, 100}], Prime[RandomInteger[{4, 100}]]}

$\{32, 50, 251\}$

p_1 = EscollirPunt [c_1]

$\{238, 82\}$

c_2 = {RandomInteger[{−10000, 10000}], RandomInteger[{−10000, 10000}], Prime[RandomInteger[{4, 10000}]]}

$\{-1883, 9784, 90007\}$

p_2 = EscollirPunt [c_2]

$\{33558, 22076\}$

c_3 = {RandomInteger[{−1000000, 1000000}], RandomInteger[{−1000000, 1000000}], Prime[RandomInteger[{4, 1000000}]]}

$\{-523153, -411492, 3972029\}$

p_3 = EscollirPunt [c_3] $\{2020341, 718402\}$

c_4 = {RandomInteger[{−100000000, 100000000}], RandomInteger[{−100000000, 100000000}], Prime[RandomInteger[{4, 100000000}]]}

$\{35894431, 15707265, 1574774059\}$

p_4 = EscollirPunt [c_4]

$\{340268922, 582942255\}$

c_5 = {RandomInteger[{−10000000000, 10000000000}], RandomInteger[{−10000000000, 10000000000}], Prime[RandomInteger[{4, 10000000000}]]}

$\{-9933057897, 389181720, 230223724903\}$

$p_5 = \text{EscollirPunt}[c_5]$

$\{198132782756, 114261677578\}$

$\{\text{AbsoluteTiming}[\text{DeterminarOrdrePunt}[p_1, c_1]], \text{AbsoluteTiming}[\text{BabyGiant}[p_1, c_1]]\}$

$\{\{0.003741, 248\}, \{0.000988, 248\}\}$

$\{\text{AbsoluteTiming}[\text{DeterminarOrdrePunt}[p_2, c_2]], \text{AbsoluteTiming}[\text{BabyGiant}[p_2, c_2]]\}$

$\{\{0.142007, 11248\}, \{0.003377, 11248\}\}$

$\{\text{AbsoluteTiming}[\text{DeterminarOrdrePunt}[p_3, c_3]], \text{AbsoluteTiming}[\text{BabyGiant}[p_3, c_3]]\}$

$\{\{25.1235, 1985942\}, \{0.007084, 1985942\}\}$

$\{\text{AbsoluteTiming}[\text{DeterminarOrdrePunt}[p_4, c_4]], \text{AbsoluteTiming}[\text{BabyGiant}[p_4, c_4]]\}$

$Aborted

$\text{AbsoluteTiming}[\text{BabyGiant}[p_4, c_4]]$

$\{0.055652, 787371189\}$

$\text{AbsoluteTiming}[\text{BabyGiant}[p_5, c_5]]$

$\{1.29098, 230224162210\}$

Un cop es té un programa que aplica l'algoritme de Baby step, Giant step per determinar l'ordre d'un punt de la corba, es pot fer un programa que calculi l'ordre de la corba. Per fer-ho es generen aleatòriament punts de la corba i es calcula el seu ordre fins que el mínim comú múltiple dels ordres sigui menor a $4\sqrt{p}+1$. Aleshores aquest mínim comú múltiple tindrà un únic múltiple a l'interval determinat pel teorema de Hasse, que és el que retorna el programa.

```
DeterminarOrdreCorba[{a_, b_, p_}]:={
  odoc = 1;
  mdoc = 1;
  idoc = (4 * p^(1/2) + 1);
  While[
    odoc < idoc,
    {xdoc, ydoc} = EscollirPunt[{a, b, p}];
    odoc = LCM[odoc, BabyGiant[{xdoc, ydoc}, {a, b, p}]];
  ];
  mdoc = odoc;
  While[
    mdoc < p + 1 - 2 * p^(1/2),
    mdoc+=odoc;
  ];
  mdoc
}[[1]]
```

Els següents tres programes serveixen per escollir corbes útils per a les aplicaci-

ons criptogràfiques: "ComprovarCorba[c]" retorna 1 si la corba c està ben definida i compleix els requisits donats al final del capítol 4 i retorna 0 si no compleix alguna d'elles, "EscollirCorbaCr" genera una corba útil per a les aplicacions criptogràfiques de la magnitud més gran possible que permet que els programes d'aquest treball convergeixin en uns pocs segons, i "EscollirPuntCr[c]" genera un punt de c d'ordre elevat. "EscollirPuntCr" no demana que l'ordre del punt sigui primer perquè aleshores el programa trigava massa temps a convergir.

```
ComprovarCorba[{a_, b_, p_}]:={
  auxcom = DeterminarOrdreCorba[{a, b, p}];
  factorcom = FactorInteger[BabyGiant[EscollirPunt[{a, b, p}], {a, b, p}]];
  If[
    Mod[4a^3 + 27b^2, p]!=0
    &&auxcom!=p
    &&auxcom!=p + 1
    &&factorcom[[Dimensions[factorcom][[1]]]][[1]] > (p/100)
    , 1, 0
  ]
}[[1]]
```

```
EscollirCorbaCr:={
  aecr = RandomInteger[{-1000000000, 1000000000}];
  becr = RandomInteger[{-1000000000, 1000000000}];
  pecr = Prime[RandomInteger[{900000000, 1000000000}]];
  While[
    ComprovarCorba[{aecr, becr, pecr}]==0,
    aecr = RandomInteger[{-1000000000, 1000000000}];
    becr = RandomInteger[{-1000000000, 1000000000}];
    pecr = Prime[RandomInteger[{900000000, 1000000000}]];
  ];
  aecr, becr, pecr
}
```

```
EscollirPuntCr[{a_, b_, p_}]:=Flatten[{
  rescr = RandomInteger[{0, p - 1}];
  sescr = Mod[rescr^3 + a * rescr + b, p];
  If[
    JacobiSymbol[sescr, p]==1,
    sescr = PowerMod[sescr, 1/2, p];
    ordescr = BabyGiant[{rescr, sescr}, {a, b, p}];
    If[
```

```
        ordescr > 1000000000,
        {rescr, sescr},
        EscollirPuntCr[{a, b, p}]
      ],
      EscollirPuntCr[{a, b, p}]
    ]
}];
```

Finalment, els dos últims programes serveixen un per convertir un enter m en un punt de la corba $\{a, b, p\}$, i l'altre per recuperar l'enter a partir del punt. S'utilitza el mètode explicat a l'inici del capítol 5 i, per tant, ha de ser $m < (p/100 - 100)$.

```
EnterAPunt[m_, {a_, b_, p_}]:=
  If[
    IntegerQ[m]&&m < (p/100 - 100),
    xeap = 100m;
    seap = Mod[xeap^3 + a * xeap + b, p];
    ieap = 0;
    While[
      JacobiSymbol[seap, p]!=1&&ieap < 100,
      xeap+=1;
      seap = Mod[xeap^3 + xeap * a + b, p];
      ieap+=1;
    ];
    If[
      JacobiSymbol[seap, p]==1,
      {xeap, PowerMod[seap, 1/2, p]},
      No s'ha pogut expressar el missatge com a punt de la corba
    ],
  Missatge no vàlid
]
```

```
PuntAEnter[{x_, y_}]:=IntegerPart[x/100]
```

8 Exemples de les aplicacions criptogràfiques

8.1 Exemple d'intercanvi de claus Diffie-Hellman

Suposem que A vol enviar un missatge secret m a B mitjançant canals públics i utilitzant el programa Mathematica. Per fer-ho, primer generaran una clau privada mitjançant el mètode de Diffie-Hellman, on a serà l'enter secret de A i b el de B. A utilitzarà la clau per encriptar el missatge mitjançant la funció "Encrypt["password",expr]". Com a "password" utilitzarà la coordenada en x de abp. Finalment B desencriptarà el missatge utilitzant la funció "Decrypt["password", enc]". L'exemple està fet des del punt de vista de B.

Primerament A i B escullen aleatòriament una corba c i un punt p [1] mitjançant les funcions creades per al treball. També comproven que l'ordre del punt generat sigui primer. Algunes línies de codi introduït no retornen res, això és perquè acaben en ";". Aquestes línies representen la informació que B rep de A i introdueix al programa per utilitzar-la.

c = EscollirCorbaCr

$\{-707382130, -737030327, 21554661241\}$

p = EscollirPunt[c]

$\{11605410490, 10630751686\}$

ordre = BabyGiant[p, c][[1]]

21554688677

PrimeQ[ordre]

True

Seguidament B escull aleatòriament un enter b menor que l'ordre del punt p i envia bp a A.

b = RandomInteger[ordre]

16277777824

bp = MultiplicaPunt[b, p, c]

$\{14058775658, 5119238447\}$

B rep ap i el missatge encriptat de A, així que procedeix a calcular abp i l'utilitza per desencriptar el missatge.

ap = {6640202439, 10619355221};

[1] S'han utilitzat lletres minúscules perquè el Mathematica 11 té reservades les lletres mayúscules per funcions específiques.

abp = MultiplicaPunt[*b*, ap, *c*]

$\{10155131870, 9745165659\}$

m = EncryptedObject[< |"Data" → ByteArray["TMxW/yu7BPC5CMJ96WjLnSdMrWXeJCZJcRCVGgI/4Uk="], "InitializationVector" → ByteArray["9UoiyvIUtvD3gdmjqyLPag=="], "OriginalForm" → String| >];

Decrypt["10155131870", m]

L'or és sota l'arbre.

8.2 Exemple d'encriptació de Massey-Omura

A vol enviar un codi $m = xyz12$ a B per canals públics. Per fer-ho el primer que fa és escollir un cos finit $\mathbb{F}_p$ i una corba el·líptica $E|\mathbb{F}_p$. També calcula $n = \#E(\mathbb{F}_p)$.

c = EscollirCorbaCr

$\{363390136, -251037771, 22277703709\}$

n = DeterminarOrdreCorba[c]

22277890324

A continuació representa el codi com a punt p de $E(\mathbb{F}_p)$.

m = 36^^xyz12

57059030

p = EnterAPunt[m, c]

$\{5705903000, 3511852068\}$

El pas següent de A és escollir un enter a secret entre 1 i n tal que $mcd(a, n) = 1$ i calcular ap i $d_a \equiv a^{-1} \pmod{n}$. Finalment envia ap a B.

a = RandomInteger[n]

21730401175

GCD[a, n]

1

d_a = PowerMod[a, −1, n]

4259684355

ap = MultiplicaPunt[a, p, c]

$\{3623332086, 11233313203\}$

Un cop té ap, B escull un enter b secret entre 1 i n tal que $mcd(b, n) = 1$ i calcula bap i $d_b = b^{-1} \mod n$. Aleshores envia bap a A.

b = RandomInteger[n]

12612422815

GCD[b, n]

1

d_b = PowerMod[$b, -1, n$]

10373176539

bap = MultiplicaPunt[b, ap, c]

$\{15691469266, 5066030331\}$

Un cop ha rebut bap, A calcula $d_a bap$ se l'envia a B.

$\mathrm{d}_a$bap = MultiplicaPunt [d_a, bap, c]

$\{11683225839, 16085396080\}$

Finalment, B pot recuperar el punt original multiplicant $d_a bap$ per d_b i llegir el codi que li ha enviat A.

$\mathrm{d}_b\mathrm{d}_a$bap = MultiplicaPunt [d_b, $\mathrm{d}_a$bap, c]

$\{5705903000, 3511852068\}$

PuntAEnter[$\mathrm{d}_b\mathrm{d}_a$bap]

57059030

BaseForm[57059030, 36]

$\mathrm{xyz12}_{36}$

8.3 Exemple de xifrat de clau pública ElGamal

Suposem que A vol enviar un missatge $m = xyz12$ a B per canals públics. Si vol utilitzar el xifrat de clau pública ElGamal, el primer pas és que B esculli la seva clau pública, és a dir, ha d'escollir un cos finit $\mathbb{F}_p$, una corba el·líptica $E(\mathbb{F}_p)$, un punt $p \in E|\mathbb{F}_p$ i un enter secret s per calcular sp.

c = EscollirCorbaCr

$\{490625788, 6449188, 21655318621\}$

p = EscollirPuntCr[c]

$\{11306368399, 3372318458\}$

s = RandomInteger[DeterminarOrdreCorba[c]]

16557030940

sp = MultiplicaPunt[s, p, c]

$\{10691295443, 17958636217\}$

Un cop A té descarregada la clau pública de B, procedeix a representar el seu missatge m com a un punt pm de la corba. A continuació escull aleatòriament un enter secret k, calcula kp, ksp i $pm + ksp$ i envia a B $(kp, pm + ksp)$.

m = 36^^xyz12

57059030

pm = EnterAPunt[m, c]

$\{5705903000, 10710747724\}$

k = RandomInteger[DeterminarOrdreCorba[c]]

6746021788

kp = MultiplicaPunt[k, p, c]

$\{1708536634, 1075019850\}$

ksp = MultiplicaPunt[k, sp, c]

$\{11981245151, 11066407447\}$

pmksp = SumaPunts[pm, ksp, c]

$\{21568120415, 8006933428\}$

Ara només queda que B utilitzi la seva clau secreta s per calcular skP i aleshores obtingui $pm = pmksp - skp$.

skp = MultiplicaPunt[s, kp, c]

$\{11981245151, 11066407447\}$

SumaPunts[pmksp, MultiplicaPunt[-1, skp, c], c]

$\{5705903000, 10710747724\}$

PuntAEnter[{5705903000, 10710747724}]

57059030

BaseForm[57059030, 36]

xyz12_{36}

8.4 Exemple de signatura digital ElGamal

A vol signar un document digital mitjançant el mètode de ElGamal. El primer que fa és generar una clau pública: E, $\mathbb{F}_p$, $P \in E(F_p)$, aP i f.

c = EscollirCorbaCr

$\{812658099, -299346868, 22671974461\}$

p = EscollirPuntCr[c]

$\{449834720, 9027233696\}$

n = BabyGiant[p, c]

11335938314

a = RandomInteger[DeterminarOrdreCorba[c]]

2437283490

f[{x_, y_}]:={x, y}[[1]]

ap = MultiplicaPunt[a, p, c]

$\{21763177162, 17314260613\}$

A continuació representa el missatge que vol signar com a $m \in \mathbb{Z}$, escull $k \in \mathbb{Z}$ tal que $mcd(k, N) = 1$ i calcula kP i $s \equiv k^{-1}(m - af(kP)) \pmod{N}$.

m = 36^^eldia5

882428621

k = RandomInteger[n]

11141439157

GCD[k, n]

1

kp = MultiplicaPunt[k, p, c]

$\{22211769277, 1658985788\}$

s = Mod[($m - a * f$[kp])PowerMod[$k, -1, n$], n]

11060125403

La clau pública serà:

{c, p, ap, "f[{x,y}]=x"}

$\{\{812658099, -299346868, 22671974461\}, \{449834720, 9027233696\},$
$\{21763177162, 17314260613\}$, f[{x,y}]=x}

I el missatge signat:

$\{m, \mathrm{kp}, s\}$

$\{882428621, \{22211769277, 1658985788\}, 11060125403\}$

Per verificar la signatura, B calcula V_1 i V_2 i comprova que siguin iguals.

$v_1 = \mathrm{SumaPunts}[\mathrm{MultiplicaPunt}[f[\mathrm{kp}], \mathrm{ap}, c], \mathrm{MultiplicaPunt}[s, \mathrm{kp}, c], c]$

$\{16654807798, 994392827\}$

$v_2 = \mathrm{MultiplicaPunt}[m, p, c]$

$\{16654807798, 994392827\}$

$v_1 == v_2$

True

9 Conclusions

Per comprendre les corbes el·líptiques es necessita una base matemàtica bastant forta perquè per definir algunes de les eines que s'utilitzen, com per exemple els aparellaments de Weil i Tate-Lichtenbaum, es requereixen coneixements no elementals. Aquest treball s'ha enfocat en poder programar alguns dels criptosistemes i, tot i no haver utilitzat els mètodes més potents, s'ha aconseguit treballar amb magnituds que ofereixen una bona seguretat amb agilitat, ja que el temps d'execució dels programes amb les magnituds utilitzades no superava els 5 segons.

Al llarg del treball s'han utilitzat conceptes assimilats a les assignatures d'Aritmètica, Geometria Afí, Geometria Projectiva, Equacions Algebraiques... Un resultat molt interessant que no s'ha vist en aquest treball, és que les corbes el·líptiques sobre els complexos són torus. Aquesta és, des del meu punt de vista, la part més bonica de les corbes el·líptiques: la seva multidisciplinarietat. Resulta molt interessant agafar conceptes de moltes branques diferents de la matemàtica i aplicar-los a criptografia. La visió obtinguda és que les corbes el·líptiques són un camp d'estudi molt ample amb aplicacions interessants, en el que encara queden moltes coses per veure.

Finalment, és important mencionar que no s'ha demostrat que el problema del logaritme discret a corbes el·líptiqes no pugui ser resolt en un temps computacional factible, per tant existeix la possibilitat que un descobriment inesperat deixi inservibles els algoritmes estudiats.

Referències

[1] Koblitz, N. (1994). *A course in Number Theory and Cryptography*, Second edition. Springer.

[2] Washington, L. C. (2003). *Elliptic curves: number theory and cryptography.* Chapman & Hall/CRC.

[3] Blake, I., Seroussi, G. i Smart, N. (1999).*Elliptic curves in Cryptography.* Cambridge University Press.

[4] Cohen, H. (1993). *A course in Computational Algebraic Number Theory.* Springer.

[5] Crespo, T. (2014). *Aritmètica, Curs 2013-2014.*

[6] Wolfram Research Inc. Wolfram Mathematica v.11 [amb llicència UB]. Champaign, Illinois: Wolfram Research Inc,1988.