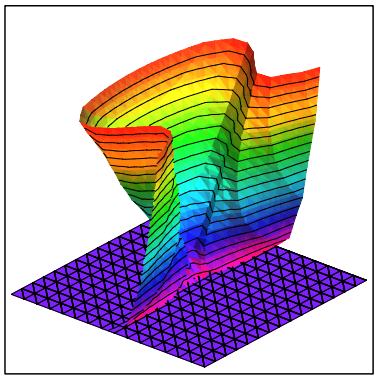
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Factorisation de Grands Nombres à l'Aide de Courbes Elliptiques



Thomas Baignères

SSC

Projet de Semestre

Mars 2003

Responsable
Prof. Serge Vaudenay
EPFL / LASEC

Superviseur
Jean Monnerat
EPFL / LASEC

LASEC

Table des matières

1 Géométrie projective **1**
1.1 Plan projectif et courbes sur le plan projectif 1
1.2 Lien avec la représentation affine 1
1.3 Courbes irréductibles 2
1.4 Intersection d'une cubique et d'une droite dans le plan projectif 3

2 Les courbes elliptiques **4**
2.1 Définition d'une courbe elliptique et premiers résultats 4
2.2 Courbe elliptique sur un corps de caractéristique $p > 3$ 5

3 La structure de groupe d'une courbe elliptique **6**
3.1 Résultat géométrique fondamental 6
3.2 Loi de groupe sur une courbe elliptique 7

4 Formules explicites **10**
4.1 Calcul de l'inverse de P 10
4.2 Calcul de l'addition de P et Q 10
4.3 Calcul du doublement de P 11

5 Une approche théorique de l'algorithme ECM **12**
5.1 L'algorithme $p-1$ de Pollard 12
5.2 La méthode ECM 12
5.3 Pourquoi chercher k tel que $[k]P = \mathcal{O}$ dans $E(\mathbb{F}_p)$? 13

6 Une approche pratique de l'algorithme ECM **14**
6.1 L'algorithme 14
6.2 Une légère modification 16
6.3 Le calcul de la borne B et la complexité de l'algorithme 16

7 Le calcul du point $[k]P$ **18**
7.1 La *Binary Method* 18
7.2 La *m-ary Method* 19

8 Comment paralléliser l'algorithme **20**
8.1 Mise en parallèle des calculs d'inverses 20
8.2 Avantage de la méthode 22

9 Une deuxième phase pour l'algorithme **22**
9.1 L'objectif et l'idée 22
9.2 La deuxième phase de l'algorithme ECM 23
9.3 Le choix de B_1 et B_2 pour la deuxième phase 24

10 Une deuxième phase du type *Paradoxe des anniversaires* **24**
10.1 Explications préliminaires 24
10.2 L'algorithme 26

11 Une (autre) deuxième phase du type *Paradoxe des anniversaires* **26**
11.1 Le déroulement de la deuxième phase 26
11.2 Implémentation et chances de succès 26

12 Une réalisation de l'algorithme ECM **28**
12.1 Les différentes versions de notre implémentation 28
12.2 Quelques résultats expérimentaux 29

13 Conclusion **30**

A Tableaux de B_1 **32**

B Une petite preuve mathématique **33**

Avant-Propos :

Cet exposé présente l'algorithme de factorisation par la méthode des courbes elliptiques mis au point par Lenstra (voir [6]). Les quatre premières sections présenterons les courbes elliptiques en rappelant un certain nombre de notions de géométrie projective. Ce qui y sera exposé devrait amplement suffire pour aborder la suite. Pour des approches plus complètes on pourra se reporter à [7] et [1].

Dans les huit sections suivantes, nous entrerons dans le vif du sujet. Nous explorerons certains détails de l'algorithme de factorisation sur des courbes elliptiques en attaquant le problème sous un angle théorique puis pratique. Nous détaillerons ensuite plusieurs améliorations venues s'ajouter à l'algorithme original depuis sa création.

Bonne lecture !

1 Géométrie projective

1.1 Plan projectif et courbes sur le plan projectif

DÉFINITION 1 *Soit un corps K. Le* ***plan projectif*** $\mathbb{P}_2(K)$ *est l'ensemble des points $P = (a, b, c) \neq (0, 0, 0) \in K^3$ de sorte que deux points $P = (a, b, c)$ et $P' = (a', b', c')$ sont considérés équivalents s'il existe $t \in K^*$ tel que $(a, b, c) = t(a', b', c')$.*

Dans la suite il nous arrivera d'écrire $\mathbb{P}_2$ à la place de $\mathbb{P}_2(K)$.

Nous allons maintenant définir la notion de courbe sur le plan projectif. Nous utiliserons pour cela des polynômes à trois variables. De par la définition du plan projectif, un point peut être représenté par plusieurs triplets différents mais équivalents. Il semble alors naturel de ne considérer que des polynômes $F(X, Y, Z)$ tels que si $F(a, b, c) = 0$ alors $F(ta, tb, tc) = 0$ pour tout t non nul.

DÉFINITION 2 *Un polynôme $F(X, Y, Z)$ est* ***homogène de degré*** *d s'il vérifie l'égalité suivante :*

$$F(tX, tY, tZ) = t^d F(X, Y, Z) \ .$$

Ces polynômes sont une somme de monômes de la forme $X^i Y^j Z^k$ avec $i + j + k = d$ et vérifient la propriété précédemment énoncée. Nous pouvons introduire le concept de courbe sur le plan projectif.

DÉFINITION 3 *Une* ***courbe*** *C sur le plan projectif $\mathbb{P}_2$ est l'ensemble des solutions d'une équation polynomiale*

$$C : \quad F(X, Y, Z) = 0 \ ,$$

où F est un polynôme homogène de degré supérieur ou égal à 1. Le degré de la courbe est le degré de ce polynôme.

Pour vérifier qu'un point $P = (a, b, c)$ appartient à la courbe, il suffira de vérifier que $F(a, b, c) = 0$. En effet si on choisit une autre représentation de ce point dans le plan projectif $\mathbb{P}_2$, soit (ta, tb, tc), on a :

$$F(ta, tb, tc) = t^d F(a, b, c) = 0$$

Toute représentation d'un point de la courbe est un zéro du polynôme F.

DÉFINITION 4 *Une courbe $D \in \mathbb{P}_2$ définie par un polynôme homogène de degré 1 est appelée une* ***droite****.*
Une courbe $C \in \mathbb{P}_2$ définie par un polynôme homogène de degré 3 est appelée une ***cubique****.*

1.2 Lien avec la représentation affine

Nous pouvons faire le lien entre une courbe du plan projectif telle que nous venons de la définir et une courbe du plan affine habituel (que nous noterons $\mathbb{A}_2$ ou $\mathbb{A}_2(K)$). Considérons une courbe C de $\mathbb{P}_2$ donnée par un polynôme homogène de degré d :

$$C : \ F(X, Y, Z) = 0$$

Si $P = (a, b, c)$ est un point de la courbe tel que $c \neq 0$, on pourra considérer que ce point du plan projectif correspond au point $(\frac{a}{c}, \frac{b}{c})$ du plan affine. Nous pouvons remarquer que deux représentations différentes d'un même point dans $\mathbb{P}_2$ donneront lieu à un unique point dans $\mathbb{A}_2$.

Par ailleurs on constate que si F est homogène de degré d et que $F(a, b, c) = 0$, alors :

$$F(\frac{a}{c}, \frac{b}{c}, 1) = \left(\frac{1}{c}\right)^d F(a, b, c) = 0 \ .$$

Nous pouvons alors définir une courbe dans le plan affine $\mathbb{A}_2$ à partir d'une courbe dans le plan projectif $\mathbb{P}_2$, dont les points (x, y) seront solution de l'équation $f(x, y) = 0$, avec f définie par :

$$\widetilde{C} : \ f(x, y) = F(x, y, 1) \ .$$

On constate qu'il y a correspondance entre les points $P = (a, b, c)$ de C tels que $c \neq 0$ et les points de $\widetilde{C}$. Sans entrer dans les détails, nous considérerons que les points de troisième coordonnée nulle sur la courbe C sont envoyé à l'infini dans le plan affine (pour plus de détails voir [10] ou [7]). Nous noterons l'ensemble de ces points à l'infini $\mathbb{P}_1 = \{(a, b, 0) \in \mathbb{P}_2\}$. Dans ce cas :

$$\boxed{\mathbb{P}_2 \approx \mathbb{A}_2 \cup \mathbb{P}_1}$$

Remarque : En ce qui concerne les courbes elliptiques, nous verrons que seul un point de la courbe appartient à $\mathbb{P}_1$. Nous le noterons $\mathcal{O}$.

1.3 Courbes irréductibles

DÉFINITION 5 *Un polynôme P est **factorisable** lorsqu'il existe deux polynômes U et V non constants de degré strictement inférieur à celui de P tels que :*

$$P(X, Y, Z) = U(X, Y, Z)V(X, Y, Z)$$

*Un polynôme est **irréductible** lorsqu'il n'est pas factorisable.*

Un polynôme peut être factorisé en un produit de polynômes irréductibles. Si $F \in K[X, Y, Z]$ est un polynôme, il existe $P_1, P_2, \ldots P_n \in K[X, Y, Z]$ tous irréductibles tels que :

$$F(X, Y, Z) = P_1(X, Y, Z)P_2(X, Y, Z) \cdots P_n(X, Y, Z)$$

Les P_i sont appelées les **composantes irréductibles** du polynôme F.

DÉFINITION 6 *Soit une courbe C définie par le polynôme $F(X, Y, Z) = 0$. La courbe C est dite **irréductible** si le polynôme F est irréductible.*

On dit que deux courbes C_1 et C_2 n'ont pas de pas de composante commune quand leur composantes irréductibles sont distinctes.

1.4 Intersection d'une cubique et d'une droite dans le plan projectif

PROPOSITION 1 *L'ensemble des points à l'intersection d'une cubique C et d'une droite D est fini si, et seulement si, ces deux courbes n'ont pas de composante irréductible en commun.*

Démonstration :

• Soit C l'ensemble des solutions de $F_1(X,Y,Z) = 0$ et D l'ensemble des solutions de $F_2(X,Y,Z) = 0$. Comme D est une droite, F_2 est un polynôme homogène de degré 1 et est donc irréductible. Dire que C et D ont une composante commune revient à dire que F_1 peut s'écrire sous la forme :

$$F_1(X,Y,Z) = F_2(X,Y,Z)G(X,Y,Z)$$

où G est un polynôme non constant. L'ensemble des points de D (i.e. les solutions de $F_2(X,Y,Z) = 0$) est alors inclus dans C. Il y a donc une infinité de points à l'intersection de C et de D. Par contraposée, si C et D se coupent en un nombre fini de points, elles n'ont pas de composante commune.

• Supposons maintenant que C et D n'ont pas de composante commune. La droite D est définie par un polynôme homogène de degré 1 :

$$D : \ F_2(X,Y,Z) = aX + bY + cZ$$

Soit $P = (X_P, Y_P, Z_P)$ un point à l'intersection de C et D.

Si $Z_P \neq 0$:

Les coordonnées affines du point P vérifient alors :

$$f_2(x_P, y_P) = ax_P + by_P + c = 0$$

On peut supposer, par symétrie, que $b \neq 0$. Dans ce cas :

$$y_P = -\frac{ax_P + c}{b}$$

L'équation suivante doit alors être vérifiée :

$$f_1(x_P, -\frac{ax_P + c}{b}) = 0$$

C'est un polynôme en x_P non nul car C et D n'ont pas de composante commune. Il admet donc un nombre fini de racines.

Si $Z_P = 0$:

Les coordonnées homogènes de P vérifient alors le système d'équation suivant :

$$\begin{cases} aX_P + bY_P = 0 \\ F_1(X_P, Y_P, 0) = 0 \end{cases}$$

En supposant, par symétrie, que $b \neq 0$, nous voyons que les cordonnées homogènes de P doivent vérifier :

$$F_1(X_P, -\frac{a}{b}X_P, 0) = 0$$

Cette équation est un polynôme en X_P non nul puisque C et D n'ont pas de composante commune. Il admet donc un nombre fini de racines.

Ainsi :

Si C et D n'ont pas de composante commune, elles se coupent en un nombre fini de points.

□

COROLLAIRE 1 *Soit C une cubique irréductible et D une droite. C et D se coupent en un nombre fini de points.*

2 Les courbes elliptiques

2.1 Définition d'une courbe elliptique et premiers résultats

DÉFINITION 7 *Soit un corps K. Une **courbe elliptique** sur K est une cubique irréductible, non singulière, définie comme l'ensemble des solutions du plan projectif $\mathbb{P}_2(K)$ de l'équation de Weierstrass homogène suivante :*

$$\boxed{E : Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3} \quad (1)$$

avec $a_1, a_2, a_3, a_4, a_5, a_6 \in K$.

Une courbe elliptique doit être non singulière, c'est à dire que si on écrit l'équation précédente sous la forme d'une équation homogène $F(X, Y, Z) = 0$, alors les dérivées partielles de F ne doivent pas s'annuler simultanément en un point de la courbe.

Remarquons qu'une telle courbe admet un unique point de coordonnée Z nulle, le point $(0, 1, 0)$. Il sera noté dans la suite $\mathcal{O}$.

Par la suite nous utiliserons la plupart du temps la représentation affine de l'équation de Weierstrass :

$$\boxed{E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6} \quad (2)$$

où $a_i \in K$. Pour $Z \neq 0$, un point (X, Y, Z) solution de l'équation (1) correspond à un point (x, y) solution de l'équation (2) avec $(x, y) = (X/Z, Y/Z)$.

L'ensemble des solutions de l'équation (1) correspond à l'union entre les solutions de l'équation (2) et du point $\mathcal{O}$.

Dans la suite il sera parfois plus simple d'utiliser les variables suivantes :

$$
\begin{aligned}
b_2 &= a_1^2 + 4a_2 \\
b_4 &= 2a_4 + a_1 a_3 \\
b_6 &= a_3^2 + 4a_6 \\
b_8 &= a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2 \\
c_4 &= b_2^2 - 24 b_4 \\
c_6 &= -b_2^3 + 36 b_2 b_4 - 216 b_6
\end{aligned}
$$

Remarque : La définition 7 (d'une courbe elliptique) est issue de [7]. Une autre version, qu'on trouvera par exemple dans [1], est d'élargir le domaine de définition de la courbe à $\mathbb{P}_2(\overline{K})$.

DÉFINITION 8 *Le **discriminant** Δ de l'équation de Weierstrass est la quantité :*

$$\boxed{\Delta = -b_2^2 b_8 - 8b_4^3 - 27b_6^2 + 9 b_2 b_4 b_6} \ .$$

*Lorsque le discriminant est non nul, on définit le **j-invariant** de la courbe elliptique E :*

$$\boxed{j(E) = \frac{c_4^3}{\Delta}} \ .$$

La courbe elliptique est non singulière lorsque son discriminant est non nul. Le j-invariant est lié à la notion d'isomorphisme sur une courbe elliptique. Sans entrer dans les détails (nous n'en aurons pas besoin par la suite), deux courbes elliptiques définies par leur équation de Weirstrass à coefficient sur un corps algébriquement clos sont isomorphes si, et seulement si, elles ont le même j-invariant (pour une démonstration de ce résultat, voir [7]).

2.2 Courbe elliptique sur un corps de caractéristique $p > 3$

PROPOSITION 2 *Soit K un corps de caractéristique $p > 3$. Une courbe E donnée par l'équation 2 prend alors la forme simplifiée suivante :*

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

avec :

$$\boxed{\Delta = -16(4a^3 + 27b^2)} \quad et \quad \boxed{j(E) = 1728 \frac{4a^3}{4a^3 + 27b^2}}$$

Preuve :

Puisque le corps n'est pas de caractéristique 2, on peut effectuer le changement de variable suivant :

$$y \rightarrow y - \frac{1}{2}(a_1 x + a_3) \ ,$$

pour que l'équation (2) devienne :

$$y^2 = x^3 + \frac{b_2}{4} x^2 + \frac{b_4}{2} x + \frac{b_6}{4} \ .$$

Puisque la caractéristique de corps n'est ni 2, ni 3 on peut effectuer le changement de variable suivant :

$$x \to x - \frac{b_2}{12} \ .$$

L'équation que nous venons d'obtenir devient alors :

$$y^2 = x^3 - \frac{c_4}{48}x - \frac{c_6}{864} \ .$$

Il suffit de poser $a = -\frac{c_4}{48}$ et $b = -\frac{c_6}{864}$ pour obtenir l'équation souhaitée.
On peut effectuer les mêmes changements de variable pour obtenir les nouvelles équations de Δ et de $j(E)$.

□

Sur un corps fini, le nombre de points de la courbe est fini. Le théorème de Hasse nous permet d'en connaître approximativement le nombre :

THÉORÈME 1 *(Théorème de Hasse) Soit $\mathbb{F}_q$ un corps fini et soit E une courbe elliptique définie sur $\mathbb{F}_q$. Le nombre de points de la courbe est tel que :*

$$\#E(\mathbb{F}_q) = q + 1 - t \ ,$$

où t est tel que :

$$|t| \leq 2\sqrt{q} \ .$$

3 La structure de groupe d'une courbe elliptique

3.1 Résultat géométrique fondamental

PROPOSITION 3 *Soient une cubique irréductible non singulière C et une droite L définies sur un corps K. Si la cubique C a au moins deux points d'intersection (comptés avec leur multiplicité) avec la droite L, alors le nombre de points d'intersection (comptés avec leur multiplicité) entre C et L est exactement 3.*

Preuve :

En effet, comme C est irréductible, nous savons grâce à la Proposition 1 que le nombre de points à l'intersection de C et D est fini. Soit la droite D : $aX + bY + cZ = 0$ où, par symétrie, nous supposons $c \neq 0$. Les points $P = (X, Y, Z)$ sont racine du polynôme $F(X, Y, -\frac{aX+bY}{c})$ où F est le polynôme homogène de degré 3 qui définit C. Notons :

$$q(X, Y) = F(X, Y, -\frac{aX + bY}{c}) \ ,$$

et soient $P_1 = (a_1, b_1, c_1)$ et $P_2 = (a_2, b_2, c_2)$ deux points à l'intersection de C et D (avec éventuellement $P_1 = P_2$). Comme $q(a_1, b_1) = q(a_2, b_2) = 0$, on peut écrire :

$$q(X, Y) = v(X, Y)(b_1X - a_1Y)(b_2X - a_2Y) \ ,$$

où v est un polynôme homogène de degré 1. Il n'a donc qu'une racine que nous noterons (a_3, b_3). Le point $P_3 = (a_3, b_3, -\frac{aa_3+bb_3}{c})$ est alors le troisième point à l'intersection de C et D

□

Cette proposition permet de définir la **loi de composition de la sécante-tangente** :

1. Si $P, Q \in C(K)$ et si $P \neq Q$, nous pouvons définir $D = PQ$ la droite sécante passant par P et Q. Grâce à la proposition précédente nous savons que cette droite coupe la courbe C en un troisième point unique qui appartient donc à $C \cap D$. Nous noterons ce troisième point $P * Q$.
2. Si $P \in C(K)$, nous pouvons définir $D = PP$, la droite tangente à C au point P. Grâce à la proposition précédente nous savons qu'il existe un troisième point unique (en comptant les multiplicités) qui appartient à $C \cap D$. Nous noterons ce troisième point $P * P$.

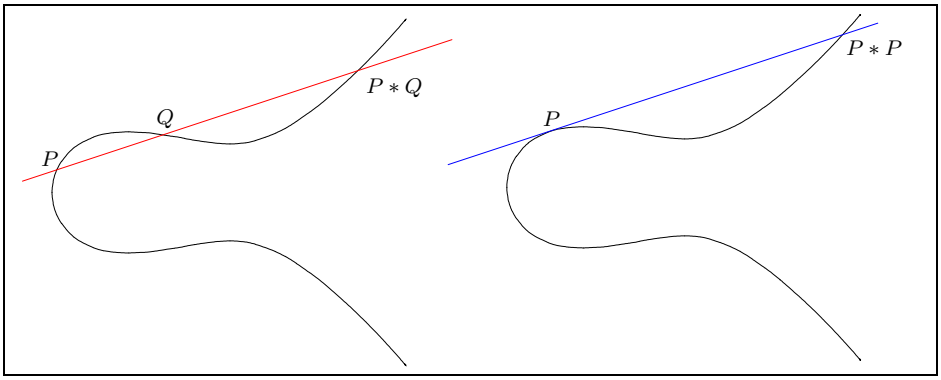


FIG. 1 – Règle de la sécante tangente

Nous pouvons constater que sur la Figure 1, une droite verticale coupant la courbe C ne semble pas la couper en un troisième point. Ce ceci est lié à la difficulté de représenter $\mathbb{P}_2$ sur un plan. Ce troisième point existe bien sur, et appartient à $\mathbb{P}_1$. Pour une courbe elliptique il correspond au point $\mathcal{O}$.

Remarque : Le meilleur moyen de considérer $\mathbb{P}_1$ est de se représenter ses éléments comme l'ensemble des directions possibles des droites du plan affine. Dans le cas particulier des courbes elliptiques, on a vu que $\mathbb{P}_1$ se limite à un seul élément, que nous avons noté $\mathcal{O}$, qui correspond à la direction des droites verticales.

PROPOSITION 4 *Soit un corps K et une cubique irréductible non singulière C. Pour tous points P_1, P_2, Q_1 et Q_2 de $C(K)$, nous avons :*

$$(P_1 * P_2) * (Q_1 * Q_2) = (P_1 * Q_1) * (P_2 * Q_2)$$

Pour une démonstration de ce résultat, voir [7].

3.2 Loi de groupe sur une courbe elliptique

THÉORÈME 2 *Soit un corps K. Soit E est une courbe elliptique définie sur K. Soient P et Q deux points de cette courbe. Alors l'opération*

$$P + Q = \mathcal{O} * (P * Q)$$

définit une structure de groupe commutatif ayant $\mathcal{O}$ comme élément neutre.

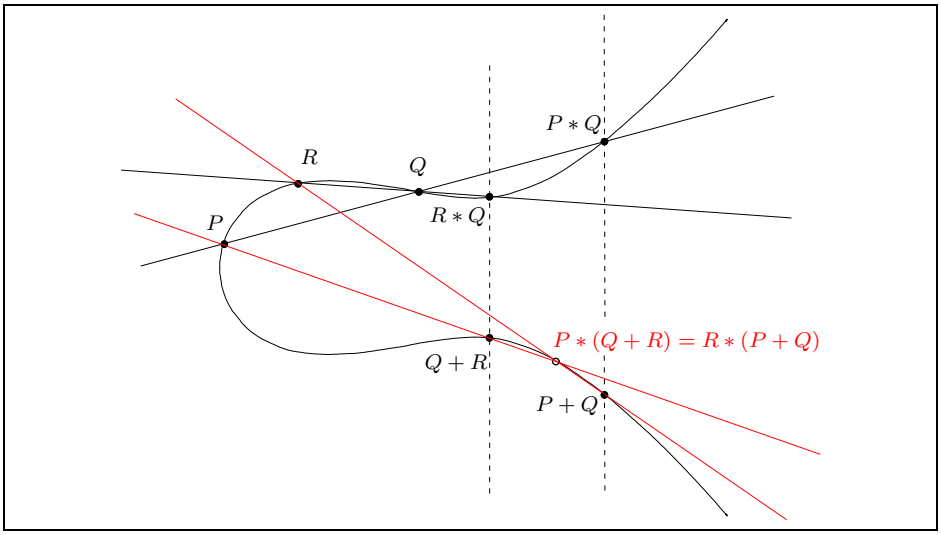


FIG. 2 – L'associativité de la loi de groupe

Preuve géométrique :

1. La loi $+$ est bien interne puisque $P + Q$ est l'intersection d'une droite et de la courbe, c'est-à-dire un point de la courbe.
2. La loi $+$ est associative (voir Figure 2). En effet, si P, Q et R sont trois points de la courbe, on a :

$$\begin{aligned} P * (Q + R) &= P * (\mathcal{O} * (Q * R)) \\ &= ((P * Q) * Q) * (\mathcal{O} * (Q * R)) \quad \text{car } P = (P * Q) * Q \\ &= ((P * Q) * \mathcal{O}) * (Q * (Q * R)) \quad \text{voir la proposition 4} \\ &= (\mathcal{O} * (P * Q)) * R) \quad \text{voir Figure 2} \\ &= (P + Q) * R \end{aligned}$$

   En appliquant $\mathcal{O}$ sur les deux membres de l'égalité, nous trouvons $P + (Q + R) = (P + Q) + R$.
3. L'élément $\mathcal{O}$ est le neutre pour la loi $+$ (voir Figure 3). En effet :

$$P + \mathcal{O} = \mathcal{O} * (P * \mathcal{O}) = P \quad \text{et} \quad \mathcal{O} + P = \mathcal{O} * (\mathcal{O} * P) = P$$

4. Tout point P possède un inverse pour la loi $+$. Vérifions que le point $-P = (\mathcal{O} * \mathcal{O}) * P$ est bien l'inverse de P :

$$\begin{aligned} P + (-P) &= \mathcal{O} * (P * ((\mathcal{O} * \mathcal{O}) * P)) = \mathcal{O} * (\mathcal{O} * \mathcal{O}) = \mathcal{O} + \mathcal{O} = \mathcal{O} \\ (-P) + P &= \mathcal{O} * (((\mathcal{O} * \mathcal{O}) * P) * P) = \mathcal{O} * (\mathcal{O} * \mathcal{O}) = \mathcal{O} + \mathcal{O} = \mathcal{O} \end{aligned}$$

5. Enfin la loi + est commutative. Si P et Q sont deux points de la courbe :

$$P + Q = \mathcal{O} * (P * Q) = \mathcal{O} * (Q * P) = Q + P$$

□

Les propriétés de la loi de groupe sur une courbe elliptique sont représentées sur la Figure 4.

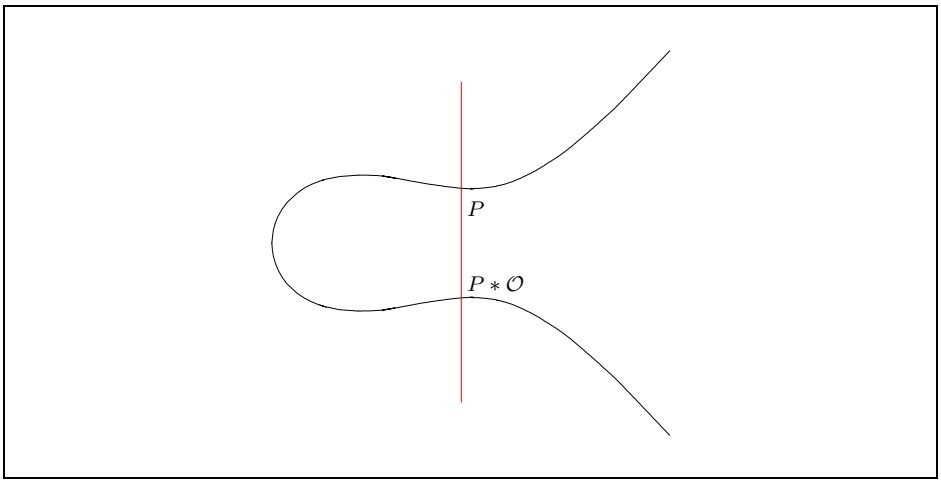


FIG. 3 – L'élément neutre de la loi de groupe

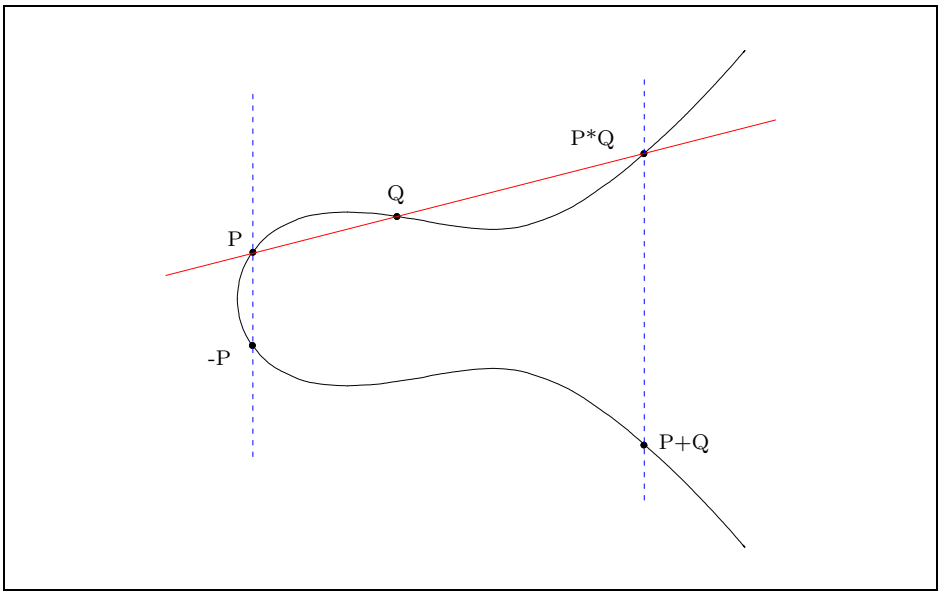


FIG. 4 – La loi de groupe + sur une courbe elliptique

4 Formules explicites

Nous allons considérer des courbes elliptiques définies sur des corps K de caractéristique $p > 3$. L'équation de Weierstrass définissant une courbe elliptique E définie sur K prend alors la forme simplifiée suivante :

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

Nous pouvons remarquer que cette courbe est symétrique par rapport à l'axe des abscisses. Dans ce paragraphe, $P = (x_P, y_P)$ et $Q = (x_Q, y_Q)$ seront deux points de notre courbe différents de $\mathcal{O}$.

4.1 Calcul de l'inverse de P

L'inverse du point P est son symétrique par rapport à l'axe des abscisses, donc :

$$\boxed{\begin{cases} x_{-P} = x_P \\ y_{-P} = -y_P \end{cases}} \tag{3}$$

4.2 Calcul de l'addition de P et Q

Considérons que $P \neq Q$, sinon additionner P et Q revient à doubler le point P.

Si $x_P \neq x_Q$:

La droite D passant par P et Q a pour équation :

$$D : \ y = \lambda x + \gamma \quad \text{avec} \quad \lambda = \frac{y_Q - y_P}{x_Q - x_P} \quad \text{et} \quad \gamma = y_P - \lambda x_P$$

Les coordonnées des points à l'intersection de la droite D et de la courbe E sont solution du système :

$$\begin{cases} y^2 = x^3 + ax + b \\ y = \lambda x + \gamma \end{cases}$$

D'où on déduit :

$$(\lambda x + \gamma)^2 = x^3 + ax + b$$

Et donc :

$$x^3 - \lambda^2 x^2 + (a - 2\lambda\gamma)x - (\gamma^2 - b) = 0 \tag{4}$$

Les coordonnées des points P, Q et $P * Q$ sont les trois solutions de notre système, l'équation (4) peut donc être écrite de la manière suivante :

$$(x - x_P)(x - x_Q)(x - x_{P*Q}) = 0$$

Ce qui donne après développement :

$$x^3 - (x_P + x_Q + x_{P*Q})x^2 + (x_P x_Q + x_P x_{P*Q} + x_Q x_{P*Q})x - x_P x_Q x_{P*Q} = 0 \tag{5}$$

En égalisant les coefficients de (4) et (5), on obtient :

$$\begin{cases} x_{P*Q} = \lambda^2 - x_P - x_Q \\ y_{P*Q} = \lambda x_{P*Q} + \gamma \end{cases}$$

En remplaçant γ par sa valeur, on obtient :

$$\begin{cases} x_{P*Q} = \lambda^2 - x_P - x_Q \\ y_{P*Q} = \lambda(x_{P*Q} - x_P) + y_P \end{cases}$$

Le point $P+Q$ est le symétrique par rapport à l'axe des abscisses du point $P*Q$, donc :

$$\boxed{\begin{cases} x_{P+Q} = \lambda^2 - x_P - x_Q \\ y_{P+Q} = \lambda(x_P - x_{P+Q}) - y_P \end{cases}} \quad (6)$$

$$\text{avec} \quad \lambda = \frac{y_P - y_Q}{x_P - x_Q}$$

Si $x_P = x_Q$:

On a supposé que $P \neq Q$. Comme $x_P = x_Q$ on a forcément $y_P \neq y_Q$. Le point Q est donc l'inverse du point P, ainsi $P + Q = \mathcal{O}$.

4.3 Calcul du doublement de P

Si $y_P \neq 0$:

La droite tangente à E passant par le point P a pour équation :

$$D : \ y = \lambda x + \gamma$$

où λ est la pente de la tangente à la courbe E en P et $\gamma = y_P - \lambda x_P$. Notons :

$$E : \ f(x, y) = y^2 - x^3 - ax - b = 0$$

Le coefficient λ est alors donné par :

$$\begin{aligned} \lambda &= -\frac{\frac{\partial f}{\partial x}(x_P, y_P)}{\frac{\partial f}{\partial y}(x_P, y_P)} \\ &= \frac{3x_P^2 + a}{2y_P} \end{aligned}$$

Le calcul est ensuite le même qu'au paragraphe précédent (il suffit de remplacer Q par P et $P*Q$ par $[2]P$). On obtient :

$$\boxed{\begin{cases} x_{[2]P} = \lambda^2 - 2x_P \\ y_{[2]P} = \lambda(x_P - x_{[2]P}) - y_P \end{cases}} \quad (7)$$

$$\text{avec} \quad \lambda = \frac{3x_P^2 + a}{2y_P}$$

Si $y_P = 0$:

La tangente en P à la courbe E est verticale et ne coupe E qu'au point P. Le point P est alors un point d'ordre 2 et on a $[2]P = \mathcal{O}$.

5 Une approche théorique de l'algorithme ECM

L'algorithme de factorisation sur une courbe elliptique présente de grandes similitudes avec l'algorithme de factorisation $p-1$ de Pollard. Avant de décrire en détail la méthode ECM, nous rappelons brièvement cet algorithme.

5.1 L'algorithme $p-1$ de Pollard

DÉFINITION 9 *Un entier n est B-lisse si tous les facteurs premiers de n sont inférieurs ou égaux à B.*
Un entier n est B-superlisse si toutes les puissances premières divisant n sont inférieures ou égales à B.

La méthode de factorisation $p-1$ de Pollard permet de factoriser un nombre n dont un facteur premier p est tel que $p-1$ est B-*superlisse*. Cela signifie que $p-1$ est un facteur de $B!$. On peut alors remarquer que :

$$\begin{aligned} x^{B!} \bmod n &= x^{k_1(p-1)} \bmod n \\ &= (x^{k_1} \bmod n)^{p-1} \bmod n \\ &= (x^{k_1} \bmod n)^{p-1} + k_2 n \end{aligned}$$

avec $k_1, k_2 \in \mathbb{Z}$ et $x \in \mathbb{Z}/n\mathbb{Z}$. Et donc, d'après le petit théorème de Fermat :

$$\begin{aligned} x^{B!} \bmod n &\equiv (x^{k_1} \bmod n)^{p-1} \pmod p \\ &\equiv 1 \pmod p \end{aligned}$$

pour peu que x soit premier avec p. L'entier p est donc un facteur de $(x^{B!} \bmod n) - 1$.

Les nombres n et $(x^{B!} \bmod n) - 1$ ont donc un facteur commun que l'on peut faire apparaître grâce à un calcul de pgcd. Pratiquement on calculera $\text{pgcd}((x^{t!} \bmod n) - 1, n)$ pour t allant de 1 à B et en s'arrêtant dès que le pgcd est différent de 1.

Dans la plupart des cas, choisir $B = \sqrt[4]{n}$ est suffisant. La complexité de l'algorithme est de l'ordre de $O(B)$ opérations arithmétiques.

5.2 La méthode ECM

La méthode ECM est une généralisation de la précédente. Plutôt que d'effectuer les calculs dans $\mathbb{Z}/n\mathbb{Z}$, nous travaillerons sur $E(\mathbb{Z}/n\mathbb{Z})$.

Nous supposons ici que le nombre n a un facteur premier $p > 3$. Nous allons travailler sur la courbe elliptique définie sur le corps $\mathbb{F}_p$ par l'équation suivante :

$$E : y^2 = x^3 + ax + b \cup \mathcal{O},$$

où x et y sont les coordonnées affines de notre courbe. La courbe $E(\mathbb{F}_p)$ doit être non singulière, c'est à dire $\Delta \neq 0$. On le vérifie en calculant $\text{pgcd}(\Delta, n)$. S'il est supérieur à 1 alors :

– soit $N \mid \Delta$. On choisi alors une autre courbe elliptique.
– soit N et Δ ont un facteur commun et le tour est joué.
On peut supposer dans la suite que $\text{pgcd}(\Delta, n) = 1$ et donc que $\Delta \bmod p \neq 0$.

Dans la méthode $p-1$ de Pollard, nous avons supposé que $p-1$ est *B-superlisse*, ce qui revient à dire que la cardinalité du groupe $\mathbb{F}_p^*$ est *B-superlisse*. Ici c'est la cardinalité de $E(\mathbb{F}_p)$ que nous supposons *B-superlisse*. L'avantage de cette méthode est que l'on peut changer plusieurs fois de courbe elliptique afin d'en trouver une telle que la cardinalité de $E(\mathbb{F}_p)$ soit *B-superlisse* pour un certain facteur p de n.

Nous considérons un point $P \neq \mathcal{O}$ de $E(\mathbb{Z}/n\mathbb{Z})$. Soit :

$$k = B! \quad \text{et} \quad m = \#E(\mathbb{F}_p)$$

La cardinalité de $E(\mathbb{F}_p)$ étant *B-superlisse*, on a $m \mid k$.

Notons x_P et y_P les coordonnées affines du point P. On a :

$$y_P^2 = x_P^3 + ax_P + b \quad \text{avec} \quad x_P,\ y_P \in \mathbb{Z}/n\mathbb{Z}$$

Comme $p \mid n$, cette équation est aussi valable dans $\mathbb{F}_p$. Donc $P \in E(\mathbb{F}_p)$.

Nous avons vu que $m \mid k$ et que $P \in E(\mathbb{F}_p)$. D'après le théorème de Lagrange, on a :

$$[k]P = \mathcal{O} \quad \text{dans} \quad E(\mathbb{F}_p)$$

La raison pour laquelle nous nous préoccupons du fait que $[k]P = \mathcal{O}$ dans $E(\mathbb{F}_p)$ est donnée au prochain paragraphe.

Comme nous ne connaissons pas le facteur p, le calcul de $[k]P$ se fera dans $E(\mathbb{Z}/n\mathbb{Z})$ et non dans $E(\mathbb{F}_p)$. Comme $\mathbb{Z}/n\mathbb{Z}$ n'est pas un corps, tout les éléments ne sont pas inversible. Ce problème peut se poser pendant le calcul de λ. Dans ce cas, et si on note α ce facteur non inversible, on a :
– soit $\alpha = 0$, dans ce cas on choisi une autre courbe elliptique,
– soit α et n on un facteur commun strictement plus grand que 1. On peut le faire apparaître en effectuant un calcul de pgcd.
Enfin, si le calcul ne donne aucun résultat, nous choisissons une nouvelle courbe elliptique.

5.3 Pourquoi chercher k tel que $[k]P = \mathcal{O}$ dans $E(\mathbb{F}_p)$?

LEMME 1 *Soient P et Q deux points de la courbe $E(\mathbb{F}_p)$ tels que $P+Q = \mathcal{O}$. Alors, si $p \mid n$, le point $P+Q$ n'est pas défini sur $E(\mathbb{Z}/n\mathbb{Z})$.*

En effet, si $P + Q = \mathcal{O}$ sur $E(\mathbb{F}_p)$ on sait que $x_P \equiv x_Q \pmod p$, donc que $x_P - x_Q \equiv 0 \pmod p$. Comme $x_P \neq x_Q$, cela signifie que $p \mid x_P - x_Q$. Si p est un facteur de n, cela implique que $x_P - x_Q$ est non inversible modulo n (car $\text{pgcd}(x_P - x_Q, n) > 1$) et donc que le calcul de λ échoue dans $\mathbb{Z}/n\mathbb{Z}$. Le point $P+Q$ n'est donc pas défini sur $E(\mathbb{Z}/n\mathbb{Z})$.

LEMME 2 *Soit P un point de la courbe $E(\mathbb{F}_p)$ tel que $[2]P = \mathcal{O}$. Alors, si $p \mid n$, le point $[2]P$ n'est pas défini sur $E(\mathbb{Z}/n\mathbb{Z})$.*

De la même manière, si $[2]P = \mathcal{O}$ sur $E(\mathbb{F}_p)$ on sait que $y_P \equiv 0 \pmod{p}$. Si p est un facteur de n cela signifie que y_P est non inversible modulo n et donc que $[2]P$ n'est pas défini sur $E(\mathbb{Z}/n\mathbb{Z})$.

Les deux lemmes précédents nous permettent de conclure que si l'on connaît k tel que $[k]P = \mathcal{O}$ sur $E(\mathbb{F}_p)$, effectuer ce calcul sur $E(\mathbb{Z}/n\mathbb{Z})$ peut suffire à faire apparaître un terme non inversible qui permettra de déduire un facteur de n. Il ne suffit pas obligatoirement, par exemple lorsque le terme non inversible est nul modulo n.

6 Une approche pratique de l'algorithme ECM

Nous venons de parcourir l'essentiel de la méthode. Nous allons maintenant présenter un algorithme qui permettra de la mettre en oeuvre.

6.1 L'algorithme

L'algorithme que nous allons étudier est donné en Figure 5. Le choix correct de la borne B est décrit au paragraphe 6.3.

Nous allons tâcher de décortiquer cet algorithme étape par étape.

Lignes 1 à 2

La première courbe sur laquelle nous allons travailler a pour équation $y^2 = x^3 + 1$. Nous pouvons remarquer que cette courbe est non singulière puisque $\Delta = -16 \cdot 27 = -2^4 \cdot 3^3$ et que N est supposé premier avec $6 = 2 \cdot 3$. Le premier point choisi est $P = (0, 1)$. Remarquons que ce point appartiendra toujours aux courbes dont l'équation est du type $y^2 = x^3 + ax + 1$ et ce, quelle que soit la valeur de a. C'est pour cette raison que dès que nous changerons de courbe, nous choisirons toujours ce point pour commencer les calculs.
Dans la suite i sera un pointeur parcourant $[0, k]$.

Lignes 3 à 5

Lorsque $i > k$, tous les nombre premiers plus petits que la borne B choisie ont été parcourus. On choisi alors une nouvelle courbe elliptique, on réinitialise le point P à $(0, 1)$ et on recommence.

Lignes 6 à 8

Le nombre q est initialisé à la valeur du nombre premier en position i dans la table pré-calculée.
Le nombre l permet d'effectuer le calcul sur q_1 en ligne 8. A la fin de la boucle, nous aurons $q_1 = q^s$ avec s tel que $q^{s-1} \leq l < q^s$, c'est-à-dire $q_1 \leq B < q \cdot q_1$. Autrement dit, q_1 est la plus grande puissance du nombre premier q plus petite que B. Nous noterons

ALGORITHME 6.1 (ECM) Soit N un entier composé, premier avec 6 et B une borne correctement choisie. L'algorithme proposé trouve un facteur p de N si le cardinal de $E(\mathbb{F}_p)$ est B−*superlisse*. Il suppose l'existence d'une table pré-calculée $p[1], \ldots, p[k]$ de tous les premiers jusqu'à B.

[Initialisation de la courbe]
1. $a \leftarrow 0$ et soit E la courbe $y^2 = x^3 + ax + 1$

[Initialisation du point]
2. $P \leftarrow (0,1)$ et $i \leftarrow 0$

[Nombre Premier suivant]
3. $i \leftarrow i + 1$
4. Si $i > k$
5. $\quad a \leftarrow a + 1$ et aller à [Initialisation du point]
6. Sinon
7. $\quad q \leftarrow p[i]$, $q_1 \leftarrow q$ et $l \leftarrow \lfloor B/q \rfloor$

[Multiplication du point]
8. Tant que $q_1 \leq l$, Faire $q_1 \leftarrow q \cdot q_1$
9. Calculer $P \leftarrow [q_1]P$
10. Si le calcul réussit, aller à [Nombre Premier suivant]

[Fin]
11. Le calcul a échoué. Soit α l'élément non inversible.
12. $g = \mathrm{pgcd}(\alpha, N)$
13. Si $g < N$, afficher g et terminer
14. Sinon, faire $a \leftarrow a + 1$ et aller à [Initialisation du point]

FIG. 5 – L'algorithme de factorisation ECM de Lenstra

cette puissance s_i.

Lignes 9 à 10

C'est le point central de l'algorithme, où on effectue les additions successives du point P. Nous avons supposé que le cardinal m de $E(\mathbb{F}_p)$ est B−*superlisse*. Décomposons m en produit de puissances premières :

$$m = p_1^{n_1} \cdot \ \ldots \ \cdot p_r^{n_r}$$

Dans l'algorithme, nous calculons progressivement le point :

$$[p[1]^{s_1} \cdot \ \ldots \ \cdot p[k]^{s_k}]P$$

Le choix des puissances s_i nous permet d'affirmer que :

$$p_1^{n_1} \cdot \ \ldots \ \cdot p_r^{n_r} \quad | \quad p[1]^{s_1} \cdot \ \ldots \ \cdot p[k]^{s_k}$$

et donc, grâce au théorème de Lagrange, que :

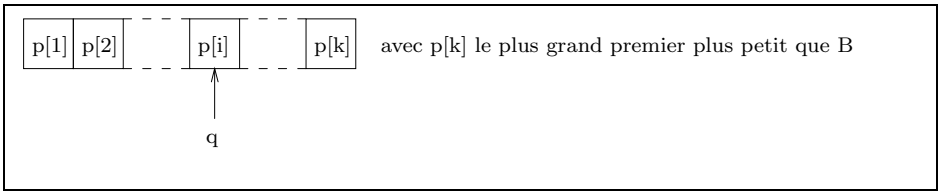


FIG. 6 – La table pré-calculée de tous les premiers plus petits que B

$$[p[1]^{s_1} \cdot \ \ldots \ \cdot p[k]^{s_k}]P = \mathcal{O}$$

Il est très probable que l'algorithme s'arrête avant d'arriver à une telle valeur. Il suffit pour cela que tous les premiers présents dans la décomposition de m soient présents dans la décomposition du facteur multiplicatif du point P.

Si nous avions tout simplement appliqué la méthode théorique, nous aurions calculé $[B!]P$. Le calcul de ce point nous aurait amené à effectuer un certain nombre de calculs inutiles, i.e. calculer par exemple $[p[i]^t]\,P$ où $t > s_i$.

Lignes 11 à 14

Si le calcul précédent a échoué, nous avons forcément trouvé une valeur α non inversible modulo N. Si cette valeur est plus petite que N nous pouvons faire apparaître un facteur non trivial de N grâce à un calcul de pgcd. Dans le cas contraire nous choisissons une nouvelle courbe elliptique, réinitialisons le point P à $(0, 1)$ et recommençons.

6.2 Une légère modification

Nous pouvons constater que l'algorithme que nous venons de présenter peut être accéléré, au prix d'un besoin en taille de stockage plus grand. Notre version de l'algorithme conserve la même valeur de B pendant toute la durée de son exécution, l'ensemble des valeurs successivement prises par q_1 peuvent alors être stockées une bonne fois pour toutes, et réutilisées pour chacune des courbes que nous allons parcourir. L'algorithme modifié est présenté en Figure 7.

Dans la nouvelle version de l'algorithme, nous avons tout simplement remplacé q_1 par un tableau $q_1[]$ qui conserve l'ensemble des valeurs possibles prises par q_1. Nous avons aussi supprimé la variable q qui était devenue inutile.

6.3 Le calcul de la borne B et la complexité de l'algorithme

Dans la première version de la méthode ECM, nous n'avons donné aucune information sur l'ordre de grandeur de B. Nous présentons ici un théorème qui permet d'en trouver la valeur optimale.

THÉORÈME 3 *(Canfield, Erdös, Pomerance) Soit :*

$$\psi(x, y) = \ \mid \{n \leq x, n \textit{ est } y\textit{-lisse}\} \mid$$

ALGORITHME 6.2 (ECM modifié) Soit N un entier composé, premier avec 6 et B une borne correctement choisie. L'algorithme proposé trouve un facteur p de N si le cardinal de $E(\mathbb{F}_p)$ est B−*superlisse*. Il suppose l'existence d'une table pré-calculée $p[1], \ldots, p[k]$ de tous les premiers jusqu'à B.

[Initialisation du tableau de puissances premières]
1. Pour i allant de 1 à k Faire
2. $q_1[i] \leftarrow p[i]$ et $l[i] \leftarrow \lfloor B/p[i] \rfloor$
3. Tant que $q_1[i] \leq l[i]$, Faire $q_1[i] \leftarrow p[i] \cdot q_1[i]$

[Initialisation de la courbe]
4. $a \leftarrow 0$ et soit E la courbe $y^2 = x^3 + ax + 1$

[Initialisation du point]
5. $P \leftarrow (0,1)$ et $i \leftarrow 0$

[Nombre Premier suivant]
6. $i \leftarrow i + 1$
7. Si $i > k$
8. $a \leftarrow a + 1$ et aller à [Initialisation du point]

[Multiplication du point]
9. Calculer $P \leftarrow [q_1[i]]\, P$
10. Si le calcul réussit, aller à [Nombre Premier suivant]

[Fin]
11. Le calcul a échoué. Soit α l'élément non inversible.
12. $g = \mathrm{pgcd}(\alpha, N)$
13. Si $g < N$, afficher g et terminer
14. Sinon, faire $a \leftarrow a + 1$ et aller à [Initialisation du point]

FIG. 7 – L'algorithme de factorisation ECM de Lenstra modifié

Si nous posons $u = \ln x / \ln y$, *alors :*

$$\psi(x, y) = x u^{-u(1+o(1))}$$

de manière uniforme pour $x \to \infty$ *et si* $(\ln x)^\epsilon < u < (\ln x)^{(1-\epsilon)}$ *pour un certain* $\epsilon \in]0, 1[$. *En particulier, si nous posons :*

$$L(x) = e^{\sqrt{\ln x \ln \ln x}}$$

alors :

$$\psi(x, L(x)^a) = x L(x)^{-1/2a+o(1)}$$

où $o(1)$ *est un terme qui tend vers 0 lorsque* x *tend vers l'infini.*

Nous avons vu que concernant la méthode ECM, nous cherchions à trouver un groupe $E(\mathbb{F}_p)$ dont la cardinalité serait *B-superlisse*. Si nous notons $g_1 \geq g_2 \geq \ldots$ les facteurs premier de l'ordre du groupe $E(\mathbb{F}_p)$ l'algorithme fonctionne la plupart du temps lorsque $g_1 \leq B$. En effet, remarquons que si la multiplicité d'un de ces facteurs dans l'ordre du

groupe est trop élevée, l'algorithme ne donnera aucun résultat. Nous allons considérer (voir [3] p.6) que cette éventualité reste négligeable et donc que l'algorithme ECM reste efficace si on se limite à chercher un groupe $E(\mathbb{F}_p)$ d'ordre *B-lisse*.

D'après le théorème de Hasse, il y a environ p points sur E. On peut alors s'attendre à ce que cette cardinalité soit $L(p)^a$*-lisse* avec une probabilité $L(p)^{-1/(2a)+o(1)}$. Si nous fixons $B = L(p)^a$, une courbe sera B$-$*lisse* avec une probabilité $L(p)^{-1/(2a)+o(1)}$. Il faudra donc en moyenne $L(p)^{1/(2a)+o(1)}$ courbes pour en trouver une adéquate (c'est une loi géométrique, autrement dit une loi du temps de premier succès). En considérant que nous effectuerons environ B opérations par courbe (comme pour la méthode $p-1$ de Pollard), il faudra environ $L(p)^{a+1/(2a)+o(1)}$ opérations au total. Cette fonction est minimale pour $a = 1/\sqrt{2}$. Dans cette optique, nous constatons que la meilleure borne B possible est $L(p)^{1/\sqrt{2}}$. La complexité de l'algorithme sera donc de l'ordre de $L(p)^{\sqrt{2}+o(1)}$ opérations de groupe.

Comme le nombre p n'est pas connu à priori, il est encore difficile d'évaluer la borne B. Si on considère $p \approx \sqrt{N}$, on obtient :

$$\begin{aligned} L(p)^{1/\sqrt{2}} &= \exp\sqrt{1/2 \cdot \ln\sqrt{N} \cdot \ln\ln\sqrt{N}} \\ &= \exp\sqrt{1/4 \cdot \ln N \cdot \ln(1/2 \cdot \ln N)} \\ &= \exp\sqrt{1/4 \cdot \ln N \cdot (\ln\ln N - \ln 2)} \\ &\approx L(N)^{1/2} \end{aligned}$$

On pourra finalement choisir

$$\boxed{B = L(p)^{1/\sqrt{2}} \approx L(N)^{1/2}}$$

pour obtenir une complexité en

$$\boxed{O(e^{(\sqrt{2}+o(1))\sqrt{\ln p \ln\ln p}})}$$

En écrivant la complexité sous cette forme, on constate que l'algorithme n'est pas sensible à la taille de N mais plutôt à **la taille de** p, le plus petit facteur premier de N. On considérant une fois encore $p \approx \sqrt{N}$, cette complexité peut s'écrire :

$$\boxed{O(e^{(1+o(1))\sqrt{\ln N \ln\ln N}})}$$

7 Le calcul du point $[k]P$

Dans ce paragraphe nous étudions plusieurs algorithmes de multiplication de point sur une courbe elliptique. Leur complexité est donnée en terme de nombre moyen d'addition de points (A) et de doublement de points (D).

7.1 La *Binary Method*

La première méthode est la plus simple. On considère la décomposition binaire de k (entier de l bits).

ALGORITHME 7.1

ENTRÉE : Un point P et un entier de l bits $k = \sum_{j=0}^{l-1} k_j 2^j$
SORTIE : $Q = [k]P$

1. $Q \leftarrow \mathcal{O}$
2. For $j = l - 1$ downto 0
3. $\quad Q \leftarrow [2]Q$
4. $\quad$ If $k_j = 1$ then $Q \leftarrow Q + P$
5. Return Q

FIG. 8 – Binary Method

Pour en comprendre le principe, considérons le cas $l = 4$. On a :

$$k = 2^3 k_3 + 2^2 k_2 + 2k_1 + k_0$$

D'après la loi de Hörner, on a :

$$\begin{aligned} [k]P &= [2^3 k_3 + 2^2 k_2 + 2k_1 + k_0]P \\ &= [2]([2]([2k_3]P + [k_2]P) + [k_1]P) + [k_0]P \end{aligned}$$

En suivant l'algorithme pas à pas, on a :
– Après $j = 3 : Q = [k_3]P$
– Après $j = 2 : Q = [2k_3]P + [k_2]P$
– Après $j = 1 : Q = [2]([2k_3]P + [k_2]P) + [k_1]P$
– Après $j = 0 : Q = [2]([2]([2k_3]P + [k_2]P) + [k_1]P) + [k_0]P$

On a donc à la fin de l'algorithme $Q = [k]P$.
Si W est le nombre de 1 dans la décompositions binaire de k, cette méthode requiert W additions et $l - 1$ doublements. En considérant qu'en moyenne $W = l/2$, le nombre estimé d'opérations sur la courbe est $lA + 0.5lD$, soit $1,5l$ opérations.

7.2 La *m-ary Method*

Cette méthode est une généralisation de la précédente. Elle considère un développement de k en base $m = 2^r$. Le cas précédent correspond à $r = 1$.

L'initialisation est claire, on a $P_i = [i]P$.

Donnons à nouveau un exemple pour $d = 4$. On a :

$$k = k_3 m^3 + k_2 m^2 + k_1 m + k_0$$

D'après la loi de Hörner :

ALGORITHME 7.2

ENTRÉE : Un point P et un entier $k = \sum_{j=0}^{d-1} k_j m^j$
avec $k_j \in \{0, 1, \ldots, m-1\}$
SORTIE : Un point $Q = [k]P$

[Initialisation]
1. $P_1 \leftarrow P$
2. For $i = 2$ downto $m-1$ $P_i \leftarrow P_{i-1} + P$ (On a $P_i = [i]P$)
3. $Q \leftarrow \mathcal{O}$.

[Boucle principale]
4. For $j = d-1$ downto 0
5. $Q \leftarrow [m]Q$ (nécessite r doublements)
6. $Q \leftarrow Q + P_{k_j}$
7. Return Q

FIG. 9 – m-ary Method

$$\begin{aligned}
[k]P &= [k_3m^3 + k_2m^2 + k_1m + k_0]P \\
&= [m]([m]([k_3m]P + [k_2]P) + [k_1]P) + [k_0]P \\
&= [m]([m]([m]P_{k_3} + P_{k_2}) + P_{k_1}) + P_{k_0} \quad \text{car} \quad [i]P = P_i
\end{aligned}$$

En suivant la boucle principale de l'algorithme, on obtient :
– Après $j = 3 : Q = P_{k_3}$
– Après $j = 2 : Q = [m]P_{k_3} + P_{k_2}$
– Après $j = 1 : Q = [m]([m]P_{k_3} + P_{k_2}) + P_{k_1}$
– Après $j = 0 : Q = [m]([m]([m]P_{k_3} + P_{k_2}) + P_{k_2}) + P_{k_0}$
A la fin de l'algorithme $Q = [k]P$.

L'initialisation requiert $m-2$ additions, la boucle principale en requiert d. A ceci, il faut rajouter $(d-1)r$ doublements. Nous obtenons comme complexité $(m-2+d)A + ((d-1)r)D$.

8 Comment paralléliser l'algorithme

8.1 Mise en parallèle des calculs d'inverses

Nous avons vu au paragraphe 4 que chaque addition ou doublement de point requiert une inversion modulaire. Dans l'algorithme de Lenstra, toutes les opérations (et donc toutes les inversions) se font modulo le même N et ce, quelle que soit la courbe sur laquelle sont faits les calculs. L'optimisation proposée par P. Montgomery profite de cette remarque pour permettre à l'algorithme de réduire le nombre d'inversions effectuées en travaillant sur plusieurs courbes en parallèle.
Voici l'explication de l'algorithme 8.1 :

Lignes 1 à 3

A la fin de cette phase, on a clairement $c_i = a_1 \cdot a_2 \cdot \cdots \cdot a_i$ pour $i = 1, 2, \ldots, k$.

ALGORITHME 8.1 On considère un entier N et k entiers $a_1, \ldots, a_k$ non divisibles par N. A sa sortie, cet algorithme fournit les inverses $b_1, \ldots, b_k$ des a_i quand cela est possible. Dans le cas contraire il fournit un facteur non trivial de N.

```
[Calcul initial]
 1. c_1 ← a_1
 2. Pour i = 2, ..., k Faire
 3.    c_i ← c_{i-1} · a_i mod N

[Utiliser algorithme d'Euclide]
 4. Calculer (u, v, d) tel que uc_k + vN = d avec d = pgcd(c_k, N)
 5. Si d = 1 aller à [Calcul des inverses]
 6. Si d = N Faire
 7.    d ← pgcd(a_i, N) pour i = 1, ..., k jusqu'à ce que d > 1
 8. Afficher d comme un diviseur non trivial de N et terminer

[Calcul des inverses]
 9. Pour i = k, k − 1, ..., 2 Faire
10.    b_i ← uc_{i-1} mod N
11.    u ← ua_i mod N
12. b_1 ← u
13. Afficher les b_i et terminer
```

FIG. 10 – Calcul d'inverses modulaires en parallèle

Lignes 4 à 8

C'est le seul calcul d'inverse que nécessite cet algorithme pour calculer les k inverses nécessaires. Trois cas peuvent se présenter :

– Si $d = 1$ on a :

$$u \equiv c_k^{-1} \equiv (a_1 \cdot \cdots \cdot a_k)^{-1} \equiv a_1^{-1} \cdot \cdots \cdot a_k^{-1} \pmod{N}$$

– Si $d = N$, cela signifie que N divise $a_1 \cdot \cdots \cdot a_k$. Comme aucun des a_i n'est divisible par N, il existe forcément un a_i tel que $1 < \text{pgcd}(a_i, N) < N$. Ce pgcd est un facteur non trivial de N. On l'affiche avant de terminer l'algorithme.

– Si $1 < d < N$, on a trouvé un facteur non trivial de N que l'on affiche avant de terminer l'algorithme.

Lignes 9 à 13

Le premier passage dans la boucle permet de calculer b_k. En effet :

$$b_k \equiv uc_{k_1} \equiv (a_1^{-1} \cdot \cdots \cdot a_{k-1}^{-1} \cdot a_k^{-1}) \cdot (a_1 \cdot \cdots \cdot a_{k-1}) \equiv a_k^{-1} \pmod{N}$$

Le nouveau u calculé vaudra ensuite :

$$u \leftarrow ua_k \equiv a_1^{-1} \cdot \cdots \cdot a_{k-1}^{-1} \cdot a_k^{-1} \cdot a_k = a_1^{-1} \cdot \cdots \cdot a_{k-1}^{-1} \pmod{N}$$

Un simple récurrence permet de constater qu'à l'étape i de la boucle, on calcule $b_i \equiv a_i^{-1} \pmod{N}$ et $u = a_1^{-1} \cdot \cdots \cdot a_{i-1}^{-1}$. A la ligne 13, on peut afficher tous les $b_i \equiv a_i^{-1} \pmod{N}$ pour $i = 1, \ldots, k$.

8.2 Avantage de la méthode

Nous avons vu au paragraphe 4 que pour additionner deux points P et Q il faut au total une inversion et deux multiplications modulaires lorsque $P \neq Q$, soit une inversion et trois multiplications modulaires lorsque $P = Q$. Nous allons considérer que K multiplications sont équivalentes à une inversion. D'après [2], lorsqu'on utilise l'amélioration proposée par Lehmer (ce qui est le cas de la librairie utilisée pour notre implémentation), on obtient $K \approx 8$. Nous considérerons donc qu'une inversion (c'est-à-dire un calcul de pgcd étendu) est équivalente en terme de temps de calcul à huit multiplications.

Pour calculer k inverses, l'algorithme 8.1 nécessite une inversion et environ $3k$ multiplications, soit l'équivalent en temps de calcul de $3k + 8$ multiplications. Sans utiliser cet algorithme, il faudrait k inversions, équivalentes à $8k$ multiplications. L'avantage de la méthode est évidente puisqu'il suffirait d'utiliser $k = 2$ (c'est-à-dire d'utiliser deux courbes en parallèle) pour commencer accélérer les calculs d'inverses.

En pratique, pour que cette accélération soit assez importante, on choisira $k \geq 10$.

9 Une deuxième phase pour l'algorithme

9.1 L'objectif et l'idée

La 2ème phase que nous allons proposer ici permet de réduire la condition sur l'ordre m du groupe $E(\mathbb{F}_p)$ à la suivante : plutôt que de considérer que m doit être B$-$*superlisse*, il suffira que m s'écrive sous la forme $f \cdot q$, où f serait B$-$*superlisse* et q un premier compris entre B et B^2. Il est clair que dans ce cas, m est $B^2 -$ *superlisse* et qu'il suffirait de choisir B^2 à la place de B dans l'algorithme précédent. Mais nous avons vu que le temps passé sur chaque courbe est sensible à la taille de B qu'il vaut mieux garder assez petit, c'est tout l'intérêt de cette deuxième phase.

A la fin d'un calcul effectué sur une courbe à l'algorithme précédent et s'il ne s'est pas terminé, nous avons calculé le point $[p[1]^{s_1} \cdot \cdots \cdot p[k]^{s_k}]P$. Nous ne supposerons plus que $m \mid p[1]^{s_1} \cdot \cdots \cdot p[k]^{s_k}$ mais plutôt que $f \mid p[1]^{s_1} \cdot \cdots \cdot p[k]^{s_k}$ où f est tel que $m = f \cdot q$ avec q un premier plus grand que B. En d'autres termes, nous supposons ici que f est B_1-*superlisse* où B_1 sera notre ancien B. Le nombre q sera compris entre B_1 et B_2, où B_2 est une nouvelle borne utile pour la deuxième que nous présentons ici.

Si nous notons $P \leftarrow [p[1]^{s_1} \cdot \cdots \cdot p[k]^{s_k}]P$, il nous reste donc à calculer $[q]P$. Il suffit pour cela de calculer successivement $[q]P$ pour q parcourant les premiers entre B_1 et B_2. L'algorithme que nous allons voir dans le paragraphe suivant propose une astuce qui nous permettra de remplacer de lourds calculs de doublement de points par de simples additions.

ALGORITHME 9.1 (ECM 2ème phase) Soit N un nombre composé, B_1 et B_2 deux bornes bien choisies (où B_1 correspond à notre précédent B). Cet algorithme trouve un facteur p de N si $m = \#E(\mathbb{F}_p)$ est égal à un produit $f \cdot q$, où f est B_1-*superlisse* et q est un premier plus petit que B_2. Nous avons calculé au préalable un tableau $p[1], \ldots, p[k_1]$ de tous les premiers jusqu'à B_1 ainsi qu'un tableau $d[1], \ldots, d[k_2]$ des différences entre les premiers successifs allant de B_1 à B_2, i.e. $d[i] = p[k_1 + i] - p[k_1 + i - 1]$ pour $i \in [1, k_2]$. Les calculs sont effectués sur la courbe $y^2 = x^3 + ax + 1$.
Si la première phase n'a donné aucun résultat, nous avons obtenu un point P à la fin des calculs effectués sur une courbe.

[Calcul initial]
1. Pour i allant de 1 à k_2 Faire
2. $\quad Q[i] \leftarrow [d[i]]P$
3. $\widetilde{P} \leftarrow [p[k_1]]P$ et $i \leftarrow 0$

[Boucle principale]
4. $i \leftarrow i + 1$
5. Si $i > k_2$, faire $a \leftarrow a + 1$ et retourner à [Initialisation du point] dans l'algorithme 6.2.
6. Essayer de calculer $Q[i] + \widetilde{P}$
7. Si le calcul réussi aller à [Boucle principale]

[Fin]
8. Le calcul a échoué. Soit α l'élément non inversible.
9. $g = \mathrm{pgcd}(\alpha, N)$
10. Si $g > N$, faire $a \leftarrow a+1$ et aller à [Initialisation du point] dans l'algorithme 6.2.
11. Sinon afficher g et terminer

FIG. 11 – Une deuxième phase pour l'algorithme ECM

9.2 La deuxième phase de l'algorithme ECM

Le deuxième stage de l'algorithme est présenté en Figure 11.
Détaillons cet algorithme.

Ligne 1

Nous essayons de factoriser N en posant $B = B_1$. Si l'algorithme trouve un facteur de N le tour est joué. S'il n'en trouve pas après avoir effectué tous les calculs sur une courbe, nous appliquons la deuxième phase avant de changer éventuellement de courbe.

Lignes 2 à 4

Nous calculons un tableau de points $Q[i]$ et un point initial P. Nous verrons dans la suite comment les exploiter. Remarquons simplement que les différences entre nombre premiers $d[i]$ sont de petits nombres, les doublements de points s'effectuent donc très ra-

pidement. Le doublement effectué en ligne 4. est le seul doublement de point un peu lourd.

Lignes 5 à 8

La première fois qu'on passe par cette boucle, on essaie de calculer $Q[1]+\widetilde{P}$, c'est à dire :

$$\begin{aligned} Q[1]+\widetilde{P} &= [d[1]]P + [p[k_1]]P \\ &= [p[k_1+1]-p[k_1]]P + [p[k_1]]P \\ &= [p[k_1+1]]P \end{aligned}$$

Nous avons remplacé un lourd doublement de point par une simple addition ! Ainsi, en passant successivement par cette boucle et en calculant $Q[i]+\widetilde{P}$, on calcule en fait $[[p[k_1+1]]P, [p[k_1+2]]P, \ldots, [p[k_1+k_2]]P$. Si le nombre premier q que nous avons considéré au départ est bien compris entre B_1 et B_2, un de ces résultats correspondra forcément à $[q]P$.

Lignes 9 à 12

La fin de l'algorithme est exactement la même que dans la phase précédente.

9.3 Le choix de B_1 et B_2 pour la deuxième phase

Pratiquement il semble judicieux de choisir la même valeur pour B_1 que celle choisie pour l'algorithme à une seule phase. La valeur de B_2 déterminera le temps que l'on passe dans la phase 2 de l'algorithme, i.e. de la taille de k_2. On peut remarquer que k_2 correspondra au nombre de premiers compris entre B_1 et B_2 soit approximativement :

$$k_2 \approx \frac{B_2}{\ln B_2} - \frac{B_1}{\ln B_1}$$

Selon [5], un bon choix expérimental est $B_2 \approx 100 \cdot B_1$.

10 Une deuxième phase du type *Paradoxe des anniversaires*

10.1 Explications préliminaires

Il existe plusieurs alternatives à la phase 2 proposée au paragraphe 9. Celle que nous allons étudier ici fut proposée par Richard P. Brent dans [2].

A l'issue de la phase 1, nous avons à notre disposition un point
$Q = [p[1]^{s_1} \cdot \cdots \cdot p[2]^{s_k}]P$. Nous allons considérer le groupe cyclique H engendré par Q. Soit $f : H \to H$ une fonction pseudo-aléatoire. Construisons (de manière analogue à la méthode rho de Pollard) la suite de point suivante :

$$\left\{ \begin{array}{lcl} Q_1 & = & Q \\ Q_{i+1} & = & f(Q_i) \quad \text{pour } i = 2, \ldots, r \end{array} \right.$$

ALGORITHME 10.1 Soit N un nombre composé. Au début de cet algorithme, nous disposons d'un point Q, d'une borne r et d'une fonction f pseudo-aléatoire.

[Calcul initial]
1. $Q_1 \leftarrow Q$
2. Pour $i = 2, \ldots, r$ Faire
3. $\quad Q_i \leftarrow f(Q_{i-1})$

[Calcul de d]
4. $d \leftarrow 1$
5. Pour $i = 1, \ldots, r-1$
6. $\quad$ Pour $j = i+1, \ldots, r$
7. $\quad\quad d \leftarrow d \cdot (x_i - x_j) \bmod N$

[Test]
8. $g \leftarrow \mathrm{pgcd}(d, N)$
9. Si $1 < g < N$, afficher g et terminer. Sinon, retour à la phase 1 de l'algorithme.

FIG. 12 – Une deuxième phase du type *paradoxe des anniversaires*

Choisissons par exemple :

$$f(Q_i) = \begin{cases} 2Q_i & \text{avec probabilité } 1/2 \\ 2Q_i + Q & \text{avec probabilité } 1/2 \end{cases}$$

Les points $Q_1, Q_2, \ldots, Q_r$ que nous venons de générer peuvent être considérés comme des points aléatoires de H, le coût de ce calcul est en $O(3r/2)$ opérations de groupe. Notons $Q_i = (x_i, y_i)$ et calculons :

$$d = \prod_{i=1}^{r-1} \prod_{j=i+1}^{r} (y_i - y_j) \bmod N$$

Le calcul de d requiert $\frac{(r-2)(r-1)}{2}$ multiplications modulaires.

Si pour un certain i et un certain j tels que $i < j \leq r$ on a $Q_i = Q_j$ dans $E(\mathbb{F}_p)$, alors $p \mid d$. Pour trouver p, il suffit alors de calculer $\mathrm{pgcd}(d, N)$.

La probabilité que $p \mid d$ est égale à la probabilité qu'au moins deux des points parmi les r considérés soient égaux dans $E(\mathbb{F}_p)$. C'est ici qu'intervient le paradoxe des anniversaires. Approximons le nombre de points de H par le nombre de points de $E(\mathbb{F}_p)$, que nous approximons à son tour par p (en utilisant le théorème de Hasse). Nous savons que, pour $r << p$ nous avons :

$$\mathrm{Prob}(p \mid d) \approx 1 - \exp\left(-\frac{r^2}{2p}\right) \quad ,$$

où $\mathrm{Prob}(p \mid d)$ est la probabilité que p divise d.
Choisir de définir :

$$d = \prod_{i=1}^{r-1} \prod_{j=i+1}^{r} (x_i - x_j) \bmod N$$

revient à identifier tout point de H avec son inverse, et donc à diviser l'ordre du groupe par deux. La probabilité de succès est alors ramenée à :

$$\text{Prob}(p \mid d) \approx 1 - \exp\left(-\frac{r^2}{p}\right)$$

10.2 L'algorithme

L'algorithme est présenté en Figure 12. La fonction f peut être celle décrite au paragraphe précédent.

11 Une (autre) deuxième phase du type *Paradoxe des anniversaires*

11.1 Le déroulement de la deuxième phase

Il existe plusieurs alternatives à la phase 2 proposée au paragraphe 9. Celle que nous allons étudier ici fut proposée par Richard P. Brent dans [3]. Dans ce paragraphe nous noterons $B_3 = \pi(B_2) - \pi(B_1)$, le nombre de premiers entre B_1 et B_2, ces deux derniers entiers étant définis au paragraphe 9.

A l'issue de la phase 1, nous avons à notre disposition un point
$Q = [p[1]^{s_1} \cdot \cdots \cdot p[2]^{s_k}]P$. Considérons aussi un paramètre entier e et un tableau de taille T de points Q_j tels que :

$$Q_j = [q_j^e]Q \quad \text{pour} \quad j = 1, \ldots, T$$

Le nombre q_j pourra être calculé de la façon suivante :

$$q_j = k_0 + k_1 j$$

où k_0 et k_1 sont deux nombres pseudo-aléatoires de taille raisonnable.

Générons encore $\lfloor B_3/T \rfloor$ nouveaux points $\overline{Q_k} = [2^{e \cdot k}]Q$. Pour chacun de ces points nous vérifions si $\overline{Q_k} = \pm Q_j$ pour tout j de $1, \ldots, T$. Si nous notons $\overline{Q_k} = (\overline{x_k}, \overline{y_k})$ et $Q_j = (x_j, y_j)$, il suffit de calculer $\text{pgcd}(\overline{x_k} - x_j, N)$. Il est ainsi éventuellement possible de faire apparaître un facteur non trivial de N.

L'algorithme est proposé en Figure 13.

11.2 Implémentation et chances de succès

Pour réaliser cet algorithme il suffira de stocker $O(T)$ points. En effet il n'est pas nécessaire de conserver les $\overline{Q_k}$, le test pouvant être effectué au fure et à mesure.

ALGORITHME 11.1 Soit N un nombre composé. Au début de cet algorithme, nous disposons d'un point Q, de deux nombres aléatoires k_0 et k_1. Le nombre T est choisi tel que $T \approx \sqrt{B_3}$, avec $B_3 \approx \pi(B_2) - \pi(B_1)$. Le nombre e est choisi comme proposé au paragraphe 11.2.

[Calcul du tableau de points]
1. $q_1 \leftarrow k_0$
2. Pour $j = 1, \ldots, T$ Faire
3. $q_1 \leftarrow q_1 + k_1$, $q_2 = q_1^e$
4. $Q_j \leftarrow [q2]Q$

[Test]
5. $q_3 \leftarrow 2^e$
6. $\overline{Q} \leftarrow Q$
7. Pour $k = 1, \ldots, T$
8. $\overline{Q} \leftarrow [q_3]\overline{Q}$
9. Pour $j = 1, \ldots, T$ Faire
10. Si $1 < \text{pgcd}(x_{\overline{Q}} - x_{Q_j}, N) < N$, afficher le pgcd et sortir.

FIG. 13 – Une deuxième phase du type *paradoxe des anniversaires*

Selon [3], il faudra choisir T le plus grand possible (de l'ordre de $\sqrt{B_3}$). Nous effectuons alors environ B_3 comparaisons de points, multiples du point Q.

A la fin de la première phase de l'algorithme, si aucun facteur n'a été trouvé, nous avons calculé :

$$Q = [p[1]^{s_1} \cdot \cdots \cdot p[2]^{s_k}]P$$

avec $p_i^{s_i} \leq B_1 < p_i^{s_i+1}$. Notons g l'ordre de $E(\mathbb{F}_p)$ et $g_1 > g_2 > \ldots$ ses facteurs premiers. Supposons que $g_2 < B_1$, et que la puissance de g_1 dans la décomposition de g est seulement de 1. Dans ce cas, à la fin de la première phase, notre point Q est un multiple du point $[\frac{g}{g_1}]P$.

Dans la deuxième phase nous calculons deux tableaux de points du type $[n]Q$ et $[n']Q$. Si $\mid n \pm n' \mid$ est un multiple de g_1, Alors le point $R = [|(n+n')(n-n')|]Q$ est un multiple du point $[g]P$. Dans ce cas, la deuxième phase de l'algorithme peut nous permettre de trouver un facteur de n.

Dans l'algorithme 11.1 nous avons $n = q_j^e$ et $n' = \overline{q_k}^e$ (avec $\overline{q_k} = 2^k$). On a alors :

$$\begin{aligned} g_1 \big| |n \pm n'| &\Leftrightarrow g_1 \big| |(n-n')(n+n')| \\ &\Leftrightarrow g_1 \big| |n^2 - {n'}^2| \\ &\Leftrightarrow g_1 \big| |q_j^{2e} - \overline{q_k}^{2e}| \\ &\Leftrightarrow q_j^{2e} - \overline{q_k}^{2e} \equiv 0 \pmod{g_1} \\ &\Leftrightarrow (q_j/\overline{q_k})^{2e} \equiv 1 \pmod{g_1} \end{aligned}$$

Or, le nombre de solutions de l'équation $x^{2e} \equiv 1 \pmod{g_1}$ est $\text{pgcd}(2e, g_1 - 1)$ (voir Annexe B). En choisissant e comme le produit de petits premiers, nous augmentons les chances de réussite de la deuxième phase de l'algorithme. Il ne devra pourtant pas être choisi trop grand, le temps de calcul en dépend fortement.

En pratique, Richard P. Brent (dans [3]) conseille de choisir e dans parmi $\{1, 2, 3, 6, 12, 24, 30\}$ sous la contrainte $32e < T$. Il propose aussi de choisir $B_2 \approx 100 \cdot B_1$, afin de ne pas passer plus de temps sur la deuxième phase de l'algorithme que sur la première.

12 Une réalisation de l'algorithme ECM

12.1 Les différentes versions de notre implémentation

L'évolution de notre implémentation de l'algorithme ECM ressemble à la progression de ce rapport. La première étape fut de réaliser la phase 1 de l'algorithme sans amélioration, exception faite de la *binary method* utilisée pour sommer et doubler des points. Le code correspondant est situé dans le répertoire `algo_ecm`. Le fichier `ecm_fact.c` implémente l'algorithme ECM proprement dit. Le fichier `ecm_lib.c` implémente les fonctions liées aux courbes elliptiques utilisées dans `ecm_fact.c`. Le fichier `ecm_lib.h` définit un certain nombre de variables et de constantes qui peuvent être modifiées dans certains cas très particuliers (par exemple si le facteur cherché est particulièrement grand). Cette répartition restera la même pour les nouvelles versions de l'algorithme.

La deuxième version implémente la deuxième phase décrite au paragraphe 9. Elle se situe dans le répertoire `algo_ecm_with_second_phase` . Une évolution importante de cette version se situe dans le répertoire `algo_ecm_parallel` . La méthode proposé par Montgomery (décrite au paragraphe 8) accélère la phase 1 de l'algorithme. Le gain en temps est assez remarquable.

La deuxième phase est alors successivement remplacée par les deux phases de type *paradoxe des anniversaires*. La première (dans le répertoire `algo_ecm_paradox`) n'a pas donnée de très bon résultat. La deuxième en revanche est celle qui nous a semblé la plus rapide. C'est cette dernière version que nous utilisons aujourd'hui.

Pour utiliser cette version de l'algorithme ECM, il faut se placer dans le répertoire `algo_ecm_paradox2` . Le nombre à factoriser doit se trouver dans un fichier (nommé par exemple `number.txt`). Si nous supposons que le facteur cherché fait 25 chiffres en base 10, la commande qui permettra de factoriser le nombre situer dans le fichier `number.txt`

sera :

```
./ecm_fact number.txt 64100 6410000
```

Le programme ne s'arrête que lorsqu'il trouve un facteur.

12.2 Quelques résultats expérimentaux

De nombreuses publications choisissent la deuxième borne B_2 de telle manière que le temps passé dans la deuxième phase soit environ deux fois plus petit que celui passé dans la première (voir [3], page 12). Si tel doit être le cas, la borne B_2 devra être choisie telle que $B_2 \approx 10 \cdot B_1$ pour notre implémentation (alors que la version implémentée par Richard P. Brent se contente de $B_2 \approx 100 \cdot B_1$). Pour donner un ordre d'idée, nous avons comparé certaines de nos moyennes avec la méthode ECM telle qu'elle est implémentée dans Maple6. Le tableau ci-dessous présente certains de nos résultats.

Taille du facteur (en nombre de décimales)	Notre implémentation	Maple	Pollard $p-1$	Pollard ρ
14	4.87	23.7	≈ 20	≈ 40
16	10.9	224		
17	44.5	306		
18	101	411		
19	102	458		

Comparatif de performances - Temps nécessaire pour trouver un facteur (en secondes) sur un Pentium III à 1.5 GHz

La première ligne nous permet de comparer la méthode ECM avec les deux méthodes de Pollard (implémentées par nos soins ...). Les résultats proposés ici sont optimistes (pour les méthodes de Pollard) en ce sens que nous n'avons jamais laissé l'algorithme tourner plus de dix minutes par chiffre à factoriser. Lorsque le facteur cherché est de l'ordre de 14 chiffres en base dix, environ trois chiffres sur quatre ont pu être factorisés avec les méthodes de Pollard.

Nous avons aussi testé l'efficacité de la deuxième phase de notre algorithme. Les résultats n'ont pas été aussi concluants que nous l'avions espéré. Le tableau ci-dessous permet de constater que la deuxième phase (utilisée avec $B_2 = 10 \cdot B_1$) augmente sensiblement le temps nécessaire pour trouver un facteur. En revanche les « records »de notre implémentation ont parfois été trouvés grâce à cette optimisation.

Taille du facteur (en bits)	Nombre de tests	sans deuxième phase	$B_2 = 10 \cdot B_1$	$B_2 = 100 \cdot B_1$
20	100	1.127	1.586	1.703
30	100	9.162	13.021	17.805
40	100	1m50	2m50	6m8
50	100	17m41	19m5	40m57
80	10	361m57	460m57	582m11

Comparatif des performances en fonction des paramètres des différentes phases

La dernière ligne de notre tableau n'est cependant pas très significative. Nous avons commis l'erreur de ne pas faire nos tests sur les même chiffres pour les différentes expériences. Lors de la factorisation des 10 chiffres de 80 bits avec deuxième phase ($B_2 = 10 \cdot B_1$), l'avant dernier chiffre n'a pu être factorisé qu'après environ 3 heures. Il fausse donc nos résultats puisque qu'aucun nombre similaire n'était venu gêner les tests effectués sans deuxième phase.

Et pour finir ...
Nous avons factorisé entièrement le onzième nombre de Fermat ($F_{11} = (2^2)^{11} + 1$),

$$F_{11} = 319489 \cdot 974849 \cdot 167985563417604751 37 \cdot 3560841906445833920513 \cdot p_{564}$$

où p_{564} est un nombre premier de 564 chiffres. Les plus grands facteurs premiers trouvés par notre implémentation faisaient 30 chiffres (trouvé en 8 heures grâce à la deuxième phase), 31 chiffres (trouvé en 33 heures), et enfin 32 chiffres (trouvé en 3 jours et demi). Ces résultats on été obtenus sur un portable tournant à 1.8Ghz.

13 Conclusion

Nous avons décrit l'algorithme ECM et de nombreuses améliorations que nous pouvons lui apporter. Ces dernières sont indispensables lorsqu'on s'attaque à des facteurs de plus de 30 chiffres. D'autre améliorations sont envisageables, nous n'avons pas pu les décrire toutes. Il est par exemple possible d'utiliser d'autres formes que celle proposée par Weierstrass. Il est alors possible de choisir les paramètres de la courbe de telle manière que l'ordre du groupe sous-jacent soit divisible par 12.

L'algorithme ECM permet de factoriser sans effort des nombres dont un des facteurs ferait entre 15 et 30 chiffres.

Remerciements :

J'aimerais remercier Serge Vaudenay d'avoir accepté ce projet et Jean Monnerat de m'avoir aidé à le réaliser.

Références

[1] Ian Blake, Gadiel Seroussi, and Nigel Smart. Elliptic Curves in Cryptography. Cambridge University Press, 1999.
Je me suis servi de ce livre comme référence. Particulièrement utile pour l'optimisation du calcul de $[k]P$.

[2] Richard P. Brent. Some integer factorization algorithms using elliptic curves. Technical report, Computer Science Laboratory, Autralian National University, September 1985.

[3] Richard P. Brent. Factorization of the tenth fermat number. Technical report, April 1998.
Une publication particulièrement complète. Description détaillée de l'algorithme ECM d'un point de vue pratique. Richard P. Brent propose aussi 3 suites différentes à l'algorithme de base. C'est sa description de la suite du type *Paradoxe des anniversaires* que nous avons utilisée ici.

[4] Henri Cohen. A Course in Computational Algebraic Number Theory. Springer, 2000.
Sans doute l'ouvrage qui m'a été le plus utile. Explication claire de la phase 1 et du calcul de la complexité de l'algorithme. Calculs en parallèle.

[5] Richard Crandall and Carl Pomerance. Prime Numbers, A Computational Perspective. Springer, 2001.

[6] Jr. H.W. Lenstra. Factoring integers with elliptic curves. Technical report, November 1987.
La publication qui a donné naissance aux suivantes ....

[7] Marc Joye. Introduction élémentaire à la théorie des courbes elliptiques, 1995. http ://www.dice.ucl.ac.be/crypto/
Une approche mathématique des courbes elliptiques. Peu de choses en ce qui concerne la méthode ECM.

[8] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. Handbook of Applied Cryptography. CRC Press, 1997.

[9] Steve Oualline. Practical C Programming, 3ed. O'Reilly, 1997.

[10] Joseph H. Silverman and John Tate. Rational Points on Elliptic Curves. Springer-Verlag, 1992.
Explications claires sur la géométrie projective et sur les aspects mathématiques concernant les courbes elliptiques.

A Tableaux de B_1

Nous proposons ici les valeurs de B_1 en fonction de la taille du facteur cherché. Nous n'indiquons pas le nombre attendu de courbes. Si l'algorithme ECM implémenté n'utilise que la phase 1, le nombre de courbe moyen nécessaire avant de trouver un facteur est de l'ordre de B_1. Si une deuxième phase est utilisée, ce nombre peut être sensiblement réduit.

taille de p en nombre de décimales	Valeur maximum de B_1
15	3'500
16	4'800
17	6'600
18	8'900
19	12'000
20	16'100
21	21'400
22	28'400
23	37'400
24	49'100
25	64'100
26	83'400
27	108'000
28	139'200
29	179'000
30	229'200
31	292'500
32	372'100
33	472'000
34	596'800
35	752'500
36	946'300
37	1'186'800
38	1'484'700
39	1'852'800
40	2'306'700

B Une petite preuve mathématique

Nous avons utilisé dans le paragraphe 11.2 le fait que le nombre de solutions de l'équation

$$x^{\alpha} \equiv 1 \pmod{p} \tag{8}$$

est $\text{pgcd}(\alpha, p-1)$ avec p un nombre premier et $\alpha \in \{1, \ldots, p-1\}$. Démontrons cette propriété :

Nous savons que $\mathbb{Z}_p$ est un groupe cyclique. Soit g un générateur de ce groupe. Nous allons considérer g^i pour $i \in \{1, \ldots, p-1\}$ et chercher combien de ces i permettent à g^i de vérifier l'équation 8. Nous avons :

$$\begin{aligned} (g^i)^{\alpha} \equiv 1 \pmod{p} &\Leftrightarrow g^{i\alpha} \equiv 1 \pmod{p} \\ &\Leftrightarrow i\alpha \equiv 0 \pmod{p-1} \end{aligned}$$

Replaçons nous dans $\mathbb{Z}$. Le plus petit i tel que $p-1 \mid i\alpha$ est :

$$\begin{aligned} i_0 &= \frac{\text{ppcm}(\alpha, p-1)}{\alpha} \\ &= \frac{\alpha \cdot (p-1)}{\alpha \cdot \text{pgcd}(\alpha, p-1)} \\ &= \frac{p-1}{\text{pgcd}(\alpha, p-1)} \end{aligned}$$

Les valeurs de $i \in \{1, \ldots, p-1\}$ telles que $i \cdot \alpha \equiv 0 \pmod{p-1}$ sont donc les $k \cdot i_0$ pour $k \in \{1, \text{pgcd}(\alpha, p-1)\}$. Il y a donc $\text{pgcd}(\alpha, p-1)$ solutions à l'équation 8.

□